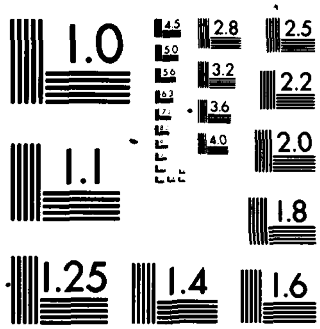
AD-A083 708 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/G 15/5
OPTIMIZATION OF MUNITIONS STORAGE.(U)
DEC 79 B A BOGGS, L M GUSMUS
UNCLASSIFIED AFIT/GSM/SM/79D-15 NL

1 OF 4



MICROCOPY RESOLUTION TEST CHART

AFIT/GSM/SM/79D-15

OPTIMIZATION OF MUNITIONS STORAGE

THESIS

Barton A. Boggs Louis M. Gusmus
Capt USAF Capt USAF

AFIT/GSM/SM/79D-15

Approved for public release; distribution unlimited.

AFIT/GSM/SM/79D-15

OPTIMIZATION OF MUNITIONS STORAGE

THESIS

Presented to the faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Master of Science

by

Barton A. Boggs	Louis M. Gusmus
Capt USAF	Capt USAF
Graduate Systems Management	Graduate Operations Research

December 1979

Approved for Public release; distribution unlimited.

Preface

This research effort was initiated as an outgrowth of a statement of need by the Air Staff for a computerized method for determining optimum storage of munitions. The model developed in this paper is capable of examining different objective functions for optimizing munitions storage. One objective function defined using storage load factors is examined in this paper. Solutions are computed for optimal storage such that volume, net explosive weight, and compatibility constraints are satisfied for each storage building and munition.

The main body of the thesis is complemented by the Munitions Storage Optimizing System (MSOS) User's Manual, Appendix A, which is written to be a "stand-alone" document. It provides detailed instruction and examples for the system described in this thesis. It is designed to provide the basic instructions needed to use the system developed here for optimizing munitions storage.

We hope that munitions planners will find this model to be useful; however, should it serve only as a solid starting point for future research efforts, we feel that our effort will have been worth while.

Our thanks go to LtC Frank Eubank, HQ USAF/LEYWC, and Maj Ray Shulstad, HQ USAF/XOFM, who helped to initiate this research by providing our initial indoctrination into the nature of the munitions storage problem. In addition, they provided some useful background documentation and helped us to establish some key points of contact for developing the research. We thank Mr Arlie Adams, HQ AFLC/IGYW, and LtC Sy Grimshaw, HQ AFLC/LOWM for their time and patience in answering questions and for providing some munitions technical information. We also wish to thank Mr Ira Saxton and Msgt Paul Grinter for sharing their expertise with the CREATE Honeywell computer system. Particular credit must be given to our Thesis Advisor, Col Charles R. Margenthaler, and to LtC John Hobbs, our Thesis Team Member, for their guidance, encouragement, and constructive criticism throughout the development and writing of this thesis.

Barton A. Boggs

Louis M. Gusmus

Contents

Abstract

This research is an effort to quantify the problem of how to store as much as possible of a required munitions inventory in the buildings of any given munitions storage area. The problem is addressed in this paper by the formulation of a mixed integer linear programming model that will calculate optimal storage subject to a complex set of constraints.

The Munitions Storage Optimization System (MSOS) provides a capability for setting up munition storage inventory linear programming problems. MSOS allows the user to create data bases containing the required information for the munition items and storage buildings. Munitions inventory is entered by stock number, number of lots, and number of packages for each lot. A program extracts the necessary information from the data bases, formulates the objective function and constraint equations, then submits the problem to a mixed integer linear programming package for solution.

Glossary

Building Density Factor--The maximum net explosive weight (NEW) for each class of munition that may be stored in a building divided by the useable building volume.

Building Load Coefficient--The building density factor divided by the the munition density factor.

Class/Division/Category--Class refers to the UN class of dangerous goods, i.e., Class 1, Explosives; Division refers to the four hazard divisions of explosives; and Category refers to four categories of fragment hazards. Class/Division/Category is sometimes referred to simply as "Class."

Compatibility Group--Munitions or explosives are divided into twelve compatibility groups designated by the letters A - H, J, K, L, and S. Items are considered compatible if they may be shipped or stored together without significantly increasing either the probability of a mishap, or for a given quantity, the magnitude of the effects of a mishap.

Munition Density Factor--The net explosive weight (NEW) of the munition package divided by the munition package volume.

Net Explosives Weight (NEW)--The total quantity, expressed in pounds of explosive material or high explosives equivalency in each munition item, to be used when applying quantity-distance standards.

Quantity-Distance (Q-D)--Quantity of explosives material and distance separation relationships which provide defined levels of protection. These relationships are tabulated in a series of Q-D tables.

Storage Area Load Factor--A measure of the overall munitions storage efficiency. It is composed of a combination of all the building load coefficients in the objective function of the optimization model.

I. Introduction

A major problem facing many Air Force installations is the storage of large quantities of munitions. Complex and demanding safety requirements for the protection of personnel and material in the event of an accidental explosion constrain how and where munitions may be stored. The need for protection from the elements and for security from theft or sabotage implies a requirement for indoor storage in properly designed facilities. Limitations on the quantity of explosives that may be stored in any one facility plus the requirement for large clear zones of land around each facility place a significant economic cost on the storage of munitions (Schreyer, 1970: 1).

Munitions storage is a subject that is often overlooked during times of peace. Upon the initiation of hostilities, however, the survival and fighting capabilities of the armed forces depend on the existing stockpiles of munitions in storage. Since the construction of new facilities, both stateside and overseas, is limited by budgetary constraints and the availability of land, it is vital to overall combat readiness that the USAF make optimal use of existing munition storage facilities (Cormier, 1979).

Conventional munition storage allocation methods rely primarily on charts, diagrams, manual computations, and experience to determine storage arrangements. This research is a first phased effort of applying the power of the computer to the problem of storing munitions.

A basic premise of this thesis effort is that significant improvements can possibly be realized through optimal or at least more efficient usage of existing storage facilities. Optimization, as used here, suggests some best allocation of munitions inventory to available storage volume used. A computerized munitions storage optimizing system is an attractive alternative to provide the user with the ability to build quantitative models to examine optimal ways of storing munitions.

Statement of Need

The need is to store a given munitions inventory efficiently in a limited set of munitions storage buildings without violating the quantity distance (Q-D) and compatibility group restrictions. The basic need can be narrowed to consideration of the location, size, and type of buildings available for storage at any particular storage area and determining an optimal storage arrangement. More simply, the need is determining how to efficiently store as much as possible of a required munitions inventory in the fewest number of storage magazines at a particular operational base.

The problem undertaken in this thesis, in response to the total perceived need, resulted in the development of the Munitions Storage Optimizing System (MSOS). MSOS provides for the creation of data bases to provide descriptive information about individual munition items and munitions storage buildings. The optimization process is initiated by entering a munitions inventory in terms of stock number, number of lots, and number of packages per lot to a program. The program then extracts needed information from the data bases, and generates the constraints to the problem and the objective function to be maximized. An optimal solution to the problem is then computed.

Research Objectives

The following objectives were established as a heirarchy of research goals to be persued in this thesis effort.

(1) Demonstrate a general approach for the application of mixed integer linear programming to the problem of optimizing munitions storage.

(2) Development of a working model for use as a tool by munitions managers to derive solutions for improved or more efficient storage allocation of munitions.

(3) Development of a computer-based procedure for formulation of a munitions storage optimization problem for accomplishment with the LP/600 mixed integer linear programming routine designed by Honeywell Information Systems, Inc. This procedure is designed to allow the user to simply input required storage and munitions data to the optimizing system. The computer then creates the objective function and constraints for the problem.

(4) Exploration of a number of alternate munition storage performance measures. The measure examined in this paper is a storage area load factor (Farwell,1970: 4). This factor is discussed in Chapter III, The Munitions Storage Optimizing System.

Organization of Thesis

The actual development of the model for the optimization of munitions storage begins with a discussion of the munitions storage system. This discussion begins by briefly explaining explosive safety criteria, then branches out into a description of the United Nations Classification System, briefly summarizes some of typical storage facilities currently in use, and concludes by describing some of the general aspects of storing munitions. Chapter III, The Munitions Storage Optimizing System, explains the actual mathematical formulation of the problem, provides some general technical discussion, and outlines the objectives, scope, and limitations of the model. Several sample problems are presented in Chapter IV, Model Verification, to verify the actual performance of the munitions storage model. The writers' views on ways to develop more sophisticated optimization models are presented in chapter V, Enhancements to the Model. Chapter VI, Summary, Conclusions and Recommendations, describes the overall results of this research effort and compares the heuristic approach used in this paper with desired real world solutions. The Munitions Storage Optimizing System User's Manual (Appendix A) and the Program Documentation (Appendix B) are of critical inportance in developing a complete understanding of the work that has been accomplished. These appendices explain the programs and data bases, and take the reader step by step through the processes required to use the Munitions Storage Optimizing System (MSOS).

II. The Munition Storage System

Munitions storage is a highly complex process because of the numerous Q-D and compatibility group constraints and the drive for indoor storage. These constraints are the result of a concerted effort to restrict possible losses of people, facilities, and inventories in the unlikely event of an accidental explosion. Present methods for determining allocations of munitions to storage appear to lend themselves to computer model applications. This chapter discusses the literature reviewed during this research effort and summarizes some of the background material pertaining to the munitions storage system.

Literature Search

The research was conducted primarily through a review of the literature available at the Air Force Institute of Technology library, the Defense Documentation Center (DDC), and at the Air Force Logistics Command explosives safety office. Contacts were made to munitions managers at several Air Force Major Commands and to the Conventional Munitions Branch of Headquarters, USAF. These contacts resulted in expressions of interest in the subject research and helped to confirm the fact that little has been done in the past to address optimization of munitions storage through the use of the computer.

Much of the basic material used in this research effort has been gleaned from Air Force publications such AFR 127-100, Explosives Safety Standards, and the Air Force 11A-series munitions technical orders. Other sources of background material that deserve mention are the minutes of the Explosives Safety Seminars sponsored biannually by the Department of Defense Explosives Safety Board (DDESB). These documents, available in microfiche from DDC, are comprehensive compilations of reports, studies, and discussions concerning topics that range from historical developments to the latest results from research and testing of munitions.

Air Force technical orders, TO 11A-1-61-3 and TO 11A-1-61-4, provide information, charts, and drawings for efficient storage of a number of Federal Stock Numbered munition items. They depict a process that requires a great deal of manual calculation for determining mixed storage of munitions. While these documents are probably highly useful to the munitions manager, they do not provide a systematic method for optimizing storage when more than one munition item at a time is considered.

One of the most useful sources that has had a bearing on the development of this research is a paper, "The Munitions Storage Problem," written by David G. Farwell 1970 at the University of Colorado. Farwell describes a linear programming model that can be used to calculate optimal storage of munitions from a single class and compatibility group combination (Farwell, 1970: 1-4). One of Farwell's recommendations was to develop an expanded model to simultaneously examine combinations of compatibility groups within the same model (Farwell,1970:18). The ideas and recommendations in Farwell's paper contribute to the foundation of the MSOS model presented here.

A recurrent theme throughout the literature on munitions storage is that safety is the driving factor and that the rigid standards for storage are a result of the high emphasis on maintaining accident-free storage areas.

Explosives Safety

Haphazard storage and transportation procedures have occasionally resulted in explosive disasters. One of the worst was an incident that occurred at the Naval Ammunition Depot, Lake Demark, New Jersey in 1926. Lightning struck some stored munitions and set off a series of explosions that destroyed a major portion of the depot and killed a number of people. One of the magazines at Lake Demark contained about three times the amount of high explosive permitted in a single storage location by today's standards. Thus, when the explosion occurred, the failure to adequately separate explosives allowed a chain reaction of explosions to spread from one magazine to another (Fliakas, 1976: 1).

This disaster set in motion a chain of events that resulted in the establishment by Congress of the Ammunition Storage Board, the beginning of what now is the Department of Defense Explosives Safety Board (DDESB). As an outgrowth of the investigation of the Lake Demark disaster, Congress eventually directed the adoption of new Q-D tables which were designed to regulate proper separation of ammunition supplies in order to provide a reasonable degree of safety. Congress also defined 'safe' within the context of munitions storage.

> As regards to the 'safety' of individuals and structures outside of ammunition depots, the word 'safety' is a relative term. No one is ever absolutely safe from injury. The average chance of the average individual escaping injury has, by custom, been termed 'safe' (Fliakas 1976: 2).

This definition has generally been held to mean that some specified hazard distance must be maintained between explosive storage points and other activities to provide a reasonable degree of safety in the event of an explosives accident. The probability that an accident will actually occur is a factor that is generally not considered (Fliakas 1978: 11). This has resulted in extremely rigid standards that are geared to keeping the worst imaginable losses to a minimum acceptable level.

The basic methods that have been developed to insure safe storage of munitions involve restrictions on the NEW that can be stored in any one building and the use of large amounts of land to allow for separation between buildings and clear areas around the entire munitions storage area. In addition, only munitions that are compatible according to standardized compatibility group criteria may be stored together in any one location. The definitions and criteria for identifying hazard classification, divisions, compatibility groups, and Q-D tables were recently standardized in the United States by the adoption of the United Nations system of classification of explosives.

The United Nations Classification System

In the past, different countries have set up their own criteria for explosives safety based on subjective as well as empirical evaluation of explosive hazards. This has sometimes resulted in confusion and difficulty in the transport and storage of munitions from one country to another. In recent years there has been a trend to develop international agreement on standards for shipment and storage of dangerous materials. Taking a lead in this area, the United Nations set up an international group to develop standards that would be acceptable world-wide. By 1977, the UN classification system had been adopted by the United States, United Kingdom, and many of the NATO countries

(Lyman, 1978: 129). For the United States this involved relatively minor changes in coding and terminology because the system previously in use in this country converted almost directly to the UN system (AFR 127-100, 1978: 5-4).

Classes of Dangerous Goods. The system developed by the UN for international use is designed to meet a wide range of military and commercial requirements. It consists of the following nine classes of dangerous goods (Adams, 1976:1574).

UN Class	Description
1	Explosives
2	Compressed or liquified gasses
3	Inflammable liquids
4	Inflammable solids
5	Oxidizing substances
6	Poisonous (toxic) and infectious substances
7	Radioactive substances
8	Corrosives
9	Miscellaneous dangerous substances

Class 1, Explosives. Most of the conventional munitions used by the armed forces are listed under Class 1, Explosives. Class 1 has been subdivided into four divisions (1.1, 1.2, 1.3, and 1.4) which indicate four different types of hazards. Class/division 1.2 is further subdivided into four categories that provide different hazard distances for each category. The divisions, categories, and the type of hazards associated with each are sumarized below (Lyman, 1978: 130).

Class/Division/Category	Hazard
1.1	Mass detonation (blast) with possible fragment threat.
1.2 (18) 1.2 (12) 1.2 (08) 1.2 (04)	Non mass detonating with most fragments falling within the distance indicated, i.e., (18), (12), (08), and (04) indicate hazard distance of 1800, 1200, 800, and 400 feet respectively.
1.3	Mass fire.
1.4	Moderate fire--no significant hazard

Quantity-Distance Criteria. Quantity-distance (Q-D) Criteria is based on a relationship between quantities of explosives from any class/division/category that may be stored in a single storage point and the separation distances required to provide a reasonable degree of protection in the event that a certain amount of explosives were to detonate (Fliakas, 1978: 11). The exact relationship between these distances and the maximum net explosives weight (NEW) that can be stored in any one building for each class of munition is specified in a series of Q-D tables. These Q-D tables specify the Inhabited Building Distance (IBD), Public Traffic Route Distance (PTR), Intermagazine Distance, and Intraline Distance for the seven classes of munitions. The maximum explosive weight that may be stored is limited by the most restrictive condition from any one of the tables. The Q-D tables currently used by the Air Force can be found in AFR 127-100, chapter 5, Principles and Applications of Explosive Quantity-Distance Criteria and Related Standards. These Q-D tables are briefly described below.

(1) The inhabited building distance (IBD) table defines the minimum permissible distance allowed between a quantity of explosives and any building inhabited by the public or where people are accustomed

to assemble, both within and outside of government establishments.

(2) The Public Traffic Route (PTR) Distance table prescribes the minimum permissible distance between an explosives site and public highways, railroad lines, or even navigable streams. It is approximately 60% of the IBD because it is reasoned that traffic receives only limited exposure as it passes by the explosives site.

(3) The Intraline Distance table specifies the distance to be maintained between any two operating buildings, at least one of which is designed to contain explosives.

(4) The Intermagazine Distance table defines the minimum permissible distance between storage magazines, and is based primarily on the type of magazine and the quantity of explosive involved.

(5) The Fragment Distance tables apply to specific explosive items which generate hazardous fragments. This primarily describes class 1.2 items which have prescribed hazard distances of 400, 800, 1200, and 1800 feet.

Typical munitions storage buildings are often designed so that the limitations imposed by the Q-D tables match the maximum allowable NEW for the facility, i.e., a storage building containing 1.1 munitions is probably positioned to allow the maximum storage of 500,000 pounds NEW. Variations in the tables occur, depending on whether the storage facilities are standard or non-standard, earth covered or aboveground, and barricaded or non-barricaded (AFR 127-100, 1978: 5-6 to 5-12).

<u>Compatibility Groups</u>. Ideal storage of munitions would probably require each individual class or type of munition to be stored separately. Since a limited storage availability usually prevents this from occurring, other factors have been specified that permit various

types of munitions to be stored together providing they are considered compatible. The basic factors that determine compatibility are (AFR 127-100: 4-10):

(1) Chemical and physical properties.

(2) Design characteristics.

(3) Inner and outer packaging configurations.

(4) Q-D class/division.

(5) Net explosive weight (NEW).

(6) Rate of deterioration.

(7) Sensitivity to initiation.

(8) Effects of deflagration, explosion, or detonation.

Based on evaluation of these factors, ammunition and explosives are assigned to one of twelve storage compatibility groups (A through H, J, K, L, and S). These groups are identified and defined below (AFR 127-100, 1978: 4-11).

> Group A--Initiating Explosives. These are bulk initiating explosives with the necessary sensitivity to heat, friction, or percussion that makes them suitable as initiating elements in an explosive train.
>
> Group B--Detonaters and Similar Initiating Devices. These items contain initiating explosives and are designed to start or continue the functioning of an explosive train.
>
> Group C--Bulk Propellents, Propelling Charges, and Devices Containing Propellent, With or Without Their Means of Initiation. These items will deflagrate, explode, or detonate upon initiation.
>
> Group D--Black Powder, High Explosives (HE), and Ammunition Containing HE Without Its Own Means of Initiation and Without Propelling Charge. These are items such as bombs, projectiles, or bulk TNT that can be expected to explode when any given item or component thereof is initiated.
>
> Group E--Ammunition Containing HE Without Its Own Means of Initiation and With Propelling Charge. Examples are artillery

ammunition, rockets, and guided missiles.

Group F--Ammunition Containing HE With Its Own Means of Initiation and With or Without a Propelling Charge. Examples are some hand and rifle grenades.

Group G--Fireworks, Illuminating, Incendiary, Smoke, or Tear-Producing Munitions Other Than Those That Are Water-Activated, or Which Contain White Phosphorus, or Flammable Liquid, or Gel. Examples are flares, signals, incendiary, smoke, and tear-producing devices.

Group H--Ammunition Containing Both Explosives and White Phosphorus or Other Pyrophoric Material. These munitions contain items that are spontaneously flammable when exposed to the atmosphere.

Group J--Ammunition Containing Both Explosives and Flammable Liquids or Gels. This group includes items such as napalm-filled fire bombs.

Group K--Ammunition Containing Both Explosives and Toxic Chemical Agents. These items contain chemicals designed to produce severe incapacitating effects.

Group L--Ammunition Not Included in Other Compatibility Groups. This group includes water-activated devices, prepackaged hypergolic liquid-fueled rocket engines, and damaged or suspect ammunition or explosives from other groups. Items within Group L may not necessarily be stored together.

Group S--Ammunition Presenting No Significant Hazard. These are items that are packaged or designed to confine hazardous effects arising from accidental functioning within the package.

In general, munitions from any given compatibility group are not to be stored with any items from any other group; however, there are some limited cases where the regulations permit combined storage; for example, groups D and E may be stored together (AFR 127-100, 1978: 4-12). <u>This paper does not address these limited cases and, therefore, it requires munitions from each compatibility group to be stored only with other munitions that have an identical compatibility group</u>.

Munition Storage Facilities

Munitions storage facilities, commonly referred to as magazines, are composed of virtually any type of structure that has been designated as a storage point for explosive devices. Most of the magazines in use today fall into two types--igloo (earth-covered) and aboveground (nonearth-covered).

Igloo Magazines. The various types of standard igloos now in use all have certain features in common (Wight, 1978: 243).

(1) A circular or oval arched roof made of steel reinforced concrete to cover the stored contents.

(2) An earth covering to contain fragments (The covering serves to effectively barricade three sides of the igloo).

(3) A concrete slab floor.

(4) Concrete head and side walls.

(5) A structural steel door.

(6) Dimensions of 24-27 feet wide, 40 to 80 feet long, and 12 to 16 feet high (There are a few in use with significantly different dimensions).

Igloos are designed so that the force of an accidental explosion is directed upward through the roof. This feature has earned the igloo a reputation as a particularly safe type of storage facility.

The design also lends itself readily to providing good security against theft or sabotage. For example, points of entry are few, the door is easily secured and locked, and forced entry is time consuming and requires special tools. These factors, when coupled with other security measures such as fenced, lighted, and patrolled storage areas, provide a high degree of security (Wight, 1978: 246).

Aboveground Magazines. Standard aboveground magazines are generally of concrete and steel construction and come in a variety of types and sizes. They range from multicubicle magazines composed of a number of small storage compartments to large standard magazines. If they are unbarricaded, the aboveground magazines require greater safety distances than those required for the better protected igloos. In some cases, special restrictions have been specified; for example, the multicubicle buildings are restricted by an explosive weight limit of 425 pounds of mass detonating explosives (AFR 127-100, 1978: 4-10).

Principles of Storage

Great care must be taken to insure that the conditions for proper and safe storage of explosives are not compromised. When the physical and chemical properties of stored materials are neglected, ammunition may begin to deteriorate rapidly or be exposed to the risk of fire or explosion (Fliakas, 1978: 6). Air Force Technical Order 11A-1-61-4, Storage and Loading Instructions, provides some general rules for proper storage. The following items serve to exemplify some of the guidelines normally applied to munitions storage (TO 11A-1-61-4, 1976: 3-1, 3-3).

(1) Lots will not be mixed in storage, but only to the extent that items of one lot will not be stacked on top of items from another lot (A lot is a quantity of identical munitions manufactured during the same production run or as otherwise labeled by the manufacturer).

(2) Stored munitions, or their containers will not contact a wall of a magazine (this requirement is designed to insure that there is adequate circulation of air around stored munitions).

(3) The stacking height limit of the specific item or type of container will not be exceeded.

(4) Palletized loads

(A) should not exceed 2,000 pounds unless required to do so by specific technical data or by waiver.

(B) should not exceed 44 inches in length, 54 inches in width, and 54 inches in height (there are many exceptions).

These principles have resulted in a number of standardized package and pallet configurations for munitions items. This allows many munition package sizes to be standardized quite easily in computerized computations.

III. The Munitions Storage Optimizing System

The methodology used to attack the problem of optimizing the storage of munitions is an application of mixed integer linear programming. Mixed integer programming allows both integer and continuous variables to be included in the model. This permits munition storage problems to vary the set of constraint equations that are used, depending on the classes of explosives to be stored in each building.

The actual problem of munitions storage is addressed in this thesis by the development of the Munitions Storage Optimization System (MSOS). This system is used to construct the problem to be solved with the LP/600 mixed integer linear programming package produced by Honeywell Information Systems, Inc. The branch and bound method is a general approach for optimization problems which aims to conduct an intelligent search through all possible solutions, cutting out large groups of possibilities early in the search. It depends on being able to judge in advance which directions of search can be eliminated (Nicholson, 1971: 138-141). One limitation to this method is that the amount of computation time required to reach a solution increases exponentially as the problem size gets larger. Thus, solutions to large munitions optimization problems will require a great deal of computer time. The remainder of this chapter will discuss the MSOS and develop the mathematical model used to optimize munitions storage.

Munitions Storage Optimizing System

The Munitions Storage Optimizing System (MSOS) is designed to allow the user to combine information about the storage facilities to be used and the munitions to be stored to generate the objective function and constraints in proper format for input to the LP/600 linear programming package. The MSOS allows the user to create and maintain three data bases; the Standard Building Data Base (SBDB), the Munitions Storage Area Data Base (MSADB), and the National Stock Number Data Base (NSNDB). Data from each of these data bases is used by the Format Generator Program (LPGEN) to formulate the constraints and objective function.

Standard Building Data Base. The Standard Building Data Base (SBDB) contains information that describes different types of munitions storage buildings. This means that buildings having identical dimensional and structural characteristics will be identified as the same type. The following information must be entered into the data base for each type of building.

(1) Type of building (any number from 1 - 99 as defined by the user is acceptable).

(2) Type of roof (RND is used to identify arched buildings; Flt is used to identify rectangular shaped buildings; other shapes must be estimated using RND or FLT).

(3) Length (inner length of the building in feet).

(4) Width (inner width of the building in feet).

(5) Height (inner height of the building in feet).

(6) Side wall height (height of the straight part of the side walls in feet--for igloo type magazines, if applicable).

(7) Radius (radius of the curved roof of igloo type buildings).

The SBDB is set up to request additional descriptive information besides the items shown, however, the items above are the only ones used in the current model. The SBDB needs to be constructed once for each munition storage area. If new types of buildings are constructed it is a simple process to modify the data base. Information from the SBDB is used to generate the volume constraints for the model. The calculated volumes are used in combination with the NEW constraints to calculate building load coefficients to be used in the objective function. Detailed information describing this data base is contained in appendix A, User's Manual.

Munition Storage Area Data Base. The Munition Storage Area Data Base (MSADB) contains the maximum allowable net explosive weight (NEW) for each class/category/division and the maximum gross weight that can be stored in every storage building in the munitions storage area. The maximum NEW for each class/cat/div must be calculated by the user based on the Q-D tables in chapter five of AFR 127-100. The User's Manual provides a detailed description of this data base and also includes a form that should be helpful in determining and arranging the data that will be requested when creating records for the MSADB. Like the SBDB, the MSADB needs to be created once and only needs to be modified as new buildings are constructed, destroyed, or when the Q-D tables are changed. The following information must be entered into this data base.

(1) Building number.

(2) Type (a number between 1 and 99 as defined in the SBDB).

(3) 1.1 NEW (the maximum NEW for class 1.1 authorized to be stored in this building).

(4) 1.2/04 NEW (the maximum NEW for class 1.2, category 04 munitions authorized to be stored in this building).

(5) 1.2/08 NEW (the maximum NEW for class 1.2, category 08 munitions authorized for storage in this building).

(6) 1.2/12 NEW (the maximum NEW for class 1.2, category 12 munitions authorized to be stored in this building).

(7) 1.2/18 NEW (the maximum NEW for class 1.2, category 18 munitions authorized to be stored in this building).

(8) 1.3 NEW (the maximum NEW for class 1.3 munitions that can be stored in this building).

(9) 1.4 NEW (the maximum NEW for class 1.4 munitions that can be stored in this building).

National Stock Number Data Base. Detailed information about individual munition items is stored in the National Stock Number Data Base (NSNDB). The following information must be entered into the NSNDB for at least those munitions that will be examined using MSOS for storage. Ideally, all the munitions identified in the 1300 and 1400 series Air Force Item Listings could be entered.

(1) Stock number (18 character National Stock Number).

(2) Package Height (munition package height in feet).

(3) Package width (munition package width in feet).

(4) Package length (munition package length in feet).

(5) Units per package (number of individual munitions in this package).

(6) Compatibility group (compatibility group assigned to this munition).

(7) Class/Division (munition class and division, i.e., 1.1).

(8) Category (for 1.2 munitions; inputs are 04, 08, 12, and 18).

The user has the capability to add, delete, and modify records within the data base. Each stock number represents a particular munition-package combination so it is possible for a given munition item to be represented by several stock numbers. Once the correct data has been loaded in the SBDB, MSADB, and NSNDB the user is ready to determine the storage configuration of a given munitions inventory.

Format Generator Program. The Format Generator Program (LPGEN) allows the user to enter the following information.

(1) Stock number of munition to be entered in inventory.

(2) Number of lots for munition identified in (1). Values from 1 to 99 are valid.

(3) Number of packages for each lot.

(4) A percentage of useable volume can be entered for each building in the munitions storage area or one value can be entered and applied to all buildings.

Once this data has been entered, LPGEN then selects the appropriate information from each of the three data bases described above and generates the objective function and the constraints for entry into the Honeywell LP/600 package. This program receives a comprehensive discussion in the User's Manual.

The MSOS Optimization Model

The formulation of the MSOS optimization model required the creation of a complex system of computer programs to bring the information from each of the data bases into the model. For example, the set of constraints specifying the maximum NEW that can be stored in each building can vary, depending on the most restrictive class of munitions actually stored. Thus the optimizing model provides a capability to vary the maximum NEW that may be stored in each building during the optimizing process. A technique was developed to provide for the complex and changining constraints using the mixed integer mode of the LP/600 package. The mathematical formulation of the model is discussed later.

The development of the optimization model requires the understanding of several terms critical to its mathematical formulation. They are:

(1) Subgroup. Munitions from different classes may be stored together providing they are from the same compatibility group. For a given munitions inventory, each identified group may contain 0 to 7 different classes of munitions. Every storage building has a NEW limit assigned for each of the seven classes and the limit actually used for a particular building is the NEW limit assigned for the most restrictive class of munition stored in the building. For example, an inventory of group D munitions belonging to classes 1.1, 1.2/04, and 1.3 require storage. The maximum NEW limits for a particular storage building are defined for this example to be: 1.1--100,000 pounds, 1.2/04--250.000 pounds, and 1.3--500,000 pounds. Storage of any combination of these munitions containing at least one 1.1 munition places the maximum NEW storage limit for the building at 100,000 pounds. If only 1.2/04 and 1.3 items are to be stored, the limit is 250,000 pounds. Similarly, if only 1.3 munitions are to be stored, the limit is 500,000 pounds. With this background, the definition of subgroup is offered. A subgroup is defined as a combination of munitions classes such that the first subgroup of a compatibility group contains all classes of munitions to be stored from that group; the next subgroup eliminates the most restrictive class from the group. Subsequent subgroups continue this

process until all seven classes have been eliminated. Thus for each compatibility group, a maximum of seven subgroups may occur. Since there are twelve compatibility groups, a maximum of 84 subgroups can occur in the model.

(2) Useable Building Volume. LPGEN calculates the total inside storage volume for each building and requests the user to enter a subjective estimate of the percentage of this total volume considered useable. This useable building volume is used both as the storage volume constraint for each building and for calculating building load coefficients.

(3) Building Load Coefficients. A building load coefficient is composed of the building density divided by the munition density. The building density is the subgroup NEW storage limit of the building divided by the the useable building volume. The munition density is the munition package NEW divided by the munition package volume. A particular munition may be identified with more than one subgroup, and therefore, be identified with the same building more than once. In each case, however, the maximum NEW limit for the building may be different, resulting in different building load coefficients being calculated for the same munition.

(4) Storage Area Load Factor. The storage area load factor is the factor that is maximized by the optimization process. It provides an overall measure of the building load coefficients for the entire munitions storage area. It is essentially a measure of how densely the munitions inventory is loaded into the munitions storage area. The density of the stored munitions is positively related to the storage area load factor. This means that the larger the value of the storage

area load factor, the more efficient the use of the munitions storage area. The formulation of the objective function is such that if there are munitions that are not stored because the NEW limit is reached or the volume used up, they are assigned a negative coefficient. Thus negative values of the objective function are possible.

Now that these terms have been explained, the actual mathematical optimization model is presented.

Mathematical Formulation

For the specific case of examining optimization where the storage area load factor is maximized, the problem is formulated as a mixed integer linear programming model that is designed to:

Maximize the Storage Area Load Factor: OBJECTIVE =

$$\sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=1}^{l} c_{ijk}\ x_{ijk} + \sum_{j=1}^{n} \sum_{k=1}^{l} d_{jk}\ y_{jk} + \sum_{i=1}^{m} M_i\ z_i$$

Subject to:

Munition Constraints $\sum_{j=1}^{n} \sum_{k=1}^{l} x_{ijk} + z_i = I_i$ for $i=1,m$

(one for each munition/lot combination)

Building Volume Constraints $\sum_{i=1}^{m} \sum_{k=1}^{l} a_{ijk}\ x_{ijk} \leq V_j$ for $j=1,n$

(one for each building)

NEW Constraints $\sum_{i=1}^{m} \sum_{k=1}^{l} b_{ijk}\ x_{ijk} - N_{jk}\ y_{jk} \leq 0$ for $j=1,n$

(one for each subgroup/building combination)

Special Set Constraints $\sum_{k=1}^{l} y_{jk} = 1$ for $j=1,n$

(one for each building)

all $x_{ijk} \geq 0$

all $z_i \geq 0$

all $y_{jk} = 0$ or 1

where:

a_{ijk} is the volume of munition i in subgroup k to be stored in building j

b_{ijk} is the NEW of package of munition i in subgroup k to be stored in building j

c_{ijk} is the building j load coefficient of munition i in subgroup k

d_{jk} is a weight assigned to the subgroup k / building j combination (initialized to zero for this model)

M_i is a weight assigned to packages of munition i that cannot be stored in the storage area (initialized to -1 for this model)

x_{ijk} is the number of packages of munition i in subgroup k to be stored in building j

y_{jk} is the bivalent (0 or 1) special set variable assigned to the subgroup k / building j combination

z_i is the left over variable assigned to munition i

I_i is the number of packages in munition/lot combination i

N_{jk} is the maximum NEW that can be stored in building j for subgroup k

V_j is the maximum useable volume of building j

l is the number of different subgroups in the inventory

m is the number of munition/lot combinations

n is the number of buildings

The d_{jk}'s defined above could be set up as a priority scheme with priorities (weights) for assigning compatibility groups and classess within the compatibility groups to storage buildings or to the overall storage area. The M_i's could be used to establish a priority scheme to determine a rank order of munitions to be excluded from storage in the event that the entire inventory cannot be stored inside the storage area buildings. Neither of these priortiy schemes are addressed in this paper.

Input Parameters

Formulation of the model has required certain input parameters to be defined in an attempt to insure that inputs to the MSOS receive standard treatment.

(1) Each munition/lot combination is entered to the program as a separate munition.

(2) Each munition/lot combination is considered to be of equal importance. This means that each munition receives no special treatment in comparison to any other munition as it is evaluated with the MSOS.

(3) All classes of munitions within a given compatibility group may be stored together. In cases where individual items require separate storage, they should not be evaluated with MSOS. For example, there are certain items from Groups K and L that require special considerations for storage.

(4) No particular prioritization schedule for storage has been established. The storage allocation decisions are an output of the linear programming model.

(5) Munitions must have a packaged NEW of greater than or equal to 0.00005 pounds to be evaluated by this model. LPGEN excludes any munition having a packaged NEW of less than 0.00005 because the calculation for the munition density factor uses the package NEW in the denominator and values less than 0.00005 are truncated to zero. This parameter excludes a large number of inert items from Group S from

consideration by the model.

(6) Individually stock numbered items are considered rather than complete assembled items unless the assembled end item is assigned its own stock number. For example, rocket warheads and rocket engines would be evaluated separately unless the assembled warhead/engine combination is identified by a single stock number.

(7) Each munition package is considered to be a rectangular box for calculations of volume. Thus, even a round iron bomb must be defined in terms of length, width, and height.

(8) Useable storage volume is a subjective input to the MSADB. For this factor to be a reasonable approximation of a real world situation, the following items are considered.

(A) Required aisle space.

(B) Placement of rectangular packages in arched buildings.

(C) Amount of clear space required around packages.

(D) Space required by dunnage.

(E) Allowable stacking height.

(F) Space for transient shipments or unserviceable items.

(9) Only conventional munitions are considered.

Limitations.

This study addresses the issue of storage from an abstract approach that centers on the descriptions of the munitions and storage facilities based on building load coefficients. It does not consider other aspects that may be equally important in real world situations. For example, storage may involve a sizeable threat from either theft or sabotage. The computer solution could conceivably place a particularly sensitive item in a facility with limited protection, or simply place the entire stock of a munition in one location where risk of loss of the entire inventory would be unacceptable. It may be that the optimal utilization of storage space as defined by the storage area load factors could be less desirable than some other combination of safety and reasonably effective storage. Thus, subjective evaluation of the computer output is required.

Some of the more specific limitations to the model are identified below.

(1) Optimality. The definition of what constitutes "optimal" storage may vary, depending on local objectives. There are any number of alternative maximization or minimization problems that could be defined using an objective function and constraint equations that differ from the ones used with MSOS. For example, a vast difference could exist between optimizing an objective of storing the greatest amount of munitions in a given storage area and a differing objective of optimizing combat readiness in terms of transfer time from storage to a weapons delivery system.

(2) Problem Size. The size of the problem matrix that can be used with the LP/600 package is limited to 262,000 decision variables (columns) and 4095 constraint equations (rows). Considering that the variables in the problem increase by the product of possible munition and building combinations plus some additional variables and the large number of constraints that are generated, it is easy to generate problems that exceed the capacity of the LP/600 package. In actuality, this is not that great of a limitation because many reasonably sized base level problems may be examined.

(3) Priority for Storage. This model does not develop any type of

priority scheme for storage. In the situation where there are more munitions to be stored than there is indoor storage space it might be desireable to have a systematic way to optimally determine which items should be stored inside and which should be left over for outdoor or alternate temporary storage.

(4) Useable Volume. The current model maximizes the storage area load factor which is based in part on the useable volume as defined by the user. Useable volume only subjectively addresses stack heights, lot integrity, aisle space, and possibly other factors that may be critical to actual storage layouts. One problem with this input for useable volume is that it must be defined by the user and entered into the LPGEN before the user knows what will be stored in each building. In reality, the useable volume is dependent on the specific combination of munitions to be stored in any given building. At best this input will be an educated guess. Of course, sensitivity analysis based on varying the useable volume input can yield a better understanding of the storage interrelationships.

Scope

The present model is limited in scope to base level, single storage area problems. It is essentially a heuristic approach that uses storage load factors to suggest optimal storage of munitions. It does not prescribe the actual placement and configuration of the items for storage, however, the solution does allocate munitions to buildings and satisfies the volume, NEW, and compatibility constraints.

This approach should be considered a tool for developing greater insight in the overall problem of storing munitions. An area where it might be particularly useful is in providing a type of sensitivity analysis to changes in the munitions storage "status quo" such as the effects of waivers, changes in inventories, and changes to the numbers or types of buildings to be used.

IV. Model Verification

A series of tests are developed to verify whether MSOS actually functions as intended. These tests are not an attempt to validate the model for optimization of a real munitions storage problem. Real world validation is left for a future effort.

The verification tests described in this chapter are based on a hypothetical munitions storage area that is used to store various combinations of hypothetical munitions. The munitions are defined to represent a mix of compatibility groups, classes, and NEW. The values are chosen in a deliberate attempt to ease the verification process. The objective is to examine the actual results of MSOS for a series of tests that progress from a simple one munition, one lot test to combinations of up to eight munitions, eight lots, three compatibility groups, and three classes.

The test data was stored in the SBDB, MSADB, and NSNDB. Each test was then accomplished by entering the stocknumber, number of lots, and number of packages for each lot using the LPGEN program. LPGEN then generated the objective function and constraint equations and submitted each problem to the LP/600 mixed integer linear program for determination of the optimal solution.

Verification Test Data

The test data that was selected and entered into the SBDB, MSADB, and NSNDB for the verification tests is listed below.

(1) The SBDB contains the description of one type of standard

building.

Length	Width	Height	Total Cube
10′	10′	10′	1000 cu ft

Although this standard building is not a typical storage building, its total cube will permit easy examination of volume used in each test.

(2) The MSADB lists the NEW storage constraints in pounds for five standard buildings that make up the munitions storage area. NEW constraints for all seven classes was actually entered into the data base, however, only the NEW constraints for the three classes used in the tests are shown below.

Maximum NEW Storage Constraints

	Class/Division/Category		
Bldg	1.1	1.2/4	1.3
1	1000	1000	1000
2	100	500	600
3	1000	5000	6000
4	500	2500	3000
5	1000	1000	1000

(3) The NSNDB is composed of eight types of munitions which represent three compatibility groups and three classes.

		Package Dimentions			Package
Munition	Class/Group	Height	Length	Width	NEW
1	1.1 D	2′	2′	2′	10
2	1.1 G	2′	2′	2′	10
3	1.1 D	2′	2′	2′	01
4	1.3 D	2′	2′	2′	10
5	1.3 G	2′	2′	2′	10
6	1.3 G	2′	2′	2′	01
7	1.2(04) C	2′	2′	2′	01
8	1.2(04) C	2′	2′	2′	10

The values shown above are defined so that demonstrating feasibility of the computed solutions is easy. Since each munition package is eight cubic feet, 125 packages of any munition will equal the volume constraint for each building. The 1.1 NEW constraint for building 2 can be equaled with 10 packages of munition 1 or with 100 packages of munition 3.

Verification Tests

A series of tests were accomplished to verify the performance of the model. For each test, LPGEN was used to enter a munitions inventory in terms of stock number, number of lots, and number of packages for each lot. A useable volume of 100 percent was entered for each building. LPGEN then extracted the required information from each of the data bases and generated the objective function and constraints for each test. It then submitted the problem to LP/600 for calculation of the solution. The inputs and outputs for each test are listed in a chart for each test. The results can be examined and compaired against the information contained in the data bases as indicated above.

(1) Test-1 is a very basic one munition, one lot test of the system. The problem generated by LPGEN contained 11 decision variables and 16 constraints.

Munitions Inventory Input:

ID Number	Stock Number	Lot	Quantity	Class/Group	Package NEW
1	1	1	100	1.1 D	10

MSOS Storage Allocation:

ID Number	Building 1	2	3	4	5	Quantity Leftover
1	100	0	0	0	0	0
Leftover Vol	200	1000	1000	1000	1000	
Leftover NEW	0	All	All	All	All	

(2) Test-2 examines a simple case of one munition with three lots. The problem generated by LPGEN contained 23 decision variable and 18 constraints.

Munitions Inventory Input:

ID Number	Stock Number	Lot	Quantity	Class/Group	Package NEW
1	3	1	100	1.1 D	01
2	3	2	100	1.1 D	01
3	3	3	225	1.1 D	01

MSOS Storage Allocation:

ID Number	Building 1	2	3	4	5	Quantity Leftover
1	100	0	0	0	0	0
2	0	0	0	50	50	0
3	25	0	125	0	0	0
Leftover Vol	0	1000	0	600	0	
Leftover NEW	875	100	875	450	875	

(3) Test-3 examines two munitions, two lots, two classes, and two groups. This problem required 22 decision variables and 22 constraints.

Munitions Inventory Input:

ID Number	Stock Number	Lot	Quantity	Class/Group	Package NEW
1	1	1	150	1.1 D	10
2	6	1	150	1.3 G	01

MSOS Storage Allocation:

ID Number	Building 1	2	3	4	5	Quantity Leftover
1	100	0	0	0	50	0
2	0	0	125	25	0	0
Leftover Vol	200	1000	0	800	600	
Leftover NEW	0	All	5875	2975	500	

(4) Test-4 examines two types of munitions, 5 lots, two classes, and one group. The inventory volume is larger than the available storage volume. This test problem has 53 decision variables and 25 constraints.

Munitions Inventory Input:

ID Number	Stock Number	Lot	Quantity	Class/Group	Package NEW
1	1	1	150	1.1 D	10
2	1	2	150	1.1 D	10
3	4	1	150	1.3 D	10
4	4	2	150	1.3 D	10
5	4	3	150	1.3 D	10

MSOS Storage Allocation:

	Building					Quantity
ID Number	1	2	3	4	5	Leftover
1	0	0	0	0	100	50
2	0	0	0	0	0	150
3	0	25	125	0	0	0
4	0	25	0	125	0	0
5	100	10	0	0	0	40
Leftover Vol	200	520	0	0	200	
Leftover NEW	0	0	4750	1750	0	

(5) Test-5 examines four munitions, one lot each, two groups, and two classes per group. The problem for Test-5 consists of 54 decision variables and 34 constraints.

Munitions Inventory Input:

ID Number	Stock Number	Lot	Quantity	Class/Group	Package NEW
1	1	1	150	1.1 D	10
2	2	1	150	1.1 G	10
3	4	1	150	1.3 D	10
4	6	1	150	1.3 G	01

MSOS Storage Allocation:

ID Number	Building 1	2	3	4	5	Quantity Leftover
1	75	0	0	0	75	0
2	0	0	97.5	0	0	52.5
3	25	0	0	125	0	0
4	0	125	25	0	0	0
Leftover Vol	200	0	20	0	400	
Leftover NEW	0	475	0	1750	250	

(6) Test-6 examines all eight munitions, three groups, and three classes. This problem required 88 decision variables and 43 constraints.

Munitions Inventory Input:

ID Number	Stock Number	Lot	Quantity	Class/Group	Package NEW
1	1	1	20	1.1 D	10
2	2	1	20	1.1 G	10
3	3	1	20	1.1 D	01
4	4	1	20	1.1 D	10
5	5	1	20	1.1 G	10
6	6	1	20	1.3 G	01
7	7	1	20	1.2/04 C	01
8	8	1	20	1.2/04 C	10

MSOS Storage Allocation:

ID Number	Building 1	2	3	4	5	Quantity Leftover
1	20	0	0	0	0	0
2	0	0	0	0	20	0
3	20	0	0	0	0	0
4	20	0	0	0	0	0
5	0	0	20	0	0	0
6	0	0	20	0	0	0
7	0	0	0	20	0	0
8	0	0	0	20	0	0
Leftover Vol	520	1000	680	680	840	
Leftover NEW	580	100	5780	2280	800	

Test Analysis and Discussion

A manual search was conducted for each of the six test cases to determine if a better solution existed. In each case, the solution computed using MSOS proved to be an optimal solution. In several cases, identical results were obtained by switching some of the munitions from one building to another. This implies that multiple optimal solutions exist for some of the test cases. Overall, the MSOS functioned for these tests as intended. The following observations summarize the results of the six verification tests.

(1) Volume Constraints were met for each test case.

(2) Munition constraints worked properly. For each munition the number of packages stored plus the number leftover was equal to the number of packages input for inventory.

(3) NEW Subgroup Constraints functioned properly. In each case, the NEW constraint used for each building was the constraint for the most restrictive class of munition stored (Maximum NEW for storage was arbitrarily assigned to each class for these tests).

(4) Special Set (SSET) constraints worked as intended. Only one NEW constraint was used for each building and only one compatibility group was assigned to each building.

(5) Lot integrity was maintained where possible.

The user can take results such as shown for the test cases and determine whether they are actually feasible for the particular storage area in question. If alternate solutions are desired for examination, changes can be made to inventory inputs or to the constraining factors. One example is the situation where a highly pilferable item is designated by the MSOS as leftover. A less pilferable item can be deleted from the inventory to be considered by MSOS and the optimizing problem solved again.

The verification tests show that MSOS functions as intended. The next step is to attempt to validate the model using a real munitions storage area and a real munitions inventory. This validation is left for future research on the model.

V. Enhancements to the Model

The MSOS was designed with the intention that enhancements to the model would be added at a later time. The present base level model can be expanded to a theater wide or even an Air Force wide model. In addition, several specific ideas were discussed as possibilities for future enhancements either to this model or to new studies relating to optimization of munitions storage. These ideas are briefly discussed as suggestions for future research developments.

Sensitivity Analysis

An interactive program could be added to this present system that would incorporate the LP/600 sensitivity analysis capabilities with the MSOS. This provides the user with an increased ability to interpret the effects of any changes to the constraints in the model. For example, this would allow direct comparison of the effect a waiver to a NEW constraint would have on the solution. Development of an interactive program for this system would require someone with a computer backgroud. In particular, the language used in LP/600 must be learned in order for an interactive program to be designed for this system.

Other Objective Functions

Other aspects of munitions storage can be examined by introducing other objective functions to the model. Some areas of study that may result in the development of other objective functions are listed below.

(1) Minimization of explosives weight density in storage.

(2) Maximization of munitions dispersion for a war environment.

(3) Optimization for storage of additional inventory in partly filled storage areas.

(4) Optimization of a storage method designed to provide a systematic break-out and delivery of munitions from storage.

Prioritizing Munitions

LPGEN could be enhanced with the addition of a capability to prioritize munitions for storage. This could involve some type of identification of particularly sensitive or strategic items for storage. The priority scheme might relate to indoor versus outdoor storage or to items requiring very high security versus moderate security. A scheme could also be developed to combine all the components required for assembly of complete weapons systems in a joint storage situation.

Munitions Handling Efficiency

Given the Q-D and class/category/group constraint, a system could be developed to calculate layouts for munitions storage that will minimize handling distance and handling time. A cost function could be developed such that the cost for maintaining a given level of war readiness or mobility requirements could be minimized. An article by J. R. Berry, "Elements of Warehouse Layout" (Berry, 1968), describes a general formulation of an approach that might be useful in the development of this model.

A variation to this area might examine the actual floor space and overall volume utilization of various palletized munitions with the optimization problem considering the ease of placement and the efficient use of storage space. Variables such as aisle width, pallet size, and alternate types of forklifts could be examined for their various effects on the model. A paper by Joseph J. Moder and Herbert Thorton, "Quantitative Analysis of the Factors Affecting Floor Space Utilization of Palletized Cargo" (Moder and Thorton, 1965), presents a general formulation of this type of problem and could probably be adapted for the special conditions involved in storing munitions.

<u>Storage Facility Design</u>

Another possible study could investigate the cost-benefits of alternate low cost storage facilities. Pre-engineered or pre-fabricated structures could be examined for optimal storage with emphasis on class 1.1 explosives. It may be feasible to develop an optimization routine for low-cost inside storage for combat areas such that the safety constraints are not excessively compromised.

VI. Summary, Conclusions, and Recommendations

One of the major problems at many Air Force installations is the storage of large quantities of munitions. The storage process is made extremely complex by the many parameters and constraints that have been adopted to insure that the risk to people, facilities and equipment is limited to some acceptable level in the event of an actual mishap involving explosive materials. Most of the rules on storage have been developed to minimize the grave consequences of a munitions accident should it actually occur, with little consideration being given to the probability of an explosive accident. The constraints that go into the storage of munitions are driven primarily by the explosive quantity-distance (Q-D) criteria and the compatibility rules for storage. The many restrictions, coupled with a drive for inside-only storage present complex problems for munitions managers to solve.

The development of the Munitions Storage Optimization System (MSOS) is an initial attempt to quantify the complex munitions storage problem and to address it using the power of the computer. The MSOS should be considered primarily as a tool for munitions managers to use in gaining insight into the complexities of the munitions storage process and as a means to examine possible results or consequences of making changes to any of the parameters (such as useable volume, NEW constraints, or variations in inventory) that go into the problem.

The MSOS, while potentially useful in its present form, is basically a first effort to incorporate all the groups and classes of munitions into a single model that can be applied to any type of storage area. The model presently maximizes an objective function of storage area load factors. The storage area load factors are based on building load coefficients that are calculated by dividing the building density by the munition densitiy for the various combinations of NEW storage limits and munitions that might be assigned to each building. The possibilities for enhancing this model in various ways and for examining the problem from the viewpoint of alternative objective functions or other constraint combinations are without limit.

The completion of this research resulted in the accomplishment of three of the basic objectives outlined in the Introduction to this paper and partial completion of the fourth objective.

(1) A general approach was demonstrated for the application of mixed integer linear programming to the problem of optimizing munitions storage.

(2) A working model was invented that will enable munitions managers to examine possibe solutions for improved or more efficient storage of munitions.

(3) The program was developed in such a way that a user, with only a limited knowledge of the LP method, can simply follow a programmed prompting technique to input the required munitions and storage area data to the MSOS; the LPGEN program then creates the constraints and the objective function, then routes the problem to the LP/600 optimization package for generation of a solution.

(4) One measure of storage efficiency, the storage area load factor, was examined. Some of the other possibilities are mentioned in Chapter IV, Model Enhancements, and are left for future studies to explore.

It is recommended that future reasearch be sponsored for development of variations and enhancements to this present model. Using the MSOS as a starting point, there are any number of alternative analyses that can be undertaken.

Bibliography

Adams, Arlie E. "The United Nations System of Classification of Explosives--Where Are We Today?" Minutes of the Seventeenth Explosives Safety Seminar, Regency Inn, Denver Colorado, 14-16 September, 1976. 1567-1585. Washington: Department of Defense Explosives Safety Board, 1976. (AD A036 015).

AFR 127-100. Explosives Safety Standards. Washington: Department of the Air Force, 31 March 1978.

Berry, J. R. "Elements of Warehouse Layout," International Journal of Production Research, 7 (2): 105-123 (February,1968).

Cormier, Johnny E., Chief, Conventional Munitions Branch, HQ USAF/LEYW (correspondence to AFIT/EN). Washington D. C., 9 May 1979.

Farwell, David C. "The Munitions Problem: Storage Optimization for Class 7 Group G Munitions." Unpublished term paper. University of Colorado, 15 April 1970.

Fliakas, Perry J. "Explosive Safety Management--50 Years After Lake Demark," Minutes of the Seventeenth Explosives Safety Seminar, Regency Inn, Denver, Colorado, 14-16 September 1976. 1-6. Washington: Department of Defense Explosives Safety Board, 1976. (AD A036 015).

Fliakas, Perry J. "Keynote Address," Minutes of the Eighteenth Explosives Safety Seminar, El Tropicano Motor Hotel, San Antonio, Texas, 12-14 September 1978. 3-18. Alexandria, Virginia: Department of Defense Explosives Safety Board, 1978. (AD A066 568/9).

Lyman, Ona R. "The History of the Quantity Distance Tables for Explosives Safety," Minutes of the Eighteenth Explosives Safety Seminar, El Tropicano Motor Hotel, San Antonio, Texas, 12-14 September 1978. 129-144. Alexandria, Virginia: Department of Defense Explosives Safety Board, 1978. (AD A066 568/9).

Moder, Joseph J. and Herbert M. Thorton. "Quantitative Analysis of the Factors Affecting Floor Space Utilization of Palletized Storage," The Journal of Industrial Engineering, XVI: 8-18 (January-February, 1965).

Nicholson, T. A. J. Optimization in Industry, Volume I, Optimization Techniques. Chicago: Aldine-Atherton, Inc., 1971.

Shreyer, H. L. and L. E. Romesburg. Analytical Model for High Explosive Munitions Storage. AFWL Technical Report 70-20. Kirtland AFB, New Mexico: Air Force Weapons Laboratory, June 1970. (AD 873 169).

TO 11A-1-46. Firefighting Guidance Transportation and Storage Management Data and Ammunition Complete Round Chart. Washington: Department of the Air Force, 1 August 1977.

TO 11A-1-61-3. Storage and Outloading Instructions (Trailor-on-Flatcar Drawings and Carloading, Truckloading and Storage Drawings). Washington: Department of the Air Force, 1 May 1976.

TO 11A-1-61-4. Storage and Outloading Instructions (Storage in Igloo, Stradley, and Standard Type Magazines Drawings, and Miscellaneous Drawings). Washington: Department of the Air Force, 1 May 1976.

Wight, Richard L. "Corps of Engineers Repitoire of Earth-Covered Magazine Designs," Minutes of the Eighteenth Explosives Safety Seminar, El Tropicano Motor Hotel, San Antonio, Texas, 12-14 September 1978. 241-261. Alexandria, Virginia: Department of Defense Explosives Safety Board, 1978. (AD A066 568/9).

Appendix A

MUNITION STORAGE OPTIMIZING SYSTEM (MSOS)

USER'S MANUAL

This appendix is considered a stand-alone document and will be page numbered accordingly.

MUNITION STORAGE OPTIMIZING SYSTEM (MSOS)

USER'S MANUAL

Prepared
by
Louis M. Gusmus
3 December 1979

CONTENTS

Glossary

Building Density Factor - The building density factor is the building net explosive weight (NEW) capacity for a given class divided by the building volume.

Building Load Coefficient - The building load coeffieient is equal to the building density divided by the munition density for a specified subgroup.

Munition Density Factor - The munition density factor is the net explosive weight of the munition package divided by the munition package volume.

Net Explosive Weight (NEW) - The NEW is the total quantity, expressed in pounds of explosive materials or high explosives equivalency, that can be stored in a building or that is obtained in a munition package.

Special Set - The special set is a feature of the LP/600 package that allows the user to exclude constraints, or use only one of a set of constraints.

Special Set Variable - A special set variable is a bivalent variable having a value of 0 or 1. These variables are used to exclude all constraints except for one in a given set.

Storage Area Load Factor - The storage area load factor is the value assigned to a particular storage arrangement of a munitions inventory. Large values indicate a dense arrangement. This is the factor to be maximized by the mixed integer linear programming package.

Subgroup - A subgroup is a set of classes belonging to the same compatibility group whose associated building NEW for a particular building is less than or equal to that building's associated NEW for any class in the group.

I. Introduction

The Munition Storage Optimizing System (MSOS) provides the munitions manager with a means of determining an optimal allocation of a given munitions inventory to existing munition storage assets. MSOS allows the user to create and maintain the Standard Building Data Base, the Munition Storage Area Data Base, and the National Stock Number Data Base using utility programs. Once these data bases have been properly created, the user can use MSOS to determine the optimal allocation for storing an inventory of munitions in a given munition storage area, given a set of limiting conditions. The optimizing technique used is a mixed integer linear programming package produced by Honeywell Information System, Inc., and is called the LP/600 package.

The formulation of the MSOS optimization model required the creation of a complex system of computer programs to bring the information from each of the data bases into the model. For example, the set of constraints specifying the maximum NEW that can be stored in each building can vary, depending on the most restrictive class of munitions actually stored. Thus the optimizing model provides a capability to vary the maximum NEW that may be stored in each building during the optimizing process. A technique was developed to provide for the complex and changinining constraints using the mixed integer mode of the LP/600 package. The mathematical formulation of the model is discussed later.

The development of the optimization model requires the understanding of several terms critical to its mathematical formulation. They are:

(1) Subgroup. Munitions from different classes may be stored together providing they are from the same compatibility group. For a given munitions inventory, each identified group may contain 0 to 7 different classes of munitions. Every storage building has a NEW limit assigned for each of the seven classes and the limit actually used for a particular building is the NEW limit assigned for the most restrictive class of munition stored in the building. For example, an inventory of group D munitions belonging to classes 1.1, 1.2/04, and 1.3 require storage. The maximum NEW limits for a particular storage building are defined for this example to be: 1.1--100,000 pounds, 1.2/04--250.000 pounds, and 1.3--500,000 pounds. Storage of any combination of these munitions containing at least one 1.1 munition places the maximum NEW storage limit for the building at 100,000 pounds. If only 1.2/04 and 1.3 items are to be stored, the limit is 250,000 pounds. Similarly, if only 1.3 munitions are to be stored, the limit is 500,000 pounds. With this background, the definition of subgroup is offered. A subgroup is defined as a combination of munitions classes such that the first subgroup of a compatibility group contains all classes of munitions to be stored from that group; the next subgroup eliminates the most restrictive class from the group. Subsequent subgroups continue this process until all seven classes have been eliminated. Thus for each compatibility group, a maximum of seven subgroups may occur. Since there are twelve compatibility groups, a maximum of 84 subgroups can occur in the model.

(2) Useable Building Volume. LPGEN calculates the total inside storage volume for each building and requests the user to enter a subjective estimate of the percentage of this total volume considered

useable. This useable building volume is used both as the storage volume constraint for each building and for calculating building load coefficients.

(3) Building Load Coefficients. A building load coefficient is composed of the building density divided by the munition density. The building density is the subgroup NEW storage limit of the building divided by the the useable building volume. The munition density is the munition package NEW divided by the munition package volume. A particular munition may be identified with more than one subgroup, and therefore, be identified with the same building more than once. In each case, however, the maximum NEW limit for the building may be different, resulting in different building load coefficients being calculated for the same munition.

(4) Storage Area Load Factor. The storage area load factor is the factor that is maximized by the optimization process. It provides an overall measure of the building load coefficients for the entire munitions storage area. It is essentially a measure of how densely the munitions inventory is loaded into the munitions storage area. The density of the stored munitions is positively related to the storage area load factor. This means that the larger the value of the storage area load factor, the more efficient the use of the munitions storage area. The formulation of the objective function is such that if there are munitions that are not stored because the NEW limit is reached or the volume used up, they are assigned a negative coefficient. Thus negative values of the objective function are possible.

Now that these terms have been explained, the actual mathematical optimization model is presented.

Mathematical Formulation

For the specific case of examining optimization where the storage area load factor is maximized, the problem is formulated as a mixed integer linear programming model that is designed to:

Maximize the Storage Area Load Factor: OBJECTIVE =

$$\sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=1}^{l} c_{ijk}\ x_{ijk} + \sum_{j=1}^{n} \sum_{k=1}^{l} d_{jk}\ y_{jk} + \sum_{i=1}^{m} M_i\ z_i$$

Subject to:

Munition Constraints $\sum_{j=1}^{n} \sum_{k=1}^{l} x_{ijk} + z_i = I_i$ for $i=1,m$
(one for each munition/lot combination)

Building Volume Constraints $\sum_{i=1}^{m} \sum_{k=1}^{l} a_{ijk}\ x_{ijk} \le V_j$ for $j=1,n$
(one for each building)

NEW Constraints $\sum_{i=1}^{m} \sum_{k=1}^{l} b_{ijk}\ x_{ijk} - N_{jk}\ y_{jk} \le 0$ for $j=1,n$
(one for each subgroup/building combination)

Special Set Constraints $\sum_{k=1}^{l} y_{jk} = 1$ for $j=1,n$
(one for each building)

all $x_{ijk} \ge 0$

all $z_i \ge 0$

all $y_{jk} = 0$ or 1

where:

a_{ijk} is the volume of munition i in subgroup k to be stored in building j

b_{ijk} is the NEW of package of munition i in subgroup k to be stored in building j

c_{ijk} is the building j load coefficient of munition i in subgroup k

d_{jk} is a weight assigned to the subgroup k / building j combination (initialized to zero for this model)

M_i is a weight assigned to packages of munition i that cannot be stored in the storage area (initialized to -1 for this model)

x_{ijk} is the number of packages of munition i in subgroup k to be stored in building j

y_{jk} is the bivalent (0 or 1) special set variable assigned to the subgroup k / building j combination

z_i is the left over variable assigned to munition i

I_i is the number of packages in munition/lot combination i

N_{jk} is the maximum NEW that can be stored in building j for subgroup k

V_j is the maximum useable volume of building j

l is the number of different subgroups in the inventory

m is the number of munition/lot combinations

n is the number of buildings

The d_{jk}'s defined above could be set up as a priority scheme with priorities (weights) for assigning compatibility groups and classess within the compatibility groups to storage buildings or to the overall storage area. The M_i's could be used to establish a priority scheme to determine a rank order of munitions to be excluded from storage in the event that the entire inventory cannot be stored inside the storage area buildings. Neither of these priortiy schemes are addressed in this paper.

Standard Building Data Base

The Standard Building Data Base (SBDB) contains information that describes different types of munitions storage buildings. This implies that buildings having identical dimensional and structural characteristics will be identified as the same type. Once the buildings have been grouped into these different types, the user only needs to enter the requested information for each type of building into the

SBDB. The SBDB needs to be constructed only once for each munition storage area and can be modified whenever new types of buildings are built. See chapter two, Standard Building Data Base, for complete information on maintaining the SBDB. After constructing the SBDB the user needs to define specific information concerning the individual buildings.

Munition Storage Area Data Base

The Munition Storage Area Data Base contains the maximum allowable net explosive weight (NEW) for each class and maximum gross weight for every storage building in the munition storage area. The NEW for each class is calculated for each building based on the rules established in chapter four, Storage and Compatibility Standards, and chapter five, Principle and Application of Explosives Quantity-Distance Criteria and Related Standards, of Air Force Regulation 127-100. The user will have to determine these weights prior to creating the Munition Storage Area Data Base (MSADB). The data form in Appendix C of this user's manual should be very helpful in arranging the data that will be requested when creating a MSADB record. A file containing a completed data form of the type given in Appendix E for every storage building in the munition storage area should be established for future reference. Like the SBDB, the MSADB needs to be created only once and can be modified as buildings are constructed or destroyed. See chapter three, Munition Storage Area Data Base, for more details. Now that the munition storage area has been accounted for, detailed information about the munitions to be stored is needed.

National Stock Number Data Base

The munition National Stock Number Data Base (NSNDB) should contain all of the 1300 and 1400 series stock numbers. As a minimum, it must have all the national stock numbers of the munitions that will be in the user's inventory. The MSOS provides the user the capabilities to add, delete, and modify records in the NSNDB. See chapter four, National Stock Number Data Base, for futher details. Once the correct data is present in the SBDB, MSADB, and NSNDB, the user is ready to formulate the problem, using the model described above to determine how to store a given inventory.

Format Generator Program

The Format Generator Program (LPGEN) allows the user to enter the stock number, number of lots, and number of packages for each munition in the inventory. Then LPGEN selects appropriate information from the data bases described above and generates data in the correct format for input into the LP/600 optimizing package. Detailed information is found in chapter five, Data Generator Program.

The MSOS is designed to operate on the Honeywell 600 or 6000 series computer systems. Specific information concerning the Honeywell optimizing package (LP/600) can be found in Honeywell manuals:

(1) BP50 - Introduction to LP600 Linear Programming System
(2) BQ01 - LP600 System Input File Preparation Reference Manual
(3) BQ19 - LP600 System Agenda Control Language Reference Manual
(4) BQ20 - LP600 System Matrix Generator Language Reference Manual
(5) BQ21 - LP600 System Format Generator Language Reference Manual
(6) BQ22 - LP600 System Output Descriptions Reference Manual
(7) DA87 - LP600 System Linear Programming Demonstration Guide
(8) DA88 - LP600 System Mixed Integer Program Reference Manual

The user is assumed to have a basic knowledge of the Honeywell Time Sharing System (TSS), but just to clarify some shady areas the following information is given. The computer will display an "=" whenever it needs information from the user. The user should read each question carefully before responding. If the user should enter a "character" when a "digit" is requested, a read error will result. To correct this, just enter the appropriate value and processing will continue. The term "digit" means any number between 0 and 9. The term "character" refers to letters "A" to "Z" and numbers 0 to 9. Consult the Customer Service Section of the computer center for instruction in the use of TSS. Check with the computer software shop to determine if LP/600 is available on your system. The software shop shop will also have the appropriate file identification (userid/catalog file string) of the four programs the user will need.

II. Standard Building Data Base

The Standard Building Data Base (SBDB) is designed to contain the dimensional data about munition storage buildings. Most munition storage areas are composed of many buildings that are identical or can be grouped into types of identical buildings. Certain information about buildings in the munition storage area may be reduced to specific information about each TYPE of building in a storage area and stored in the SBDB. A TYPE of building is defined as a unique design of a building with specific dimensions. Therefore, if two buildings have the same unique design, but different dimensions, they will be of different types. The SBDB has to be created only once and updated as new types of buildings are constructed. It should be pointed out here, that the SBDB will probably be unique for every operational storage area. The reason for this is that certain storage buildings may be identical in one storage area, but at another base this type of building may not exist. Data contained in the SBDB is used by the Format Generator Program (LPGEN) in creating data to be submitted to the optimizing technique (LP/600). This data base is maintained by the Standard Building Data Base Utility Program (SBDBUP). Appendix A-3 gives a brief look at the available options of SBDBUP.

After logging on and receiving a "SYSTEM ?" message from the computer, the user can engage SBDBUP by entering the userid/catalog file string of SBDBUP and depressing the RETURN key. If the file SBDB exists and no other errors are encountered, the computer will display

WELCOME TO THE STANDARD BUILDING DATA BASE

ENTER THE ONE DIGIT TRANSACTION YOU DESIRE

At this point the user has six options.

(1) Press the RETURN key or enter a character other than 1 thru 5 - the computer will then list the valid options (see part 1.A. of Appendix D).

(2) Enter a "1" to add a record - the computer will then display

ENTER THE TWO DIGIT BUILDING TYPE OF RECORD TO BE ADDED

The user now enters the building TYPE of the new record; it must be a value between 1 and 99. Once it is created, the only way to change the value of TYPE is to delete the record and add a new record with the desired value for TYPE. If the TYPE entered by the user already exists in the data base a message is displayed noting this fact and the computer returns the user to the option level. If the TYPE entered by the user does not exist in the data base, the computer responds by requesting information about this type of building and displays the format needed. Values that must be entered are:

(A) ROOF - enter "RND" for round or igloo type buildings or "FLT" for regular non-igloo type buildings;

(B) LENGTH - enter the inner length (in feet) of the building;

(C) WIDTH - enter the inner width (in feet) of the building;

(D) SIDE WALL HEIGHT - the height of the straight part of the side walls (in feet), see Appendix B; and

(E) RADIUS - used only for igloo type buildings and must be the radius (in feet) of the curvature of the roof (see Appendix B).

While the remaining items for which information is requested are not currently being used, they may be used in later variations of the MSOS. Therefore, if the user knows the information, it should be entered now. Once the user has answered all the questions for the new record, the computer displays the new record and returns the user to the option level (see part 1.B. of Appendix D).

(3) Enter a "2" to change the value of one or more items of a particular record - the computer will display

ENTER THE TWO DIGIT BUILDING TYPE OF RECORD TO BE CHANGED

The user now enters the building TYPE value of the record. If the record does not exist, the computer will display a message stating this fact and returns the user to the option level. If the record is contained in SBDB, the computer requests the item number to be changed for this record. If the user does not know the appropriate item number, pressing the RETURN key will cause the computer to display all valid item numbers. Once the items for a particular record have been updated, the user must enter a "14" to terminate the transactions for this record. The computer will display the changed record and return the user to the option level (see part 1.C. of Appendix D).

(4) Enter a "3" to delete a record - the computer will display

ENTER THE TWO DIGIT BUILDING TYPE OF RECORD TO BE DELETED

The user now enters the building TYPE value of the record to be deleted. If the record is contained in SBDB, the computer displays a message stating that the record has been deleted. If the record does not exist in the data base, the computer displays a message stating this fact. In either case the computer returns the user to the option level (see part 1.D. of Appendix D).

(5) Enter a "4" to display a record - the computer will display

ENTER THE TWO DIGIT BUILDING TYPE OF RECORD TO BE DISPLAYED

The user now enters the building TYPE value of the desired record. If the record does not exist in SBDB, the computer displays a message stating this and returns the user to the option level. If the record is present, the computer displays the record and returns the user to the option level (see part 1.E. of Appendix D).

(6) Enter a "5" to terminate this session with SBDBUP - the computer will display a summary of transactions performed and the current number of records in SBDB (see part 1.F. of Appendix D).

III. Munition Storage Area Data Base

The Munition Storage Area Data Base (MSADB) contains information about the maximum allowable net explosive weight (NEW) for each class of munition that can be housed in each building belonging to the munition storage area. Each building record in the MSADB is identified as a type of building that is defined in the Standard Building Data Base (SBDB). The Format Generator Program (LPGEN) retrieves the data it needs in generating the information needed for the LP/600 optimizing package. The buildings have to be added to the MSADB only once. Thereafter, only new building records must be added or existing records changed or deleted. The Munition Storage Area Data Base Utility Program (MSADBUP) is used to maintain this data base. Appendix A-4 gives a brief look at the available options for MSADBUP.

After logging on and receiving a "SYSTEM ?" message from the computer, the user can engage MSADBUP by entering the userid/catalog file string of MSADBUP and hitting the RETURN key. If the file MSADB exists and no other errors are encountered, the computer will display

WELCOME TO THE MUNITION STORAGE AREA DATA BASE

ENTER THE ONE DIGIT TRANSACTION YOU DESIRE

At this point the user has six options.

(1) Press the RETURN key or enter a character other than 1 thru 5 - the computer will then list the valid options (see part 2.A. of Appendix D).

(2) Enter a "1" to add a record - the computer will display

ENTER THE 6 CHARACTER BUILDING NUMBER OF RECORD TO BE ADDED

The user now enters the building number to be added; any alphanumeric string of six or less characters is valid. Once this record is added, the building number cannot be changed, only deleted and a new record added, having the correct building number. If the building number already exists the computer displays a message noting this fact and returns the user to the option level. If the building number does not exist in the data base, the computer responds by requesting information about the building and displays the format needed. Values that must be entered are:

(A) TYPE - a value between 1 and 99 that corresponds with the building type located in SBDB, unmatched TYPE will cause an error in LPGEN;

(B) 1.1 NEW - the maximum NEW for class 1.1 munitions that can be stored in this building;

(C) 1.2/04 NEW - the maximum NEW for class 1.2, category 04 munitions that can be stored in this building;

(D) 1.2/08 NEW - the maximum NEW for class 1.2, category 08 munitions that can be stored in this building;

(E) 1.2/12 NEW - the maximum NEW for class 1.2, category 12 munitions that can be stored in this building;

(F) 1.2/18 NEW - the maximum NEW for class 1.2, category 18 munitions that can be stored in this building;

(G) 1.3 NEW - the maximum NEW for class 1.3 munitions that can be stored in this building; and

(H) 1.4 NEW - the maximum NEW for class 1.4 munitions that can be stored in this building.

Once the user has entered all data for this new record, the computer will display the new record and return the user to the option level (see part 2.B. of Appendix D).

(3) Enter a "2" to change the value of one or more items of a particular building's record - the computer will display

ENTER THE 6 CHARACTER BUILDING NUMBER OF RECORD TO BE CHANGED

The user enters the building number of record to be changed. If the record does not exist the computer displays a message stating this fact and returns the user to the option level. If the record exists, the computer will request the item number to be changed for this record. If the user does not know the appropriate item number, pressing the RETURN key will cause the computer to display all valid item numbers. Once the item values for a particular record have been updated, the user must enter a "11" to terminate the transaction for this record. The computer will display the changed record and return the user to the option level (see part 2.C. of Appendix D).

(4) Enter a "3" to delete a record - the computer will display

ENTER THE 6 CHARACTER BUILDING NUMBER OF RECORD TO BE DELETED

The user now enters the building number of the record to be deleted. If the record exists in MSADB, the computer displays a message stating that the record has been deleted. If the record is not contained in the data base, the computer displays a message stating this fact. In either case, the computer returns the user to the option level (see part 2.D. of Appendix D).

(5) Enter a "4" to display a record - the computer displays a message

ENTER THE 6 CHARACTER BUILDING NUMBER OF RECORD TO BE DISPLAYED

The user now enters the building number of the desired record. If the record does not exist in MSADB, the computer displays a message stating this fact and returns the user to option level. If the record is present, the computer displays the record and returns the user to the option level (see part 2.E. of Appendix D).

(6) Enter a "5" to terminate this session with MSADB - the computer will display a summary of transactions performed and the current number of records in MSADB (see part 2.F. of Appendix D).

IV. National Stock Number Data Base

The National Stock Number Data Base (NSNDB) contains pertinent information about munitions and their packaging. Each stock number represents a particular munition packaged in a unique way. The same munition may be identified by several stock numbers, the only difference being in the way the munition is packaged. The Format Generator Program (LPGEN) uses this file in generating information about the munition inventory for input into the Honyewell LP/600 optimizing package. The munition records need to be entered only once. Thereafter, only new munition records need to be added to NSNDB or existing records updated or deleted. The National Stock Number Utility Program (NSNDBUP) is used to maintain this data base. Appendix A-5 gives a brief look at the available options for NSNDBUP.

After logging on and receiving the "SYSTEM ?" message from the computer the user can engage NSNDUP by entering the userid/catalog file string of NSNDBUP and pressing the RETURN key. If the file NSNDB exists and no other errors are encountered, the computer will display

WELCOME TO THE MUNITONS STORAGE DATA BASE

ENTER THE ONE DIGIT TRANSACTION YOU DESIRE

At this point the user has six options.

(1) Press the RETURN key or enter a character other than 1 thru 5 - the computer will then list the valid options (see part 3.A. of appendix D).

(2) Enter a "1" to add a record - the computer will display

ENTER THE 18 CHARACTER STOCK NUMBER OF RECORD TO BE ADDED

The user now enters the stock number, consisting of up to 18 characters, to be added. Once this record is added, the stock number cannot be changed, only deleted. If the stock number entered already exists, the computer displays a message to this effect and returns the user to the option level. If the stock number is not in NSNDB, the computer responds by requesting information about the munition and displays the input format needed. The mandatory values that must be entered are:

(A) PACKAGE HEIGHT - the height (in feet) of the munition package;

(B) PACKAGE WIDTH - the width (in feet) of the munition package;

(C) PACKAGE LENGTH - the length (in feet) of the munition package;

(D) UNITS PER PACKAGE - the number of individual munitions in this package;

(E) NEW EXPLOSIVE WEIGHT - the NEW in pounds of the munition package;

(F) COMPATIBILITY GROUP - the munition compatibility group;

(G) CLASS/DIVISION - the munition class and division information;

(H) CATEGORY - used only for class 1.2 munitions and is the restrictive distance code, valid values are 04, 08, 12, 18 (these values represent the distance in hundreds of feet that this munition can be stored from a hazard or obstacle).

The other information should be entered if it is known, but is not currently used by LPGEN. Once the user has entered all requested data for this new record, the computer will display the new record and return the user to the option level (see part 3.B. of Appendix D).

(3) Enter a "2" to change the value of one or more items of a particular stock number record - the computer will display

ENTER THE 18 CHARACTER STOCK NUMBER OF RECORD TO BE CHANGED

The user enters the stock number of record. If the record does not exist, the computer displays a message stating this fact and returns the user to the option level. If the record exists, the computer will request the item number to be changed for this record. If the user does not know the appropriate item number, pressing the RETURN key will cause the computer to display all valid item numbers. Once the item values for a particular record have been updated, the user must enter a "12" to terminate the transactions for this record. The computer will display the changed record and return the user to the option level (see part 3.C. of Appendix D).

(4) Enter a "3" to delete a record - the computer will display

ENTER THE 18 CHARACTER STOCK NUMBER OF RECORD TO BE DELETED

The user now enters the stock number of the record to be deleted. If the record exists in NSNDB, the computer displays a message stating that the record has been deleted. If the record is not contained in the data base, the computer displays a message stating this fact. In either case, the computer returns the user to the option level (see part 3.D. of Appendix D).

(5) Enter a "4" to display a record - the computer responds with

ENTER THE 18 CHARACTER STOCK NUMBER OF RECORD TO BE DISPLAYED

The user now enters the stock number of the desired record. If the record does not exist in NSNDB, the computer displays a message to this effect and returns the user to the option level. If the record is present, the computer displays the record before returning the user to the option level (see part 3.E. of Appendix D).

(6) Enter a "5" to terminate this session with NSNDB - the computer will display a summary of transactions performed and the current number of records in NSNDB (see part 3.F. of Appendix D).

V. Data Generator Program

The Data Generator Program (LPGEN) reads information from NSNDB, MSADB, and SBDB, then receives munition inventory information and usable building volume from the user. It manipulates and transforms this information into properly formatted data and stores the formatted data on file LPINFO. LPGEN also produces three cross reference lists:

(1) Munition cross reference displaying internal identification, corresponding munition stock number, lot number, and number of packages per lot;

(2) Group cross reference displaying internal identification and corresponding compatability group; and

(3) Class cross reference displaying internal identification and the corresponding class/division/catagory.

The cross references are stored on file CRSREF. LPGEN then submits LPINFO to the Honeywell LP/600 optimizing package using the job control language it creates. The final output of LP/600 will be stored on file LPOUT and will also be listed on the main printer. See Appendix E for instructions on interpreting the output.

In order for LPGEN to function properly the following files must be accessible on the user's system:

(1) NSNDB - the national stock number data base;

(2) SBDB - the standard building data base;

(3) MSADB - the munition storage area date base;

(4) LPINFO - the formatted input to LP/600;

(5) CRSREF - the munition, group, and class/division/category cross

reference lists; and

(6) LPOUT - the LP/600 output of the optimizing package.

After logging on and receiving a "SYSTEM ?" message from the computer, the user can engage LPGEN by entering the userid/catalog file string of LPGEN and hitting the RETURN key. If all the files described above do exist and contain the proper information the computer will display

```
WELCOME TO THE INVENTORY PROGRAM

ENTER THE NATIONAL STOCK NUMBER OF MUNITION TO BE
     ENTERED IN INVENTORY AND HIT RETURN KEY
     IF FINISHED ENTER '*' AND HIT RETURN
```

The user should now enter the first munition stock number using a maximum of 18 characters and press the RETURN key. The computer will display one of the following messages:

(1) If the stock number is not contained in NSNDB -

```
**WARNING**  STOCK NUMBER XXXXXXXXXXXXXXXXXX
   DOES NOT RESIDE IN NSNDB -- MUNITION WILL NOT BE ACCEPTED

ENTER NEXT STOCK NR OR '*' IF FINISHED
```

This message simply means that this particluar stock number is bypassed. The user should enter the next stock number to continue or enter an "*" to terminate LPGEN and update NSNDB; or

(2) If the stock number is found, the computer will display

```
ENTER THE NUMBER OF LOTS FOR MUNITION XXXXXXXXXXXXXXXXXX
```

The user will now enter the number of lots (1 to 99 are valid values) and the computer will display

```
ENTER THE NUMBER OF PACKAGES FOR LOT n
```

The user will now enter the number of packages for lot n (1 to 9999 are valid values). This message will be repeated for each lot, then the computer will display

ENTER NEXT STOCK NUMBER OR '*' IF FINISHED

The user will continue entering stock numbers, following the above procedure until the inventory has been entered. When the user enters an "*", the computer will display

DO YOU WISH TO STOP THE PROGRAM NOW ? (Y OR N)

The user will respond with "Y" if the answer is yes or "N" if the answer is no. If the program detects invalid data in the compatability group, class, or category, it will display the following message

```
MUNITION XXXXXXXXXXXXXXXXXX HAS INVALID DATA IN GROUP, CLASS OR CAT
VALUES ARE:  X,XXX,XX

                    CAUSING PROGRAM TO TERMINATE....
```

The user should update the NSNDB before running LPGEN again. If no errors are detected by the computer during the munition inventory input, the computer will display

STORAGE FACILITY DATA IS NOW BEING GENERATED

PLEASE READ THE FOLLOWING QUESTION CAREFULLY...

```
   DO YOU WISH TO ENTER A DIFFERENT PERCENTAGE OF USABLE
VOLUME FOR EACH BUILDING ? (Y OR N)
```

If the user enters "N" for no, the computer displays

```
ENTER THE PERCENTAGE OF BUILDING VOLUME THAT
     IS CONSIDERED USABLE, E.G., 75.8
```

The user will enter the appropriate percentage. This percentage will be used for all buildings in the storage area. If the user enters "Y" for yes, the computer will display

ENTER THE PERCENTAGE OF USABLE VOLUME FOR BUILDING XXXXXX, E.G., 75.8

The user will enter the appropriate percentage (0 to 100 are valid values) for each building in the storage area. LPGEN matches each building in MSADB with its corresponding type of building in SBDB. If there is an unmatched buildings, the computer will display

```
BUILDING XXXXXX IS IDENTIFIED AS TYPE XX
     BUT THIS TYPE OF BUILDING IS NOT DEFINED IN THE SBDB

 ***** FATAL ERROR -- PROGRAM NOW TERMINATING
```

Before running LPGEN again the user must add a record for type XX to SBDB, or update building XXXXXX record in NSADB by changing TYPE to a valid one. If all the buildings in MSADB have valid TYPE identifications, the computer will display the following messages

```
GENERATING OBJECTIVE FUNCTION AND CONSTRAINTS NOW
     PLEASE WAIT....

THE OBJECTIVE FUNCTION IS NOW BEING ORGANIZED

NOW GENERATING MUNITION CONSTRAINTS....

STARTING THE BUILDING VOLUME CONSTRAINTS NOW....

GENERATING GROUP AND SUBGROUP CONSTRAINTS NOW....

CONCLUDING BY GENERATING RHS...
```

The computer will then print the cross reference lists and display

```
THIS PROBLEM CONTAINS XXXX DECISION VARIABLES IN THE
     OBJECTIVE FUNCTION AND XXXX CONSTRAINTS
```

The program will stop if the number of constraints is greater than 4095 or the number of decision variables is greater than 262,000. The cross reference lists and the above message will also be printed on the main printer. If no error conditions are detected, the computer will display

```
SPAWNING THE LP JOB NOW...

             BYE

SNUMB # XXXXT
```

The user should keep a record of this snumb # until he has received the output for this job. Examples of program execution are found in part 4 of Appendix D. Appendix E shows the user how to interpret the output.

Appendix A

Utility Program Options

This appendix gives the user an easy to follow overview of the three utility programs used in MSOS.

1. STANDARD BUILDING DATA BASE UTILITY PROGRAM 23 NOV 79

ENTER A ONE DIGIT VALUE AND HIT CARRIAGE RETURN

1 - ADD
2 - CHANGE
3 - DELETE
4 - DISPLAY
5 - TERMINATE

IF "1" (ADD RECORD)
FILL IN APPROPRIATE DATA ITEMS AS THEY ARE PRESENTED
AND HIT THE CARRIAGE RETURN

IF "2" (CHANGE RECORD)
ENTER TWO DIGIT BUILDING TYPE OF RECORD TO BE CHANGED
AND HIT THE CARRIAGE RETURN

THEN ENTER TWO DIGIT ITEM NUMBER TO BE CHANGED
AND HIT THE CARRIAGE RETURN

ITEM NR	ITEM	INPUT FORMAT
02	NAME (6A)	AAAAAA
03	ROOF (ROUND=RND, FLAT=FLT)	AAA
04	LENGTH (IN FEET)	999.99
05	WIDTH (IN FEET)	999.99
06	RADIUS (IN FEET)	999.99
07	SIDE WALL HEIGHT (IN FEET)	99.99
08	WALL THICKNESS (IN FEET)	9.99
09	ROOF THICKNESS (IN FEET)	9.99
10	ENTRANCE HEIGHT (IN FEET)	99.99
11	ENTRANCE WIDTH (IN FEET)	99.99
12	DOOR THICKNESS (IN FEET)	9.99
13	MAXIMUM ALLOWABLE WEIGHT (IN TONS)	9999.99
14	FINISHED CURRENT TRANSACTION	

IF "3" (DELETE RECORD)
ENTER TWO DIGIT "99" BUILDING TYPE OF RECORD TO BE DELETED
AND HIT THE CARRIAGE RETURN

IF "4" (DISPLAY RECORD)
ENTER TWO DIGIT "99" BUILDING TYPE OF RECORD TO BE DISPLAYED
AND HIT THE CARRIAGE RETURN

IF "5" TERMINATE THE EXECUTION OF THIS PROGRAM

2. MUNITION STORAGE AREA DATA BASE UTILITY PROGRAM   23 NOV 79

ENTER A ONE DIGIT VALUE AND HIT CARRIAGE RETURN
- 1 - ADD
- 2 - CHANGE
- 3 - DELETE
- 4 - DISPLAY
- 5 - TERMINATE

IF "1" (ADD RECORD)
FILL IN APPROPRIATE DATA ITEMS AS THEY ARE PRESENTED AND HIT THE CARRIAGE RETURN

IF "2" (CHANGE RECORD)
ENTER 6 CHARACTER BUILDING NUMBER OF RECORD TO BE CHANGED AND HIT THE CARRIAGE RETURN

THEN ENTER TWO DIGIT ITEM NUMBER TO BE CHANGED AND HIT THE CARRIAGE RETURN

ITEM NR	ITEM	INPUT FORMAT
02	NAME (6A)	AAAAAA
03	TYPE (2N)	99
04	CLASS/DIV 1.1 NEW (IN POUNDS)	9999999
05	CLASS/DIV/CAT 1.2 04 NEW (IN POUNDS)	9999999
06	CLASS/DIV/CAT 1.2 08 NEW (IN POUNDS)	9999999
07	CLASS/DIV/CAT 1.2 12 NEW (IN POUNDS)	9999999
08	CLASS/DIV/CAT 1.2 18 NEW (IN POUNDS)	9999999
09	CLASS/DIV 1.3 NEW (IN POUNDS)	9999999
10	CLASS/DIV 1.4 NEW (IN POUNDS)	9999999

11 - FINISHED CURRENT TRANSACTION

IF "3" (DELETE RECORD)

ENTER 6 CHARACTER BUILDING NUMBER OF RECORD TO BE DELETED AND HIT THE CARRIAGE RETURN

IF "4" (DISPLAY RECORD)
ENTER 6 CHARACTER BUILDING NUMBER OF RECORD TO BE DISPLAYED AND HIT THE CARRIAGE RETURN

IF "5" TERMINATE THE EXECUTION OF THIS PROGRAM

3. NATIONAL STOCK NUMBER DATA BASE UTILITY PROGRAM 23 NOV 79

ENTER A ONE DIGIT VALUE AND HIT CARRIAGE RETURN
1 - ADD
2 - CHANGE
3 - DELETE
4 - DISPLAY
5 - TERMINATE

IF "1" (ADD RECORD)
FILL IN APPROPRIATE DATA ITEMS AS THEY ARE PRESENTED AND HIT THE CARRIAGE RETURN

IF "2" (CHANGE RECORD)
ENTER 18 DIGIT NATIONAL STOCK NUMBER OF RECORD TO BE CHANGED (REF AFTO 11A-1-46) AND HIT THE CARRIAGE RETURN

THEN ENTER TWO DIGIT ITEM NUMBER TO BE CHANGED AND HIT THE CARRIAGE RETURN

ITEM NR	ITEM	INPUT FORMAT
01	NATIONAL STOCK NUMBER	9999-99-999-9999AA
02	STACKING HEIGHT (NR OF PACKAGES)	9999
03	PACKAGE HEIGHT (IN FEET)	999.9
04	PACKAGE WIDTH (IN FEET)	999.9
05	PACKAGE LENGTH (IN FEET)	999.9
06	UNITS PER PACKAGE	9999
07	GROSS WEIGHT (IN POUNDS)	99999.9999
08	NET EXPLOSIVE WEIGHT (IN POUNDS)	99999.9999
09	COMPATIBILITY GROUP	A
10	CLASS/DIVISION	9.9
11	CATEGORY FOR 1.2	99
12	FINISHED CURRENT TRANSACTION	

IF "3" (DELETE RECORD)
ENTER 18 DIGIT STOCK NUMBER OF RECORD TO BE DELETED AND HIT THE CARRIAGE RETURN

IF "4" (DISPLAY RECORD)
ENTER 4 DIGIT STOCK NUMBER OF RECORD TO BE DISPLAYED AND HIT THE CARRIAGE RETURN

IF "5" TERMINATE THE EXECUTION OF THIS PROGRAM

Appendix B

Building Volume Calculations

The Format Generator Program (LPGEN) considers two basic types of storage buildings in calculating the building volume:

(A) Above ground magazines - volume is equal to the (length) x (width) x (wall height) x (percent usable volume); and,

(B) Igloos - subdivided into three designs:

(1) Diameter of roof is equal to the width of building - volume is equal to (the sector area x length + the side wall height x length x width) x the percent usable volume (the sector area is the area under the curved roof and above the height of the side walls);

(2) Diameter of roof is greater than the width of the building - volume is equal to (the portion of sector area in building x the length + the side wall height x length x width) x the percent usable volume;

(3) Roof is eliptical or parabolic. The user will have to approximate the volume of any building having this design by redefining the building dimensions so that thy meet the criteria for one of the other two igloo designs.

Appendix C

Net Explosive Weight Determination

One of the key features of the MSOS centers on the calculations of the density factors for each building within the munitions storage area. This density factor, when divided by the munitions density factor forms an overall storage load factor that is the heart of the optimization routine. The building density factor for a given class is calculated by dividing the building NEW capacity for that class by the useable storage volume of the building. The building NEW capacity for each class must be determined by the user for every building to be entered in the MSADB. These NEW capacities may already exist or might have to be calculated using information contained in chapter 5, Principles and Applications of Explosives Quantity-Distance Criteria and Related Standards, of AFR 127-100, Explosives and Safety Standards.

To make the actual determination of the maximum NEW of each class for each building, the user must have the following information:

(1) Type of magazine (igloo of aboveground).

(2) Barricaded of unbarricaded (front, sides,rear).

(3) Distance to nearest inhabited building (IBD).

(4) Distance to nearest public traffic route (PTR).

(5) Distance separating any two operating buildings within an operating line (Intraline Distance).

(6) Distance between any two explosive storage locations (Intermagazine Distance).

Distances are measured from the nearest outside point (or wall) of the storage location to the nearest outside point (or wall) of the other location. With this information, the quantity-distance (Q-D) charts found in chapter 5 of AFR 127-100 may then be used to determine the maximum NEW weight that may be stored in each storage facility/class combination. Classes 1.1, 1.2 (12), and 1.2 (18) reach maximum limits of 500,000 pounds NEW; classes 1.2 (04), 1.2 (08), and 1.4 have no NEW limit at distances greater than 400, 800, and 100 feet respectively. In cases where the distances are such that the NEW is unlimited, an actual weight limit must still be entered in the MSADB. If the maximum gross weight that may be stored in the building is known, this figure could be used or, if that is not available, 9999999 pounds must be entered.

A form is provided to help organize and calculate the required data. Once all the data is entered, the most restrictive NEW for each class of munition can be read directly from the form. These are the values that must be entered in the MSADB. If any waivers have been approved, or if the potential effect of a waiver is desired, it is easy to determine the next most restrictive NEW from the form. It is suggested that a copy of this form be completed and kept on file for each facility within the munitions storage area. The following instructions will help the user complete the building NEW capacity form.

Section I contains the building description information (items 1 through 8).

1 BUILDING NUMBER - enter the building number of the subject storage building.

2 DESIGN TYPE - enter the type of building i.e., aboveground, igloo, etc.

(3) TYPE CONSTRUCTION - enter the type of construction i.e., concrete, steel, etc.

(4) EXTENT OF BARRICADE - enter barricade type and placement, e.g. earth, top, sides, rear.

(5) GR STORAGE WT - enter the gross building storage weight, in tons, if known.

(6) INSIDE DIMENSIONS

(A) LENGTH - enter the building length, in feet, measured from inside.

(B) WIDTH - enter the building width, in feet, measured from inside.

(C) SIDE WALL HEIGHT - enter the height of the walls, in feet, measured from inside. This measurement should be the straight portion of the side walls for igloos or any building with curved roofs, or the distance to the ceiling for regular buildings.

(D) ROOF RADIUS - enter the radius of the roof (valid only for igloos or other buildings having a curved roof).

(7) STANDARD BUILDING TYPE - enter the two digit code for the type, if known (this entry will be the value used in the MSADB and SBDB).

(8) OTHER - enter any other information deemed to be useful.

Section II of this form is used to describe the physical location of this building with respect to nearby obstacles in terms of quantity-distance information for each class of munition that may be stored in this building. Nine pieces of information are need for each obstacle. Six obstacles have been identified, with space available for additional ones, if they are present. The obstacle distances identified are:

(1) MAGAZINE DISTANCE - this is the distance to the nearest munition storage building;

(2) INTRALINE DISTANCE - this is the distance to the closest building building in the storage area;

(3) PTR DISTANCE - this is the distance to the closest public transportation route;

(4) IHB DISTANCE - this is the distance to the closest inhabited building;

(5) RUNWAY DISTANCE - this is the closest distance to the runway (if appropriate); and

(6) REC AREA DISTANCE - this is the distance to the closest recreation area (if appropriate).

Very detailed information about these obstacles is contained in chapter 5 of AFR 127-100. The following information is needed for all identified obstacles.

9 OBSTACLE IDENTITY - enter the building number, highway number or street name, runway number, etc.

10 DISTANCE - enter the distance, in feet, from this building to the specified obstacle.

11 CLASS 1.1 - *
12 CLASS 1.2 04 - * * enter the NEW, found in the appropriate
13 CLASS 1.2 08 - * * tables in chapter 5 of AFR 127-100,
14 CLASS 1.2 12 - * * for each class of munitions (if capacity
15 CLASS 1.2 18 - * * of building is the limit, enter gross weight
16 CLASS 1.3 - * * if known, otherwise enter 9999999)
17 CLASS 1.4 - *

Section III contains the information needed by the MSADB for each building. The smallest value from each class column in section II (items (11) through (17)) will be entered into the corresponding class columns in this section (items (18) through (24)). If a waiver has been established for a particular obstacle, e.g. IHB DISTANCE, then that row will be ignored in compiling values for each class column.

Once this section has compiled been completed, the user has the necessary data for this building needed by the MSADB. If the surrounding conditions of a building change, this form should be updated and if any item of section III changes, the MSADB must be updated with the appropriate information.

BUILDING NET EXPLOSIVES WEIGHT CALCULATION FORM

SECTION I. BUILDING DESCRIPTION

1 BUILDING NUMBER:	6 BLDG INSIDE DIMENSIONS (IN FT)	7 STANDARD BUILDING TYPE:
2 DESIGN TYPE:	LENGTH:	8 OTHER INFORMATION
3 TYPE CONSTRUCTION:	WIDTH:	
4 EXTENT OF BARRICADE:	SIDE WALL HEIGHT:	
5 GR STORAGE WT (IN TONS):	ROOF RADIUS (FOR IGLOOS):	

SECTION II. QUANTITY - DISTANCE COMPUTATIONAL DATA

NEAREST OBSTACLE THAT REQUIRES:	9 OBSTACLE IDENTITY	10 DISTANCE (IN FEET)	11 CLASS 1.1	CLASS 1.2 12 CAT 04	 13 CAT 08	 14 CAT 12	 15 CAT 18	16 CLASS	17 CLASS
MAGAZINE DISTANCE									
INTRALINE DISTANCE									
PTR DISTANCE									
IHB DISTANCE									
RUNWAY DISTANCE									
REC AREA DISTANCE									
OTHERS									

SECTION III. NET EXPLOSIVE WEIGHT STORAGE AUTHORIZATION
(smallest value of each class column in section II, items 10 - 17)

18 CLASS 1.1	CLASS 1.2 19 CAT 04	 20 CAT 08	 21 CAT 12	 22 CAT 18	23 CLASS 1.3	24 CLASS 1.4

PREPARED BY:

DATE

REMARKS

Appendix D

Program Execution Examples

This appendix takes the user, step by step, through all paths of each program in the MSOS.

1. Standard Building Data Base Utility Program examples:

A. <u>Starting the SBDBUP program</u>:

```
SYSTEM ?79C06/OBJECT/SBDBUP - - enter the appropriate file string
     WELCOME TO THE STANDARD BUILDING DATA BASE

     ENTER THE ONE DIGIT TRANSACTION DESIRED:

=    - - just press RETURN key to display options
          OPTIONS:
                         1 - ADD RECORD
                         2 - CHANGE RECORD
                         3 - DELETE RECORD
                         4 - DISPLAY RECORD
                         5 - TERMINATE JOB

     ENTER THE ONE DIGIT TRANSACTION DESIRED:
```

B. Exercising the ADD option:

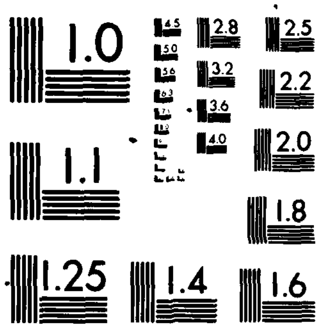
```
     ENTER THE ONE DIGIT TRANSACTION DESIRED:
=1
     ENTER THE TWO DIGIT BUILDING TYPE OF RECORD TO BE ADDED
=23
     RECORD 23 DOES NOT EXIST

     ENTER THE ONE DIGIT TRANSACTION DESIRED:
=1
     ENTER THE TWO DIGIT BUILDING TYPE OF RECORD TO BE ADDED
=55
     ENTER THE INFORMATION FOR THE BUILDING NAME (XXXXXX)
=IGLOO4
     ENTER THE INFORMATION FOR THE ROOF 'RND' - IGLOO OR 'FLT' - REGULAR
=RND
     ENTER THE INFORMATION FOR THE BUILDING LENGTH (999.99) IN FT
=120
     ENTER THE INFORMATION FOR THE BUILDING WIDTH (999.99) IN FT
=30
     ENTER THE INFORMATION FOR THE RADIUS OF ROOF (999.99) IN FT
=26
     ENTER THE INFORMATION FOR THE SIDE WALL HEIGHT (99.99) IN FT
=5
     ENTER THE INFORMATION FOR THE WALL THICKNESS (9.99) IN FT
=
     ENTER THE INFORMATION FOR THE ROOF THICKNESS (9.99) IN FT
=
     ENTER THE INFORMATION FOR THE DOOR HEIGHT (99.99) IN FT
=
     ENTER THE INFORMATION FOR THE DOOR WIDTH (99.99) IN FT
=
     ENTER THE INFORMATION FOR THE DOOR THICKNESS (9.99) IN FT
=
     ENTER THE INFORMATION FOR THE MAX WEIGHT (9999.99) IN TONS
=9999.99
```

```
BUILDING ID - 55
     NAME - IGLOO4
     ROOF - RND
     LENGTH - 120.00 FT
     WIDTH -  30.00 FT
     RADIUS -  26.00 FT
     SIDE WALL HEIGHT -  5.00 FT
     WALL THICKNESS - 0.   FT
     ROOF THICKNESS - 0.   FT
     ENTRANCE HEIGHT -  0.   FT
     ENTRANCE WIDTH -  0.   FT
     DOOR THICKNESS - 0.   FT
     MAXIMUM WEIGHT - 9999.99 TONS

ENTER THE ONE DIGIT TRANSACTION DESIRED:
```

AD-A083 708 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/G 15/5
OPTIMIZATION OF MUNITIONS STORAGE.(U)
DEC 79 B A BOGGS, L M GUSMUS
UNCLASSIFIED AFIT/GSM/SM/79D-15 NL

2 of 4

AD
A083708



MICROCOPY RESOLUTION TEST CHART

C. <u>Exercising the CHANGE option</u>:

```
ENTER THE ONE DIGIT TRANSACTION DESIRED:

=2
    ENTER THE TWO DIGIT BUILDING TYPE OF RECORD TO BE CHANGED
=5
    RECORD  5 DOES NOT EXIST

    ENTER THE ONE DIGIT TRANSACTION DESIRED:

=2
    ENTER THE TWO DIGIT BUILDING TYPE OF RECORD TO BE CHANGED
=55
    ENTER THE 2 DIGIT ITEM NUMBER OF THE ITEM TO BE
         CHANGED FOR BUILDING TYPE 55

=  - - just press RETURN key to display options
    OPTIONS:
                    02 - NAME (6A)
                    03 - ROOF (3A) 'RND' - ROUND OR 'FLT' - FLAT
                    04 - LENGTH (999.99) IN FT
                    05 - WIDTH (999.99) IN FT
                    06 - RADIUS (999.99) IN FT
                    07 - SIDE WALL HEIGHT (99.99) IN FT
                    08 - WALL THICKNESS (9.99) IN FT
                    09 - ROOF THICKNESS (9.99) IN FT
                    10 - ENTRANCE HEIGHT (99.99) IN FT
                    11 - ENTRANCE WIDTH (99.99) IN FT
                    12 - DOOR THICKNESS (9.99) IN FT
                    13 - MAX WEIGHT (9999.99) IN TONS

                    14 - FINISHED THIS TRANSACTION

    ENTER THE 2 DIGIT ITEM NUMBER OF THE ITEM TO BE
         CHANGED FOR BUILDING TYPE 55

=2
    ENTER THE INFORMATION FOR THE BUILDING NAME (XXXXXX)

=IGLOO5
    ENTER THE 2 DIGIT ITEM NUMBER OF THE ITEM TO BE
         CHANGED FOR BUILDING TYPE 55

=13
    ENTER THE INFORMATION FOR THE MAX WEIGHT (9999.99) IN TONS

=1000
    ENTER THE 2 DIGIT ITEM NUMBER OF THE ITEM TO BE
         CHANGED FOR BUILDING TYPE 55

=14
```

```
BUILDING ID - 55           NAME - IGLOO5
     ROOF - RND
     LENGTH - 120.00 FT
     WIDTH -  30.00 FT
     RADIUS -   26.00 FT
     SIDE WALL HEIGHT -  5.00 FT
     WALL THICKNESS - 0.   FT
     ROOF THICKNESS - 0.   FT
     ENTRANCE HEIGHT -  0.   FT
     ENTRANCE WIDTH -  0.   FT
     DOOR THICKNESS - 0.   FT
     MAXIMUM WEIGHT - 1000.00 TONS

ENTER THE ONE DIGIT TRANSACTION DESIRED:
```

D. Exercising the DELETE option:

```
     ENTER THE ONE DIGIT TRANSACTION DESIRED:

=3
     ENTER THE TWO DIGIT BUILDING TYPE OF RECORD TO BE DELETED
=55
     RECORD 5 HAS BEEN DELETED FROM THE DATA BASE

     ENTER THE ONE DIGIT TRANSACTION DESIRED:

=4
     ENTER THE TWO DIGIT BUILDING TYPE OF RECORD TO BE DISPLAYED
=55
     RECORD 55 DOES NOT EXIST

     ENTER THE ONE DIGIT TRANSACTION DESIRED:
```

E. <u>Exercising the DISPLAY option</u>:

```
     ENTER THE ONE DIGIT TRANSACTION DESIRED:

=4
     ENTER THE TWO DIGIT BUILDING TYPE OF RECORD TO BE DISPLAYED
=5
     RECORD    5 DOES NOT EXIST

     ENTER THE ONE DIGIT TRANSACTION DESIRED:

=4
     ENTER THE TWO DIGIT BUILDING TYPE OF RECORD TO BE DISPLAYED
     BUILDING ID -  1
          NAME - A
          ROOF - FLT
          LENGTH -  60.00 FT
          WIDTH -  25.00 FT
          RADIUS -   0.   FT
          SIDE WALL HEIGHT - 12.00 FT
          WALL THICKNESS - 0.   FT
          ROOF THICKNESS - 0.   FT
          ENTRANCE HEIGHT -  0.   FT
          ENTRANCE WIDTH -  0.   FT
          DOOR THICKNESS - 0.   FT
          MAXIMUM WEIGHT -    0.   TONS

     ENTER THE ONE DIGIT TRANSACTION DESIRED:
```

F. <u>Exercising the TERMINATE option</u>:

```
     ENTER THE ONE DIGIT TRANSACTION DESIRED:

=5
     YOU ARE NOW EXITING THE UPDATE PROGRAM

               ADDED -    1
               CHANGED -    1
               DELETED -    1
               TOTAL NUMBER OF RECORDS IN DATA BASE -    4
```

2. Munition Storage Area Data Base Utility Program examples:

A. <u>Starting the MSADBUP program</u>:

```
SYSTEM ?79C06/OBJECT/MSADBUP - - enter the appropriate file id
     WELCOME TO THE MUNITION STORAGE AREA DATA BASE

     ENTER THE ONE DIGIT TRANSACTION DESIRED:
=  - - just press RETURN key to display options
     OPTIONS:
                         1 - ADD RECORD
                         2 - CHANGE RECORD
                         3 - DELETE RECORD
                         4 - DISPLAY RECORD
                         5 - TERMINATE JOB

     ENTER THE ONE DIGIT TRANSACTION DESIRED:
```

B. Exercising the ADD option:

```
     ENTER THE ONE DIGIT TRANSACTION DESIRED:
=1
     ENTER THE 6 CHARACTER BUILDING NUMBER OF RECORD TO BE ADDED
=12345
     ENTER THE INFORMATION FOR THE NAME (AAAAAA)

=ABCDEF
     ENTER THE INFOMATION FOR THE TYPE (99)

     ENTER THE INFORMATION FOR THE CLASS/DIV 1.1 NEW (9999999) IN LBS

=123456
     ENTER THE INFORMATION FOR THE CLASS/DIV/CAT 1.2 04 NEW (9999999) IN
LBS
=1234567
     ENTER THE INFORMATION FOR THE CLASS/DIV/CAT 1.2 08 NEW (9999999) IN
LBS
=12345
     ENTER THE INFORMATION FOR THE CLASS/DIV/CAT 1.2 12 NEW (9999999) IN
LBS
=1234
     ENTER THE INFORMATION FOR THE CLASS/DIV/CAT 1.2 18 NEW (9999999) IN
LBS
=123
     ENTER THE INFORMATION FOR THE CLASS/DIV 1.3 NEW (9999999) IN LBS

=12
     ENTER THE INFORMATION FOR THE CLASS/DIV 1.4 NEW (9999999) IN LBS

=1
     BUILDING NR - 12345
          NAME - ABCDEF
          TYPE - 65
          CLASS/DIV 1.1 NEW -  123456 LBS
          CLASS/DIV/CAT 1.2 04 NEW - 1234567 LBS
          CLASS/DIV/CAT 1.2 08 NEW -   12345 LBS
          CLASS/DIV/CAT 1.2 12 NEW -    1234 LBS
          CLASS/DIV/CAT 1.2 18 NEW -     123 LBS
          CLASS/DIV 1.3 NEW -      12 LBS
          CLASS/DIV 1.4 NEW -       1 LBS
     ENTER THE ONE DIGIT TRANSACTION DESIRED:
=1
     ENTER THE 6 CHARACTER BUILDING NUMBER OF RECORD TO BE ADDED
=1A
     RECORD 1A ALREADY EXISTS

     ENTER THE ONE DIGIT TRANSACTION DESIRED:
```

C. Exercising the CHANGE option:

```
    ENTER THE ONE DIGIT TRANSACTION DESIRED:
=2
    ENTER THE 6 CHARACTER BUILDING NUMBER OF RECORD TO BE CHANGED
=999
    RECORD 999 DOES NOT EXIST

    ENTER THE ONE DIGIT TRANSACTION DESIRED:
=2
    ENTER THE 6 CHARACTER BUILDING  NUMBER  OF  RECORD  TO  BE  CHANGED
=12345
    ENTER THE 2 DIGIT ITEM NUMBER OF THE ITEM TO BE
         CHANGED FOR BUILDING NR 12345

=   - - just press the RETURN key to display options
    OPTIONS:
         02 - NAME (6A)
         03 - STD TYPE BLDG (99)
         04 - CLASS/DIV 1.1 NEW (9999999) IN LBS
         05 - CLASS/DIV/CAT 1.2 04 NEW (9999999) IN LBS
         06 - CLASS/DIV/CAT 1.2 08 NEW (9999999) IN LBS
         07 - CLASS/DIV/CAT 1.2 12 NEW (9999999) IN LBS
         08 - CLASS/DIV/CAT 1.2 18 NEW (9999999) IN LBS
         09 - CLASS/DIV 1.3 NEW (9999999) IN LBS
         10 - CLASS/DIV 1.4 NEW (9999999) IN LBS

         11 - FINISHED WITH THIS RECORD

    ENTER THE 2 DIGIT ITEM NUMBER OF THE ITEM TO BE
         CHANGED FOR BUILDING NR 12345

=4
    ENTER THE INFORMATION FOR THE CLASS/DIV 1.1 NEW (9999999) IN LBS
=1234567
    ENTER THE 2 DIGIT ITEM NUMBER OF THE ITEM TO BE
         CHANGED FOR BUILDING NR 12345

=5
    ENTER THE INFORMATION FOR THE CLASS/DIV/CAT 1.2 04 NEW (9999999) IN
LBS
=123456
    ENTER THE 2 DIGIT NUMBER OF THE ITEM TO BE
         CHANGED FOR BUILDING NR 12345

=11
```

```
BUILDING NR - 12345
     NAME - ABCDEF
     TYPE - 65
     CLASS/DIV 1.1 NEW - 1234567 LBS
     CLASS/DIV/CAT 1.2 04 NEW -  123456 LBS
     CLASS/DIV/CAT 1.2 08 NEW -   12345 LBS
     CLASS/DIV/CAT 1.2 12 NEW -    1234 LBS
     CLASS/DIV/CAT 1.2 18 NEW -     123 LBS
     CLASS/DIV 1.3 NEW -      12 LBS
     CLASS/DIV 1.4 NEW -       1 LBS
ENTER THE ONE DIGIT TRANSACTION DESIRED:
```

D. Exercising the DELETE option:

```
     ENTER THE ONE DIGIT TRANSACTION DESIRED:
=3
     ENTER THE 6 CHARACTER BUILDING NUMBER OF RECORD TO BE DELETED
=123
     RECORD 123    DOES NOT EXIST

     ENTER THE ONE DIGIT TRANSACTION DESIRED:
=3
     ENTER THE 6 CHARACTER BUILDING NUMBER OF RECORD TO BE DELETED
=12345
     RECORD 12345  HAS BEEN DELETED FROM THE DATA BASE

     ENTER THE ONE DIGIT TRANSACTION DESIRED:
```

E. Exercising the DISPLAY option:

```
     ENTER THE ONE DIGIT TRANSACTION DESIRED:
=4
     ENTER THE 6 CHARACTER BUILDING NUMBER OF RECORD TO BE DISPLAYED
=12345
     RECORD 12345  DOES NOT EXIST

     ENTER THE ONE DIGIT TRANSACTION DESIRED:
=4
     ENTER THE 6 CHARACTER BUILDING NUMBER OF RECORD TO BE DISPLAYED
=1A
     BUILDING NR - 1A
          NAME -
          TYPE - 4
          CLASS/DIV 1.1 NEW -      100 LBS
          CLASS/DIV/CAT 1.2 04 NEW -      500 LBS
          CLASS/DIV/CAT 1.2 08 NEW -      400 LBS
          CLASS/DIV/CAT 1.2 12 NEW -      300 LBS
          CLASS/DIV/CAT 1.2 18 NEW -      200 LBS
          CLASS/DIV 1.3 NEW -      600 LBS
          CLASS/DIV 1.4 NEW -      700 LBS
     ENTER THE ONE DIGIT TRANSACTION DESIRED:
```

F. Exercising the TERMINATE option:

```
     ENTER THE ONE DIGIT TRANSACTION DESIRED:
=5
     YOU ARE NOW EXITING THE UPDATE PROGRAM

               ADDED -    1
               CHANGED -    2
               DELETED -    1
               TOTAL NUMBER OF RECORDS IN DATA BASE -     5

SYSTEM ?
```

3. National Stock Number Data Base Utility Program examples:

A. Starting the NSNDBUP program:

```
SYSTEM ?79C06/OBJECT/NSNDBUP - -enter the appropriate file id
     WELCOME TO THE MUNITION NATIONAL STOCK NUMBER DATA BASE

     ENTER THE ONE DIGIT TRANSACTION DESIRED:
=   - - just press the RETURN key to display options
     OPTIONS:
                             1 - ADD RECORD
                             2 - CHANGE RECORD
                             3 - DELETE RECORD
                             4 - DISPLAY RECORD
                             5 - TERMINATE JOB

     ENTER THE ONE DIGIT TRANSACTION DESIRED:
```

B. <u>Exercising the ADD option</u>:

```
     ENTER THE ONE DIGIT TRANSACTION DESIRED:
=1
     ENTER THE 18 CHARACTER STOCK NUMBER OF RECORD TO BE ADDED
=1325-00-710-6771
     RECORD 1325-00-710-6771   ALREADY EXISTS

     ENTER THE ONE DIGIT TRANSACTION DESIRED:
=1
     ENTER THE 18 CHARACTER STOCK NUMBER OF RECORD TO BE ADDED
=1330-00-679-6043
     ENTER THE INFORMATION FOR THE STACK HEIGHT (9999) IN PACKS
=
     ENTER THE INFORMATION FOR THE PACKAGE HEIGHT (999.9) IN FT
=14.5
     ENTER THE INFORMATION FOR THE PACKAGE WIDTH (999.9) IN FT
=3.4
     ENTER THE INFORMATION FOR THE PACKAGE LENGTH (999.9) IN FT
=6.8
     ENTER THE INFORMATION FOR THE UNITS PER PACK (9999)
=50
     ENTER THE INFORMATION FOR THE PACKAGE GR WT (99999.9999) IN LBS
=3478
     ENTER THE INFORMATION FOR THE PACKAGE NEW (99999.9999) IN LBS
=402
     ENTER THE INFORMATION FOR THE COMPATBL GROUP (A)
=C
     ENTER THE INFORMATION FOR THE CLASS/DIVISION (9.9)
=1.2
     ENTER THE INFORMATION FOR THE CATEGORY (99)
=18
     NATIONAL STK NR - 1330-00-679-6043
          STACKING HEIGHT -     0 PACKAGES
          PACKAGE HEIGHT -   4.5 FT
          PACKAGE WIDTH -   3.4 FT
          PACKAGE LENGTH -   6.8 FT
          UNITS PER PACKAGE -   50
          PACKAGE GROSS WT -  3478.0000 LBS
          PACKAGE NEW -   402.0000 LBS
          COMPATIBILITY GROUP - C
          CLASS/DIVISION - 1.2
          CATEGORY - 18

     ENTER THE ONE DIGIT TRANSACTION DESIRED:
```

C. Exercising the CHANGE option:

```
        ENTER THE ONE DIGIT TRANSACTION DESIRED:
=2
        ENTER THE 18 CHARACTER STOCK NUMBER OF RECORD TO BE CHANGED
=1333-00-679-6043
        RECORD 1333-00-679-6043   DOES NOT EXIST

        ENTER THE ONE DIGIT TRANSACTION DESIRED:
=2
        ENTER THE 18 CHARACTER STOCK NUMBER OF RECORD TO BE CHANGED
=1330-00-679-6043
        ENTER THE 2 DIGIT ITEM NUMBER OF THE ITEM TO BE
             CHANGED FOR STOCK NR 1330-00-679-6043

=   - - just press the RETURN key to display the options
        OPTIONS:
             02 - STACKING HEIGHT (9999) IN PACKAGES
             03 - PACKAGE HEIGHT (999.9) IN FT
             04 - PACKAGE WIDTH (999.9) IN FT
             05 - PACKAGE LENGTH (999.9) IN FT
             06 - UNITS PER PACKAGE (9999)
             07 - PACKAGE GROSS WT (99999.9999) IN LBS
             08 - PACKAGE NEW (99999.9999) IN LBS
             09 - COMPATIBILITY GROUP (A)
             10 - CLASS/DIVISION (9.9)
             11 - CATEGORY (99)

             12 - FINISHED WITH THIS RECORD

        ENTER THE 2 DIGIT ITEM NUMBER OF THE ITEM TO BE
             CHANGED FOR THE STOCK NR 1330-00-679-6043

=9
        ENTER THE INFORMATION FOR THE COMPATBL GROUP (A)
=B
        ENTER THE 2 DIGIT ITEM NUMBER OF THE ITEM TO BE
             CHANGED FOR STOCK NR 1330-00-679-6043

=10
        ENTER THE INFORMATION FOR THE CLASS/DIVISION (9.9)
=1.3
        ENTER THE 2 DIGIT ITEM NUMBER OF THE ITEM TO BE
             CHANGED FOR STOCK NR 1330-00-679-6043

=11
        ENTER THE INFORMATION FOR THE CATEGORY (99)
=
        ENTER THE 2 DIGIT ITEM NUMBER OF THE ITEM TO BE
             CHANGED FOR STOCK NR 1330-00-679-6043

=12
```

```
NATIONAL STK NR - 1330-00-679-6043
     STACKING HEIGHT -     0 PACKAGES
     PACKAGE HEIGHT -   4.5 FT
     PACKAGE WIDTH -   3.4 FT
     PACKAGE LENGTH -   6.8 FT
     UNITS PER PACKAGE -   50
     PACKAGE GROSS WT -  3478.0000 LBS
     PACKAGE NEW -   402.0000 LBS
     COMPATIBILITY GROUP - B
     CLASS/DIVISION - 1.3
     CATEGORY - 0

ENTER THE ONE DIGIT TRANSACTION DESIRED:
```

D. <u>Exercising the DELETE option</u>:

```
     ENTER THE ONE DIGIT TRANSACTION DESIRED:
=3
     ENTER THE 18 CHARACTER STOCK NUMBER OF RECORD TO BE DELETED
=1333-00-679-6043
     RECORD 1333-00-679-6043   DOES NOT EXIST

     ENTER THE ONE DIGIT TRANSACTION DESIRED:
=3
     ENTER THE 18 CHARACTER STOCK NUMBER OF RECORD TO BE DELETED
=1330-00-679-6043
     RECORD 1330-00-679-6043   HAS BEEN DELETED FROM THE DATA BASE

     ENTER THE ONE DIGIT TRANSACTION DESIRED:
```

E. Exercising the DISPLAY option:

```
     ENTER THE ONE DIGIT TRANSACTION DESIRED:
=4
     ENTER THE 18 CHARACTER STOCK NUMBER OF RECORD TO BE DISPLAYED
=1330-00-679-6043
     RECORD 1330-00-679-6043   DOES NOT EXIST

     ENTER THE ONE DIGIT TRANSACTION DESIRED:
=4
     ENTER THE 18 CHARACTER STOCK NUMBER OF RECORD TO BE DISPLAYED
=1325-00-710-6771
     NATIONAL STK NR - 1325-00-710-6771
          STACKING HEIGHT -     0 PACKAGES
          PACKAGE HEIGHT -   3.3 FT
          PACKAGE WIDTH -   3.3 FT
          PACKAGE LENGTH -   4.0 FT
          UNITS PER PACKAGE -   12
          PACKAGE GROSS WT -     0.     LBS
          PACKAGE NEW -  1200.0000 LBS
          COMPATIBILITY ROUP - D
          CLASS/DIVISION - 1.1
          CATEGORY -  0

     ENTER THE ONE DIGIT TRANSACTION DESIRED:
```

F. Exercising the TERMINATE option:

```
     ENTER THE ONE DIGIT TRANSACTION DESIRED:
=5
     YOU ARE NOW EXITING THE UPDATE PROGRAM

               ADDED -   1
               CHANGED -   1
               DELETED -   1
               TOTAL NUMBER OF RECORDS IN DATA BASE -   5

SYSTEM ?
```

4. Format Generator Program examples:

```
SYSTEM ?79C06/OBJECT/LPGEN - - enter the appropriate file string
     WELCOME TO THE INVENTORY PROGRAM

     ENTER THE NATIONAL STOCK NUMBER OF MUNITION TO BE
          ENTERED IN INVENTORY AND HIT RETURN KEY
          IF FINISHED ENTER '*' AND HIT RETURN
=1325-00-710-6771

     ENTER THE NUMBER OF LOTS FOR MUNITION 1325-00-710-6771
=3
     ENTER THE NUMBER OF PACKAGES FOR LOT  1
=400
     ENTER THE NUMBER OF PACKAGES FOR LOT  2
=350
     ENTER THE NUMBER OF PACKAGES FOR LOT  3
=400

     ENTER NEXT STOCK NR OR '*' IF FINISHED
=1325-00-453-9903

     STOCK NUMBER 1325-00-453-9903   IS REJECTED BECAUSE NEW = 0

     ENTER NEXT STOCK NR OR '*' IF FINISHED
=*

     DO YOU WISH TO STOP PROGRAM NOW? (Y OR N)
=N - - if you answer "Y" the program will stop


     STORAGE FACILITY DATA IS NOW BEING GENERATED


     PLEASE READ THIS QUESTION CAREFULLY...

        DO YOU WISH TO ENTER A DIFFERENT PERCENTAGE OF USABLE
     VOLUME FOR EACH BUILDING? (Y OR N)

=Y  - *** the following question will be repeated for EACH storage building

     ENTER THE PERCENTABE OF USABLE VOLUME FOR BUILDING XXXXXX E.G.,75.8
=** - - enter the percentage
     .
     .
   if you answer "N" to the above question, the computer will display:

     ENTER THE PERCENTAGE OF BUILDING VOLUME THAT
          IS CONSIDERED USABLE, E.G., 75.8
=** - - enter the percentage
```

and in either case the computer will continue with:

.

.

.

GENERATING OBJECTIVE FUNTCION AND CONSTRAINTS NOW,
PLEASE WAIT....

THE OBJECTIVE FUNCTION IS NOW BEING ORGANIZED

NOW GENERATING MUNITION CONSTRAINTS....

STARTING THE BUILDING VOLUME CONSTRAINTS....

GENERATING GROUP AND SUBGROUP CONSTRAINTS NOW....

CONCLUDING BY GENERATING RHS...

MUNITION INVENTORY CROSS REFERENCE LIST

ID NR	STOCK NUMBER	LOT	PACKAGES
1	1325-00-710-6771	1	400
2	1325-00-710-6771	2	350
3	1325-00-710-6771	3	400

GROUP CROSS REFERENCE LIST

ID NR	GROUP
1	A
2	B
3	C
4	D
5	E
6	F
7	G
8	H
9	J
10	K
11	L
12	S

CLASS CROSS REFERENCE LIST

ID NR	CLASS
1	1.1
2	1.2/18
3	1.2/12
4	1.2/08
5	1.2/04
6	1.3
7	1.4

THIS PROBLEM CONTAINS 23 DECISION VARIABLES IN THE
OBJECTIVE FUNCTION AND 18 CONSTRAINTS

SPAWNING THE LP JOB NOW...

BYE

SNUMB # XXXXT

SYSTEM ?

Appendix E

LP/600 Output

After the user completes a LPGEN session, the computer will display the system identification number (item 1 in the example at the end of this appendix) for the LP/600 job. The user should keep a record of these job numbers to use in identifying the job listings.

The first three pages of the job listing are called the execution report. The user can determine if the three activities of the job were successfully completed by examining the execution report. Each BEGIN ACTIVITY message should have a corresponding NORMAL TERMINATION message (see items 2 , 3 , 4). The next three pages of the job listing will contain the Agendum Control Program (item 5) responsible for controlling the LP/600 package.

The following page starts the row or constraint information for the inventory problem and is identified by the title ROWS (item 6). Each row is identified (item 7) by a KJ (internal identification) number, type (ZERO for = constraint, PLUS for < constraint), and row (constraint) name. An * appearing after a KJ number implies that preceding rows not printed have zero L, PI, and RHS column values. The LOGICAL INDIC. column (item 8) indicates the status of each row and takes on one of the following values;

*BASIS - row is in the basis (final solution) and not at its bound;

*ATBND - row is in the basis and is at its bound (constraint is completely used up);

blank - row is not in the basis.

The L-VALUE column (item 9) defines the slack or unused value for each row, e.g. the unused volume of a particular building. The PI column (item 10) identifies the marginal value - the amount of change in the slack value (L-VALUE) of the objective function (OBJECTIVE) row per unit increase in the right hand side value (RHS) for the row - for each row. The RHS column (item 11) displays the original right hand side values. There are four classes of constraints produced by LPGEN:

MUXXXX - each munition in the inventory is identified by internal identification number XXXX (see munition cross reference for associated national stock number).

BLDGXXXXXXVOL - identifies the volume constraint for building XXXXXX.

SSETXXXXXX - identifies the special set constraint for building XXXXXX.

BLDGXXXXXXGYCZ - identifies the subgroup net explosive weight constraint for building XXXXXX, group Y, and most restrictive class Z in this particular subgroup.

After all rows have been displayed, the next page will start the column or decision variable information and is identified by the title COLUMNS (item 12). As in the ROW output, the first three columns (item 13) identify each of the columns by a KJ number, type (INTEGER for integer decision variables, PLUS for > 0 decision variables, SSET for the special set decision variables), and column (decision variable) name. An * after the KJ column number indicates that preceding columns not printed have zero X, DJ, and COST*SCALE column values. The STRUCT. INDIC. column (item 14) indicates the status of each column and takes on one of the following values:

*BASIS - column is in the basis (final solution) and not at its bound;

*ATBND - column is in the basis and is at its bound (maximum value);

blank - column is not in the basis.

The X-VALUE column (item 15) displays the number of packages of the indicated munition to be stored in the indicated building. The user will have to use the munition cross reference list discussed later to identify the actual national stock number and lot of this munition. For example, column MU1 C1BD1A refers to munition 1 to be stored in building 1A. Munition 1 might actually refer to lot 1 of national stock number 1325-00-710-6771. If fractions of packages are displayed in the X-VALUE column simply round to the nearest whole number. The DJ column (item 16) displays the amount by which the building load coefficient (COST*SCALE) would have to be changed before the munition/building/subgroup variable could profitably be introduced into the basis. Another meaning of the DJ column value is the amount of change in the value of the objective function (OBJECTIVE row) if one package of the munition/building/subgroup variable was forced into the final solution. The COST*SCALE column (item 17) displays the original building load coefficient for each column of the objective function. The RANGES column (item 18) displays the lower and upper bounds of each decision variable that is classified as an INTEGER type variable. There are three classes of decision variables produced by LPGEN:

MUXXXXCYBDZZZZZZ - XXXX is the internal munition/lot identification (see munition cross reference for associated national stock number and lot number), Y is the interanl identification of the most restrictive class of munitions that X-VALUE packages of this munition are stored with (see class cross reference list for actual class/division), in building ZZZZZZ.

LEFT OVER MUXXXX - X-VALUE packages of munition XXXX could not be stored in the munition storage area.

SSBLDGXXXXXXGYCZ - a special set variable that allows for the selection of only one constraint from a set of net explosive weight constraints for each building to be used. The X-VALUE of this class of variable will be either 1 - if used or 0 - if not used. XXXXXX is the building number, Y is the internal identification of the group (see group cross reference list for actual group), and Z is the internal identification of the most restrictive class of munitions contained in the subgroup (see class cross reference list for the actual class/division).

After all columns have been displayed, the three cross reference lists will be printed. The munitions cross reference (item 19) displays the internal munition identification, national stock number, lot number, and number of packages of each munition entered in inventory. The group cross reference (item 20) displays the internal group identifcation number and associated compatability group. The class cross reference (item 21) displays the internal class identification number and associated munition class/division/category as defined in AFR 127-100.

The following hypothetical example was composed, using the procedure described in section 4 of Appendix D in this manual. The munitions cross reference (item 19) indicates that three lots of munition national stock number 1325-00-710-6771, containing 400, 350, and 400 packages respectively, were entered into the inventory. One package of munition 1325-00-710-6771 is 3.3 ft by 3.3 ft by 4.0 ft, which is 43.56 cubic feet and contains 1200 pounds of NEW. This munition belongs to class 1.1 and compatibility group D. The munition storage area is composed of five buildings:

BLDG	LEN	WID	SIDE WALL	RADIUS	1.1 NEW(LBS)
1A	40.3	26.5	0.0	13.4	100000
2B	60.0	25.0	12.0	0.0	100000
3C	100.0	50.0	12.0	0.0	100000
4D	60.0	25.0	12.0	13.0	100
5E	60.0	25.0	12.0	0.0	1000

The COLUMNS page (item 12) displays the answers to this munitions inventory problem. The user only needs to search the X-VALUE column (item 15) for non-zero entries to determine which munitions will be stored where, e.g. the first entry (item 22) implies that 83 packages of munition 1 should be stored in building 1A. The overall results of this example are:

	BUILDING					
MUN	1A	2B	3C	4D	5E	LEFT OVER
1	83		83			234
2		83				267
3						400

Any non-zero value appearing in the left over columns indicates that not all of the inventory could be stored in the storage area. The storage area load factor (item 23) displays a relative value of the density of the storage arrangement of the inventory i.e., large values imply high density storage configurations.

```
0004  $       IDENT    #####,AFIT, MUNITION OPTIMIZATION

$$    L477T ENTERED CREATE AT 02.026 FROM TSS/S    1-05-22

0001  $       SNUMB    L477T
0002  $       COMMENT  79C06              TSS CARDIN
0003  $$      USERID   79C06$#########
0004  $       IDENT    #####,AFIT, MUNITION OPTIMIZATION
0005  $       ENTRY    .LHSP
0006  $       USE      .LHSP
0007  $       EXECUTE
0008  $       LIMITS   ,15,029K,,40.5K
0009  $$      PRMFL    H*,R,R,AF,LIB/LP.PAC
0010  $$      PRMFL    SO,W,L,79C06/DATA/LPOUT
0011  $       DISC     AA,A1,10R
0012  $       DISC     AB,A2,10R
0013  $       DISC     AC,A3,10R
0014  $       DISC     AD,A4,10R
0015  $       DISC     AE,A5,10R
0016  $$      PRMFL    IN,R,L,79C06/DATA/LPINFO
0017  $       DATA     I*
0018  $       CONVER   NSPIN
0019  $       LIMITS   1,6K,,2K
0020  $$      PRMFL    IN,R,L,79C06/DATA/LPOUT
0021  $       REMOTE   OT
0022  $       CONVER   NSPIN
0023  $       LIMITS   1,6K,,2K
0024  $$      PRMFL    IN,R,L,79C06/DATA/CRSREP
0025  $       REMOTE   OT
0026  $       ENDJOB
  TOTAL CARD COUNT THIS JOB = 00.039

* BEGIN ACTIVITY - 1-  GELOAD    12/07/79    SB=000000000000
```

1

2

2

```
* NORMAL TERMINATION     AT 032056 I=4000 SW=000000000000

    START  2.221     LINES     26     PROC  0.0042     I/O   0.005    XU  5   MEMORY    29K
    STOP   2.243     LIMIT   5120     LIMIT 0.1500     LIMIT          CU  5   M*T      2420
    SWAP   0.001
    LAPSE  0.022   PC D TYPE     BUSY    TP/AP     PP/RT    IS/#C MS/#B    ADDRESS T#

                   I* R D180 *     27       #        1        1     1      1-05-16
                   R* R D180 *     48       #        0        1     1      1-05-16
                   M* R D180 P   6526       #        0      840   840B     0-04-17
                   SO R D180 P    427       #        0       56    56      1-05-07
                   AA R D180 *   3181       #        0      120   120B     0-04-05
                   AB R D180 *   1919       #        0      120   120B     0-04-07
                   AC R D180 *    769       #        0      120   120B     0-04-05
                   AD R D180 *    217       #        0      120   120B     0-04-07
                   AE R D180 *     79       #        0      120   120B     0-04-05
                   IN R D180 P    258       #        4       22    22      1-05-07
                   P*   STOUT
                   L* R D270 *    289       #        0      540   540B     0-00-01
                   *L R D270 *     26       #        0       60    60B     0-00-02

    LIST     26 LINES AT STA.  SL

 * BEGIN ACTIVITY - 2-  CONVER   12/07/79    SW=000000000000
  SMC-TTL DATE      730615
 INPUT COUNT        000133       OUTPUT COUNT       000133
 BLOCKS SKIPPED     000000       IGNORE COUNT       000000
  * NORMAL TERMINATION      AT 00.676 I=2060 SW=000000000000

    START  2.244     LINES    133     PROC  0.0002     I/O   0.001    XU  5   MEMORY     6K
    STOP   2.246     LIMIT   2048     LIMIT 0.0100     LIMIT          CU  5   M*T        53
    SWAP   0.000
    LAPSE  0.002   PC D TYPE     BUSY    TP/AP     PP/RT    IS/#C MS/#B    ADDRESS T#

                   IN R D180 P    308       #        9       56    56      1-05-07
                   OT   STOUT
                   P*   STOUT

    LIST    133 LINES AT STA.  SL

 * BEGIN ACTIVITY - 3-  CONVER   12/07/79    SW=000000000000
  SMC-TTL DATE      730615
 INPUT COUNT        000040       OUTPUT COUNT       000040
 BLOCKS SKIPPED     000000       IGNORE COUNT       000000
  * NORMAL TERMINATION      AT 00.676 I=2060 SW=000000000000

    START  2.247     LINES     40     PROC  0.0002     I/O   0.001    XU  5   MEMORY     6K
    STOP   2.249     LIMIT   2048     LIMIT 0.0100     LIMIT          CU  5   M*T        55
    SWAP   0.000
    LAPSE  0.002   PC D TYPE     BUSY    TP/AP     PP/RT    IS/#C MS/#B    ADDRESS T#

                   IN R D180 P     77       #        2       10    10      1-05-07
                   OT   STOUT
                   P*   STOUT

    RC-13    40 LINES AT STA.  SL
```

3

4

```
L477T 01   12-07-79     2.223                                                                    PAGE        1

    ORIGIN    DATE MODULE ENTRY LOCATION      ENTRY LOCATION      ENTRY LOCATION      ENTRY LOCATION      ENTRY LOCATION

                                  SUBPROGRAMS INCLUDED IN DECK.

                            $        ENTRY     .LHSP
                            $        USE       .LHSP

                                  SUBPROGRAMS OBTAINED FROM SYSTEM LIBRARY

    071762 07/09/72 FLHS .LHSP   071762       .LHSNB 071767
    071674 05/24/73 GSET .SETU.  071701
                                                 RANGE                SIZE
                            ALLOCATED CORE    000005 THRU 071777      072000
                               RELOCATABLE    071674 THRU 071777      000104
                            $        PPMFL    H*,R,R,AP.LIB/LP.PAC
                            $        PRMFL    SO,W,L,79C06/DATA/LPOUT
                            $        DISC     AA,A1,10R
                            $        DISC     AB,A2,10R
                            $        DISC     AC,A3,10R
                            $        DISC     AD,A4,10R
                            $        DISC     AE,A5,10R
                            $        PRMFL    IN,R,L,79C06/DATA/LPINFO
                            $        DATA     I*
                                  1K, IS THE MINIMUM MEMORY NEEDED TO LOAD THIS ACTIVITY 730517 F/8
                            000012 LOCATIONS REQUIRED FOR LOAD TABLE
                            EXECUTION PROGRAM ENTERED AT  071762    THROUGH   .SETU.
```

AGENDUM CONTROL PROGRAM

```
TERM    OUTPUT
ABJOB   SAVE
        PUNCH
        ENDLP
STATX   STATUS
ENDLP   ENDLP
DUMPN   DUMP
NEXT    NEXT
LPDUMP  LPDUMP
JUMP    JUMP
ISOL    INVERT
        OUTPUT
        CURRENT
DINV    INVERT
CURNT   CURRENT
DCSH    CRASH
        CURRENT
DPRI    PRIMAL
        CURRENT
DPRN    PRIMAL
        NEXT
        MACRO    DECOMP
        PERFORM  DODCM
        ENDM
DODCM   SET      UNB=NEXT,NFS=NEXT,DOPRI= DCPRI
        RESET    CRHS,CRHSW
        DCMONE
        RESET    UNB, NFS
        SET      DOPRI = DODCM
        DCMTWO
        RESET    DOPRI
        NEXT
DCPRI   PRIMAL
        OUTPUT
        CURRENT
DDINV   DCINV
        CURRENT
START   RESET
         PREPROCESS
         TITLE    MUNITION STORAGE OPTIMIZING
         SET      NOSOXO=ON
         CONVERT  SOURCE=AMMO/IN,IDENT=MU
         SETUP    SOURCE=MU
         SET      OBJ=OBJECTIVE,RHS=RHS
         SET      SCALE=-4
         CRASH
         INTEGER
         RESET    NOSOXO
         OUTPUT
```

5

```
                    AGENDUM CONTROL PROGRAM
  50                ENDLP
  51                EXECUTE
 38 START  RESET                                                             8040    9
 39        PREPROCESS                                                        8040    9
 40        TITLE   MUNITION STORAGE OPTIMIZING                               8040    9
 41        SET     NOSOXO=ON                                                 8040    9
```

5

```
L477T 01  12/07/79          MUNITION STORAGE OPTIMIZING                                    VERB6  OUTPU  PAGE     3
PRNAM=MU     :      :       OBJ=OBJECTIVE    :           RHS=RHS    L       :
    49            OUTPUT                                                                      8067     13
    14 DINV     INVERT                                     DOINV      DEMAND 45676            8058     13
    15 CURNT    CURRENT                                                                       8070     14
    49            OUTPUT                                                                      8071     14
```
5

L477T 01 12/07/79 MUNITION STORAGE OPTIMIZING VERB= OUTPU PAGE 6

PRNAM=MU : FUNCT= 826.183822 OBJ=OBJECT:IVE : RHS=RHS : :

ROWS 6 7 8 9 10 11

LOGICAL

ROW	KJ	TYPE	ROW NAME			INDIC.	L=VALUE	PI	RHS	RANGES
1		FREE	OBJECT IVE			*BASIS	826.18382040+	1.00000000-	.+	
2		ZERO	MU1				.+	1.00000000-	400.00000000+	
3		ZERO	MU2				.+	1.00000000-	350.00000000+	
4		ZERO	MU3				.+	1.00000000-	400.00000000+	
5		PLUS	BLDG	1A	VOL	*BASIS	3281.50295575+	.+	6911.50299072+	
6		PLUS	BLDG	2B	VOL	*BASIS	9869.99996503+	.+	13500.00000000+	
7		PLUS	BLDG	3C	VOL	*BASIS	41369.99996503+	.+	45000.00000000+	
8		PLUS	BLDG	4D	VOL	*BASIS	10067.68188961+	.+	10071.31188965+	
9		PLUS	BLDG	5E	VOL	*BASIS	13463.69999965+	.+	13500.00000000+	
10		PLUS	BLDG	1A	G4C1		.+	.00127101+	.+	
11		ZERO	SSET	1A			.+	127.10094576+	1.00000000+	
12		PLUS	BLDG	2B	G4C1		.+	.00105740+	.+	
13		ZERO	SSET	2B			.+	105.74074058+	1.00000000+	
14		PLUS	BLDG	3C	G4C1		.+	.00090055+	.+	
15		ZERO	SSET	3C			.+	90.05555580+	1.00000000+	
16		PLUS	BLDG	4D	G4C1		.+	.00083363+	.+	
17		ZERO	SSET	4D			.+	.08336337+	1.00000000+	
18		PLUS	BLDG	5E	G4C1		.+	.00083557+	.+	
19		ZERO	SSET	5E			.+	.83557407+	1.00000000+	

L477T 01 12/07/79 MUNITION STORAGE OPTIMIZING VERB= OUTPU PAGE 5

PRNAM=MU : PUNCT= 826.1838224 OBJ=OBJECT:IVE : RHS=RHS (:

COLUMNS 12 13 14 15 16 17 18

22

COL NO	TYPE	COLUMN NAME			STRUCT. INDIC.	X-VALUE	DJ		COST*SCALE	RANGES	
2	RANGE	MU1	C1BD	1A	*BASIS	83.33333333+	.	-	.52521135-	.0	400.0
21	RANGE	MU1	C1BD	2B		. +	.	-	.26888888-	.0	400.0
22	RANGE	MU1	C1BD	3C	*BASIS	83.33333333+	.	-	.08066667-	.0	400.0
23	RANGE	MU1	C1BD	4D		. +	.	-	.00036043-	.0	400.0
24	RANGE	MU1	C1BD	5E	*BASIS	.83333333+	.	-	.00268889-	.0	400.0
25	RANGE	MU2	C1BD	1A		. +	.	-	.52521135-	.0	350.0
26	RANGE	MU2	C1BD	2B	*BASIS	83.33333333+	.	-	.26888888-	.0	350.0
27	RANGE	MU2	C1BD	3C		. +	.	-	.08066667-	.0	350.0
28	RANGE	MU2	C1BD	4D	*BASIS	.08333333+	.	-	.00036043-	.0	350.0
29	RANGE	MU2	C1BD	5E		. +	.	-	.00268889-	.0	350.0
3	RANGE	MU3	C1BD	1A		. +	.	-	.52521135-	.0	400.0
31	RANGE	MU3	C1BD	2B		. +	.	-	.26888888-	.0	400.0
32	RANGE	MU3	C1BD	3C		. +	.	-	.08066667-	.0	400.0
33	RANGE	MU3	C1BD	4D		. +	.	-	.00036043-	.0	400.0
34	RANGE	MU3	C1BD	5E		. +	.	-	.00268889-	.0	400.0
35	PLUS	LEFT	OVER	MU1	*BASIS	232.50000000+	.	-	1.00000000+		
36	PLUS	LEFT	OVER	MU2	*BASIS	266.58333333+	.	-	1.00000000+		
37	PLUS	LEFT	OVER	MU3	*BASIS	400.00000000+	.	-	1.00000000+		
39	SSET	SSBLDG	1A	G4C1	*BASIS	1.00000000+	.	-	. +		
41	SSET	SSBLDG	2B	G4C1	*BASIS	1.00000000+	.	-	. +		
43	SSET	SSBLDG	3C	G4C1	*BASIS	1.00000000+	.	-	. +		
45	SSET	SSBLDG	4D	G4C1	*BASIS	1.00000000+	.	-	. +		
47	SSET	SSBLDG	5E	G4C1	*BASIS	1.00000000+	.	-	. +		
48	RHS	RHS				. +	826.18382040-		. +		

```
L477T 01  12/07/79          MUNITION STORAGE OPTIMIZING                                   VERB=  ENDLP  PAGE    6
PRNAM=MU    I     I        FUNCT= 826.1838224  OBJ=OBJECTIVE   I          RHS=RHS   (     I
   50        ENDLP                                                                                 8074    14
```

23

SNUMB = L477T, ACTIVITY # = 03, REPORT CODE = 13, RECORD COUNT = 000040

MUNITION INVENTORY CROSS REFERENCE LIST

ID NR	STOCK NUMBER	LOT	PACKAGES
1	1325-00-710-6771	1	480
2	1325-00-710-6771	2	380
3	1325-00-710-6771	3	480

19

GROUP CROSS REFERENCE LIST

ID NR	GROUP
1	A
2	B
3	C
4	D
5	E
6	F
7	G
8	H
9	J
1	K
11	L
12	S

20

CLASS CROSS REFERENCE LIST

ID NR	CLASS
1	1.1
2	1.2/18
3	1.2/12
4	1.2/08
5	1.2/04
6	1.3
7	1.4

21

THIS PROBLEM CONTAINS 23 DECISION VARIABLES IN THE
OBJECTIVE FUNCTION AND 18 CONSTRAINTS

Appendix B

MUNITION STORAGE OPTIMIZING SYSTEM (MSOS)

PROGRAM DOCUMENTATION

This appendix is considered a stand-alone document and will be paged numbered accordingly.

MUNITION STORAGE OPTIMIZING SYSTEM (MSOS)

PROGRAM DOCUMENTATION

Prepared
by
Louis M. Gusmus
3 December 1979

CONTENTS

I. Munition Storage Optimizing System Description

The Munition Storage Optimizing System (MSOS) is designed to operate on the Honeywell 600 or 6000 series computer systems. The MSOS consists of four Time Sharing System FORTRAN programs: the Standard Building Data Base Utility Program (SBDBUP); the Munition Storage Area Data Base Utility Program (MSADBUP); the National Stock Number Data Base Utility Program (NSNDBUP); and the Format Generator Program (LPGEN). Since the utility programs perform the same functions on the different data bases, only a general discussion will be presented about them.

Before using the utility programs the corresponding data bases must be created as sequential ASCII files using one of the following procedures:

(1) If data is to be preloaded, then enter the data according to the appropriate format defined in Appendix A; or

(2) If no data is to be preloaded then the file must have one record that contains contains the value "099b̸" starting in column one.

This record must be deleted during the first run of the corresponding utility program in which data records are added, by requesting to delete record "9". The proper file name of the data base will have to be entered in the INITIAL subroutine of the corresponding utility program source code. This must be done for each of the three data bases. Once the data bases are created, the utility programs can be used.

The flow of the utility program is to first check the mode of operation in subroutine START. If it is not ASCII, then the program requests user to restart in ASCII mode. Next, the data base is opened and a work file created by subroutine OPEN. The data base is copied on to the work file by subroutine INITIAL. Then any desired transactions are performed by subroutine ACTION. When all transactions are completed, the work file is written on to the data base file. Both files are then closed and program execution is terminated. See Appendix B and appropriate appendices for more details.

In order to use LPGEN, three more files must be created:

(1) LPINFO - the formatted output of LPGEN used by the LP/600 package;

(2) CRSREF - the munition, group, and class cross references file; and

(3) LPOUT - the output of the LP/600 package.

These three files should be created as sequential files. Maximum file size should be set to 2000 blocks for LPINFO and 100 blocks for both LPOUT and CRSREF. The actual size of LPINFO is highly dependent on munition inventory and number of storage buildings. LPOUT is not used directly by LPGEN, but must be created prior to exercising LPGEN.

The flow of LPGEN is to first, check the mode of operation in subroutine START. If it is BCD, the program requests user to restart in ASCII mode. Next, the SBDB, MSADB, NSNDB, LPINFO, and CRSREF files are opened and munition, building, decision variables, and JCL work files are created by subroutine OPEN. The user enters the stock number, number of lots, and number of packages in each lot for the munitions inventory in subroutine MUNINV. MUNINV identifies every group/class combination of munition inventory in array MGP for later use, calculates the volume and density factor (NEW / volume) of each package of munition, and writes selected information on the munition work file (file code 02). Since the munition stock number is 18 characters, MUNINV attaches an internal identification number to each munition/lot combination, then writes this information along with the number of packages of the lot to the cross reference file and closes the NSNDB file. Subroutine STORE allows the user to enter the percentage of usable volume for each building separately or once for the entire storage area. Then subroutine BLDVOL calculates the usable volume for each building, writes this information to the building work file (file code 07), and closes the SBDB and MSADB files. Subroutine FORM is the driver routine for formatting the LPINFO file and performs, in succession:

(1) subroutine OBJCTV - responsible for generating the objective function and its coefficients;

(2) subroutine MUNITN - responsible for generating the munition constraints;

(3) subroutine VOLUME - responsible for generating the building volume constraints;

(4) subroutine SSET - responsible for generating special set variables;

(5) subroutine BLDNEW - responsible for generating the NEW constraints for each building; and

(6) subroutine RHANDS - responsible for generating the right hand side of the constraints.

FORM then closes the LPINFO file, the munition, building, and decision variable work files. Subroutine CLOSE writes the group and class cross references to CRSREF, prints a message informing the user how many decision variables and constraints were generated, then closes the CRSREF file. Finally, subroutine SPAWN calculates the time and core requirements based on the number of constraints generated, creates the JCL needed to call LP/600, spawns the job, closes the JCL work file, and terminates the execution of LPGEN. See Appendices I, J, and K for more details.

The most complicated subroutine in LPGEN is BLDNEW. This routine can generate up to 84 constraints per building, depending on the number of munition group/class combinations identified in array MGP. After a building record is read from the building work file, the maximum NEW of each class is sorted in descending order along with the class identifier. The first group (column) of MGP is then checked to see if any of the munition inventory belongs to it. If none belong, each subsequent group is checked until a group if found that contains some of the munitions. The classes in that group are then identified and counted. Starting with the most restrictive NEW class, a constraint is generated that contains all munitions belonging to the current group and class or to less restrictive classes. The next most restrictive NEW class belonging to the current group is found and a constraint is generated that contains all munitions belonging to the current group and to the current class or less restrictive classes. This procedure is continued until all classes of each group have been processed for the current building. Then the next building record is read and the above process is repeated.

The following appendices should give a very detailed view of each of the programs. Appendix A displays the layout of each of the data bases; Appendices B and I are the Program Design Language (PDL) structures used in developing the programs; Appendices C, E, G, and J are computer generated flow charts of the programs; and finally Appendices D, F, H, and K are the source listings.

Appendix B

General Utility Program Flow

LEVEL 1

```
XXXDB      MAIN
  "check mode of operation"
  "open and create needed files"
  "perform desired transactions on records"
  "update the data base"
  "close the files"
END        XXXDB
```

LEVEL 2

```
XXXDB      MAIN

  "if the mode of operation is ASCII continue otherwise restart"
  "open data base file"
  "create work file"
  "read data base into work file"
  "add records"
  "change records"
  "delete records"
  "display records"
  "finished all transactions for data base"
  "write updated data base"
  "stop"

END        XXXDB
```

LEVEL 3

```
XXXDB      MAIN

  PERFORM  START
  IF  "error switch not on"
    THEN
      PERFORM  OPEN
      PERFORM  INITIAL
      PERFORM  ACTION
    ELSE  "stop"
  ENDIF

END        XXXDB
```

```
LEVEL 3

  START      ROUTINE

    IF  "mode is BCD"
      THEN  "turn error switch on and
             print restart message"
      ELSE
    ENDIF

  END      START

LEVEL 3

  OPEN      ROUTINE

    OPEN  "the data base file"
    CREATE  the data base work file"

  END        OPEN

LEVEL 3

  INITIAL      ROUTINE

    READ  "first record from data base file"
    DO WHILE  "not end of data base file"
      WRITE  "record on data base work file"
      INCREMENT  "record counter by 1"
      READ  "next record from data base file"
    ENDDO

  END        INITIAL
```

LEVEL 3

```
ACTION      ROUTINE

  PRINT  "enter transaction desired"
  DO WHILE  "type transaction NE finished"
    READ  "type transaction"
    IF  "type transaction EQ add
      THEN  "add new record"
      ELSE
        IF  "type transaction EQ change"
          THEN  "change existing record"
          ELSE
            IF  "type transaction EQ delete"
              THEN  "delete existing record
              ELSE
                IF  "type transaction EQ display"
                  THEN  "display existing record"
                  ELSE
                ENDIF
            ENDIF
        ENDIF
    ENDIF
    PRINT  "enter next type transaction desired"
  ENDDO
  REWIND  "data base file and work file"
  READ  "first record from work file"
  DO WHILE  "not end of work file"
    WRITE  "record to data base file"
    READ  "next record from work file"
  ENDDO
  PRINT  "transaction summary"

END        ACTION
```

LEVEL 4

```
  ACTION      ROUTINE

    PRINT  "type transaction desired ?"
    READ  "type transaction"
    DO WHILE  "type transaction NE (5) finished all transactions"
      IF  "type transaction EQ (1) add"
        THEN  "ask for new record id"
              "search work file for a match"
              IF  "no match"
                THEN  "add and display new record"
                      INCREMENT  "add counter by 1"
                ELSE  "display message stating record already exists"
              ENDIF
                IF  "type transaction EQ (2) change"
                  THEN  "ask for record id"
                        "search work file for match"
                        IF  "match"
                          THEN  "ask for the item to be changed and
                                  repeat until all items for this
                                record are updated, then display
                                updated record"
                                INCREMENT  "change counter by 1"
                          ELSE  "display message stating
                                  record does not exist"
                        ENDIF
                  ELSE
                    IF  "type transaction EQ (3) delete"
                     THEN  "ask for record id"
                           "search work file for a match"
                           IF  "match"
                              THEN  "delete record and display
                                     message stating record was deleted"
                                     INCREMENT  "delete counter by 1"
                              ELSE  "display message stating record does
                                     not exist"
                           ENDIF
                     ELSE
                       IF  "type transaction EQ (4) display"
                         THEN  "ask for record id"
                               "search work file for match"
                               IF  "match"
                                 THEN  "display the record"
                                 ELSE  "display message stating record
                                         does not exist"
                               ENDIF
                         ELSE
                       ENDIF
                    ENDIF
                ENDIF
      ENDIF
      PRINT  "type transaction desired ?"
      READ  "next type transaction"
    ENDDO
    PERFORM  CLOSE
    PRINT  "transaction summary"
  END        ACTION
```

LEVEL 4

```
CLOSE      ROUTINE

  REWIND  "data base and work file"
  READ  "first record from work file"
  DO WHILE  "not end of work file"
    IF  "record was not deleted"
      THEN  "write record to data base"
            INCREMENT  "total record counter by 1"
      ELSE
    ENDIF
    READ  "next record from work file"
  ENDDO
  CLOSE  "data base and work file"

END        CLOSE
```

LEVEL 5

```
ACTION      ROUTINE

  PERFORM  MESSAG
  READ  "ACT"
  DO WHILE  "ACT NE 5"
    IF  "ACT EQ 1"
      THEN  PERFORM  ITEM  "add"
            PERFORM  MESSAG  "display record"
      ELSE
        IF  "ACT EQ 2"
          THEN  PERFORM  ITEM  "change"
                PERFORM  MESSAG  "display record"
          ELSE
            IF  "ACT EQ 3"
              THEN  PERFORM  DELREC  "delete record"
              ELSE
                IF  "ACT EQ 4"
                  THEN  PERFORM  SEARCH
                        PERFORM  MESSAG  "display record"
                  ELSE
                ENDIF
            ENDIF
        ENDIF
    ENDIF
    PERFORM  MESSAG
    READ  "next ACT"
  ENDDO
  PERFORM  CLOSE
  PERFORM  MESSAG

END        ACTION
```

```
LEVEL 5

  MESSAG      ROUTINE

    "depending on type information needed display:"
      "welcome to utility program"
      "enter type action desired"
      "record does not exist"
      "record already exists"
      "record has been deleted"
      "record description"
      "transaction summary"

  END         MESSAG


LEVEL 5

  ITEM      ROUTINE

    IF  "ACT EQ 1 (add record)"
      THEN  PERFORM  SEARCH
            IF  "record does not exist"
              THEN  "add new record"
                    INCREMENT  "add counter by 1"
                    PERFORM  RITE
              ELSE
            ENDIF
      ELSE  "ACT EQ 2 (change record)"
            PERFORM  SEARCH
            IF  "record exists"
              THEN  PRINT  "enter item number to be changed"
                    READ  "item number to be changed"
                    DO WHILE  "all items are not changed"
                      PRINT  "enter value for item"
                      READ  "value for item"
                      PRINT  "enter next item number to be changed"
                      READ  "next item number to be changed"
                    ENDDO
                    INCREMENT  "change counter by 1"
                    PERFORM  RITE
                    PERFORM  MESSAG
              ELSE
            ENDIF
    ENDIF

  END         ITEM
```

```
LEVEL 5

  DELREC      ROUTINE

    PERFORM  SEARCH
    IF  "record exists"
      THEN  "mark record to be skipped at closing time"
            PERFORM  RITE
            INCREMENT  "delete counter by 1"
      ELSE
    ENDIF

  END        DELREC


LEVEL 5

  SEARCH      ROUTINE

    PRINT  "enter the record you are looking for"
    READ  "record id"
    REWIND  "work file"
    READ  "first record from work file"
    DO WHILE  "not end of work file"
      IF  "match"
        THEN  "identify record as found and stop search"
        ELSE
      ENDIF
      READ  "next record from work file"
    ENDDO

  END        SEARCH


LEVEL 5

  RITE      ROUTINE

    WRITE  "new, changed, or deleted record to work file"

  END        RITE
```

Appendix C

SBDB Utility Program Flow Chart

Appendix A

Data Base Layouts

I. Standard Building Data Base - contains 13 items:

1.	BUILDING TYPE (value of 01 to 99)		
2.	BUILDING TYPE NAME (six character id) not used by MSOS		
3.	ROOF TYPE ("RND" for igloo or "FLT" for other type buildings)		
4.	INNER BUILDING LENGTH	(999.99) FT	
5.	INNER BUILDING WIDTH	(999.99) FT	
6.	ROOF RADIUS	(999.99) FT	
7.	SIDE WALL HEIGHT	(99.99) FT	
8.	WALL THICKNESS	(9.99) FT	*
9.	ROOF THICKNESS	(9.99) FT	*
10.	ENTRANCE HEIGHT	(99.99) FT	* * not used by MSOS
11.	ENTRANCE WIDTH	(99.99) FT	*
12.	DOOR THICKNESS	(9.99) FT	*
13.	BUILDING MAX WT	(9999.99) TONS	*

FORMAT (I2,A6,A3,3F6.2,F5.2,2F4.2,2F5.2,F4.2,F7.2)

II. Munition Storage Area Data Base - contains 10 items:

1.	BUILDING TYPE (value of 01 to 99, must match a record in SBDB)	
2.	BUILDING NAME (six character id) not used by MSOS	
3.	BUILDING NUMBER (six character building number)	
4.	MAX NEW FOR CLASS 1.1	*
5.	MAX NEW FOR CLASS 1.2 CATEGORY 04	*
6.	MAX NEW FOR CLASS 1.2 CATEGORY 08	*
7.	MAX NEW FOR CLASS 1.2 CATEGORY 12	* * * (9999999) LBS
8.	MAX NEW FOR CLASS 1.2 CATEGORY 18	*
9.	MAX NEW FOR CLASS 1.3	*
10.	MAX NEW FOR CLASS 1.4	*

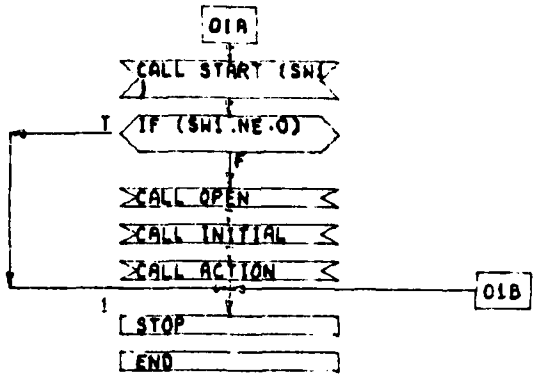
FORMAT (I2,2A6,7I7)

III. National Stock Number Data Base - contains 11 items:

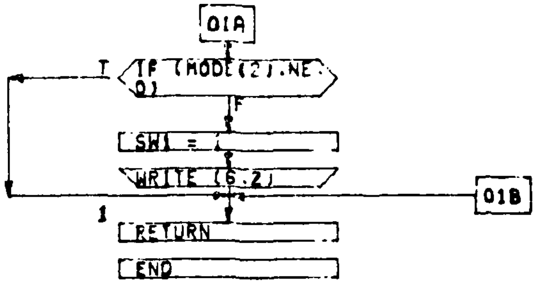
1.	STACKING HEIGHT in packages	(9999) packages
2.	NATIONAL STOCK NUMBER	(9999-999-99-9999AA)
3.	PACKAGE HEIGHT	(999.9) FT
4.	PACKAGE WIDTH	(999.9) FT
5.	PACKAGE LENGTH	(999.9) FT
6.	UNITS PER PACKAGE	(9999) UNITS
7.	PACKAGE GROSS WEIGHT	(99999.9999) LBS
8.	PACKAGE NEW	(99999.9999) LBS
9.	MUNITION COMPATIBILITY GROUP	(A)
10.	MUNITION CLASS/DIVISION	(9.9)
11.	MUNITION CATEGORY	(99) used for class 1.2 only

FORMAT (I4,A18,3F5.1,I4,2F10.4,A1,A3,I2)

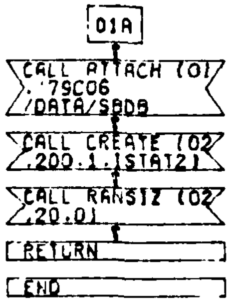
SBDBUP


01A
CALL START (SW1)
IF (SW1.NE.0)
T
F
CALL OPEN
CALL INITIAL
CALL ACTION
01B
STOP
END

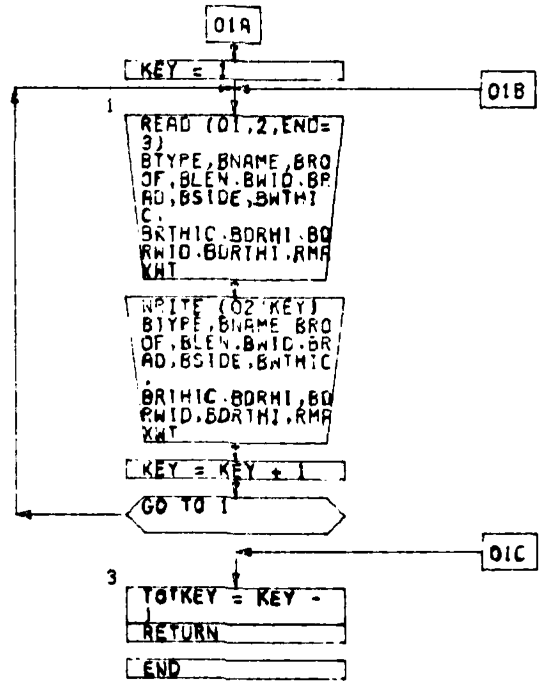
SUBROUTINE START



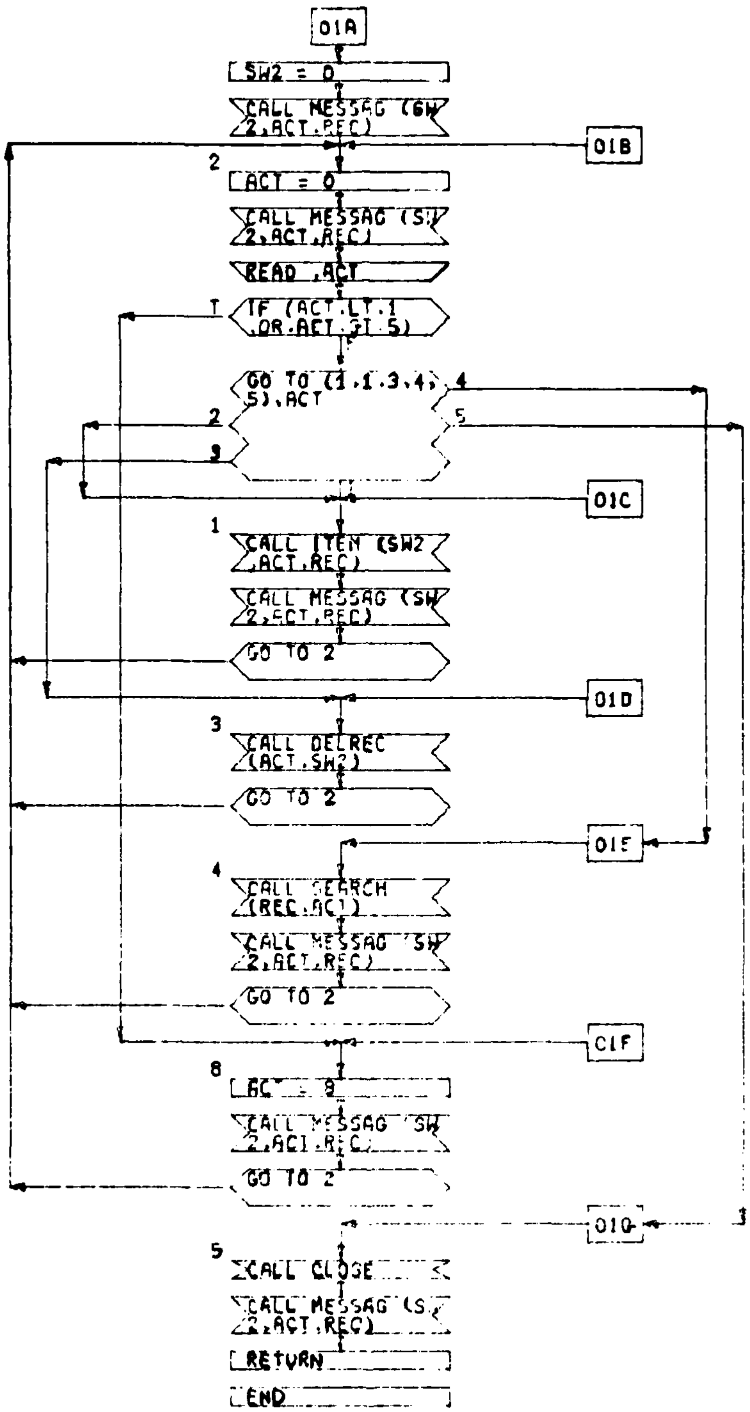
SUBROUTINE OPEN



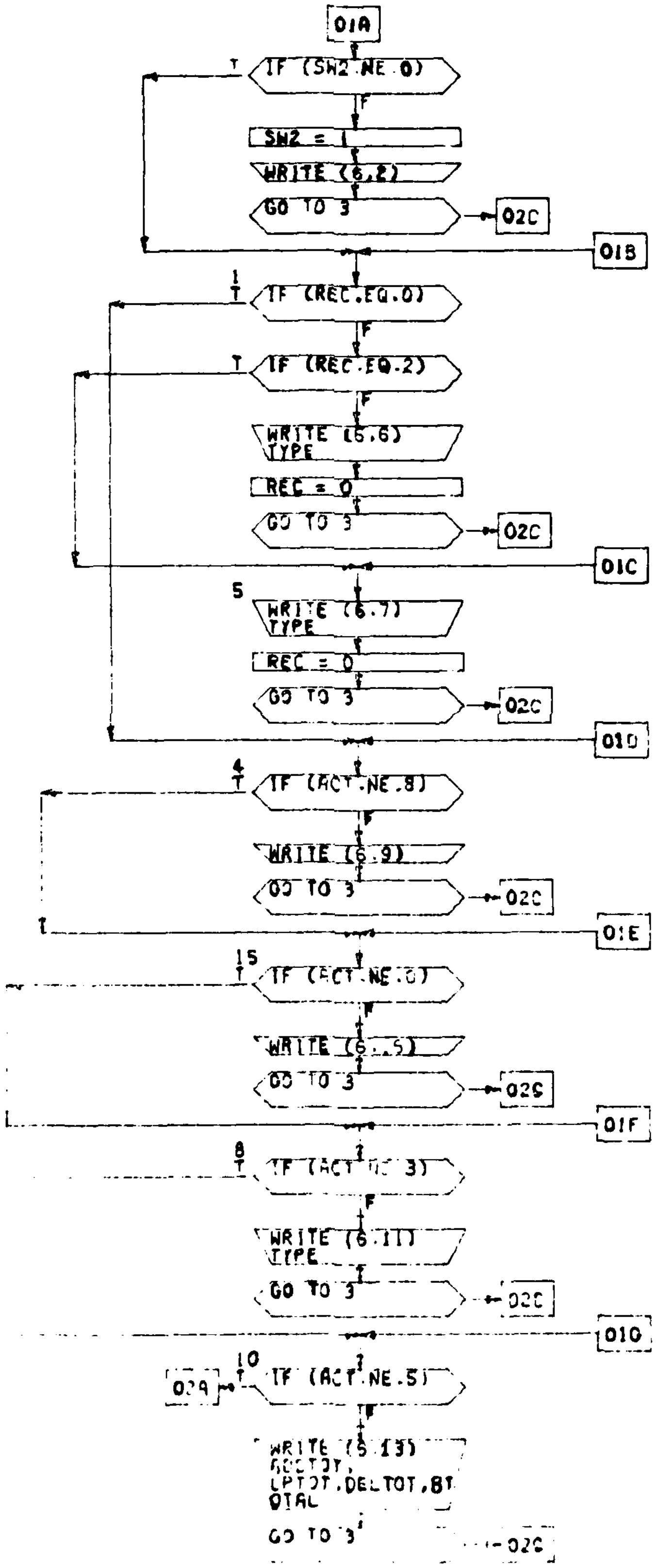
SUBROUTINE INITIAL



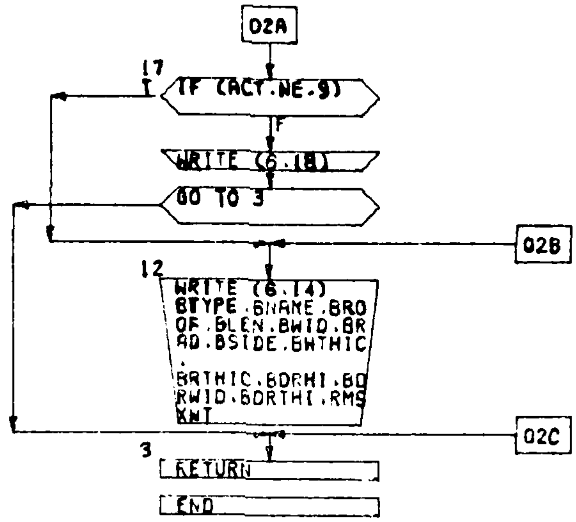
SUBROUTINE ACTION



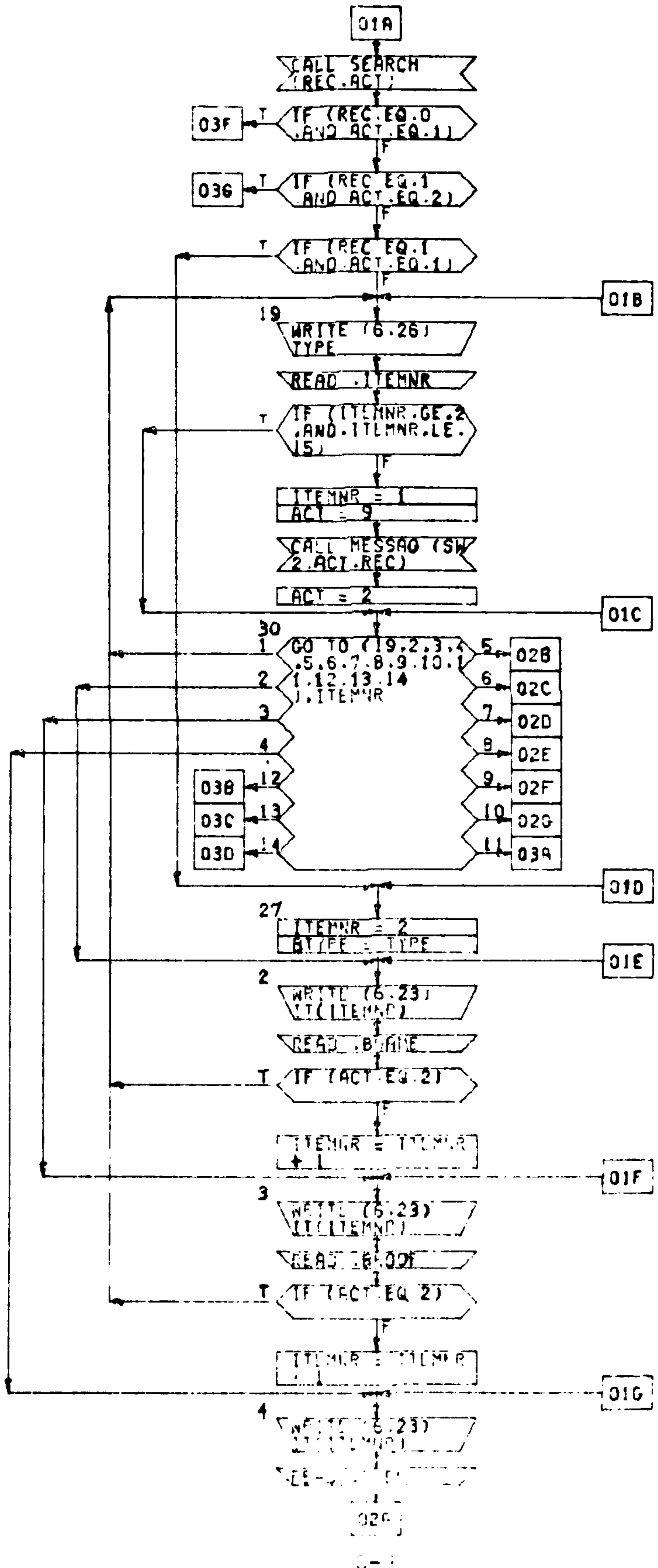
-INE MESSAG


01A
IF (SW2.NE.0)
SW2 = 1
WRITE (6,2)
GO TO 3
02C
01B
IF (REC.EQ.0)
IF (REC.EQ.2)
WRITE (6,6) TYPE
REC = 0
GO TO 3
02C
01C
5
WRITE (6,7) TYPE
REC = 0
GO TO 3
02C
01D
4
IF (ACT.NE.8)
WRITE (6,9)
GO TO 3
02C
01E
15
WRITE (6..5)
GO TO 3
02C
01F
8
WRITE (6.11) TYPE
GO TO 3
02C
01G
10
IF (ACT.NE.5)
WRITE (6.13)
DELTOT,BT
GO TO 3
02C

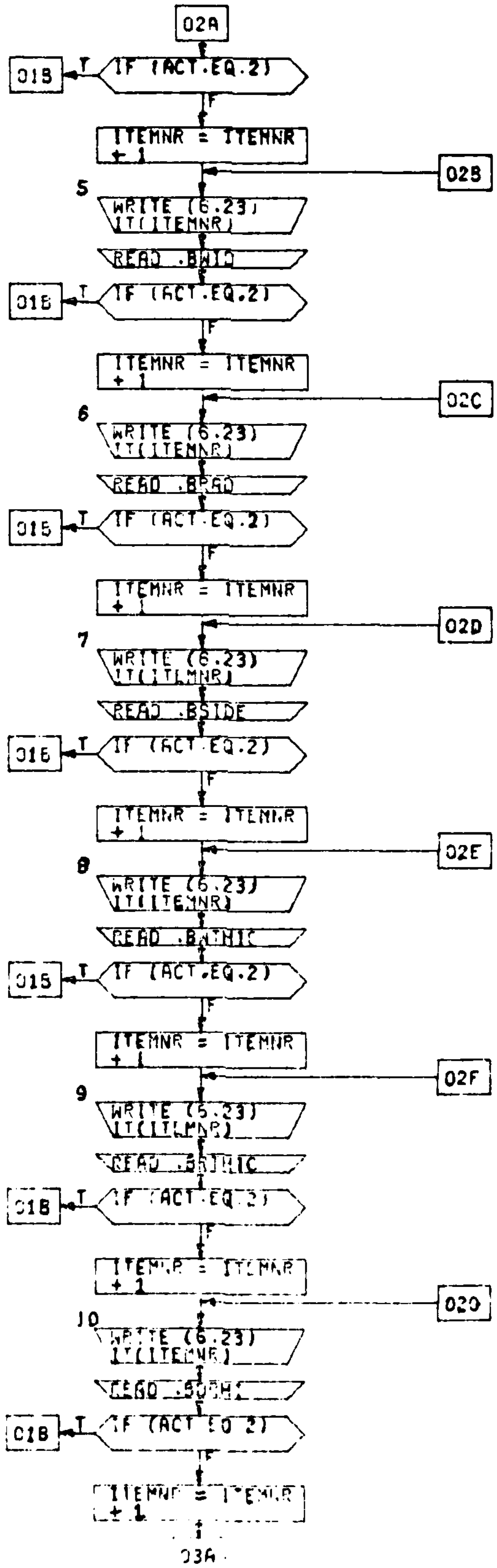
SUBROUTINE MESSAG



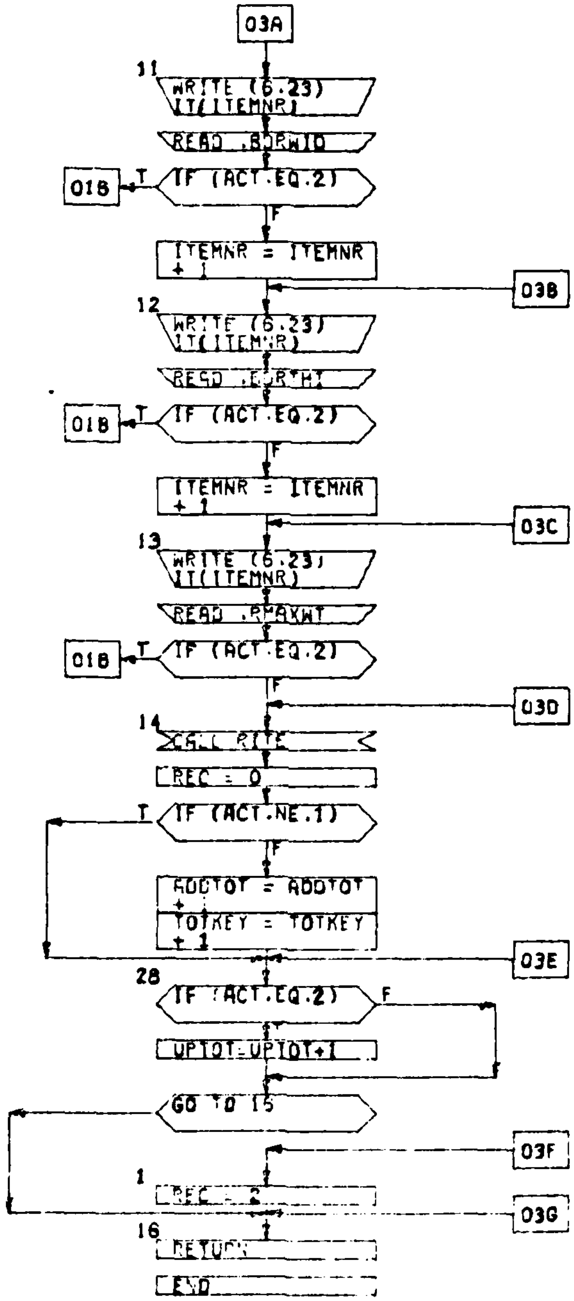
SUBROUTINE ITEM



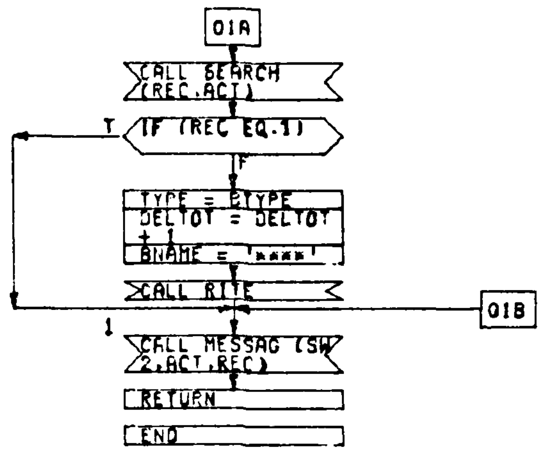
SUBROUTINE ITEM



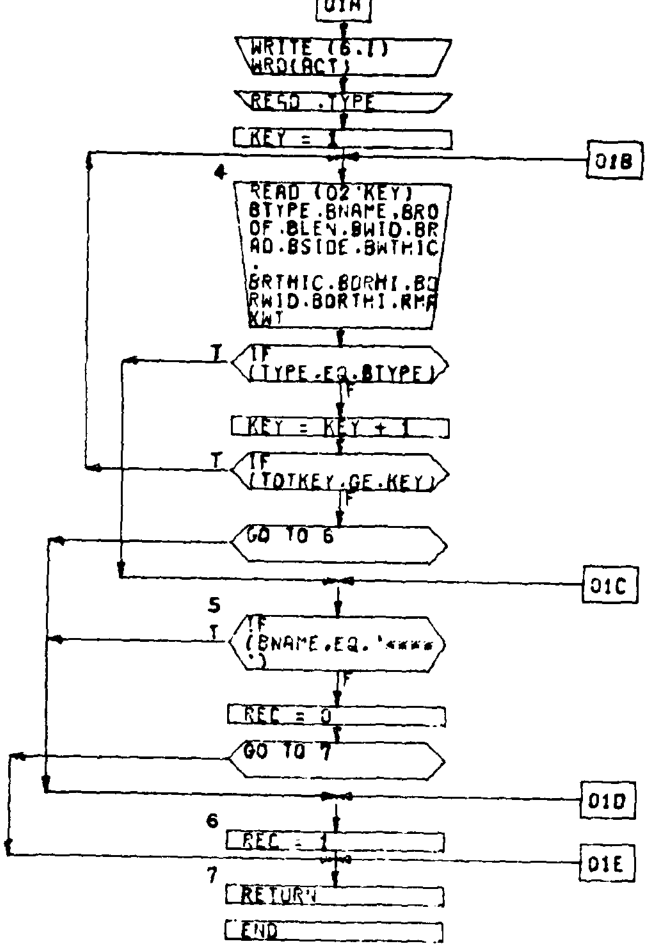
SUBROUTINE ITEM



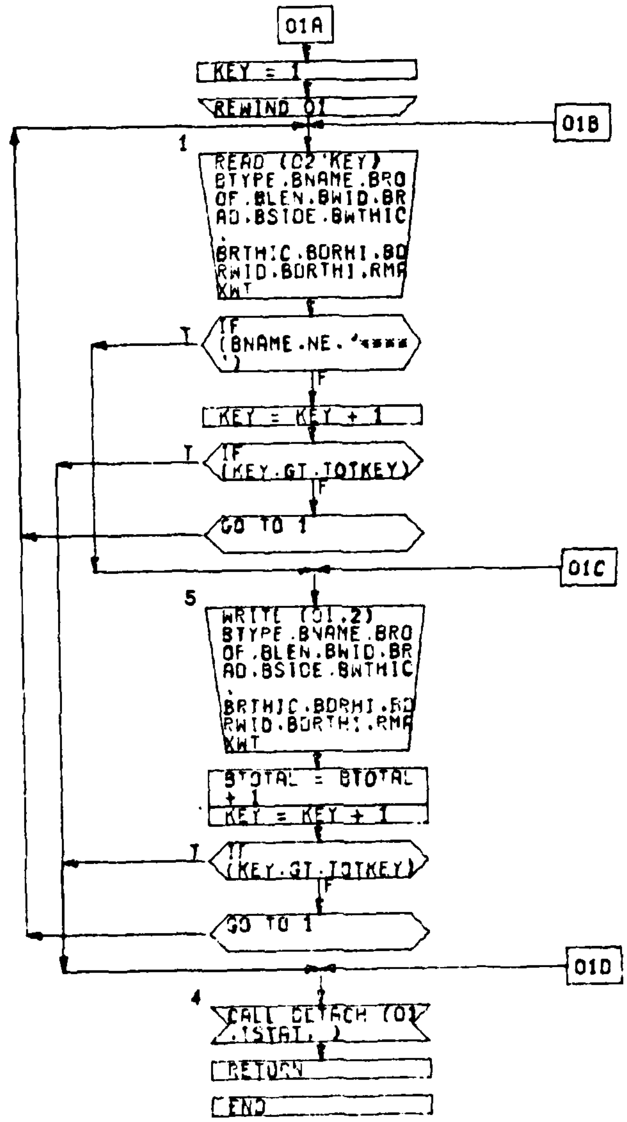
SUBROUTINE DELREC



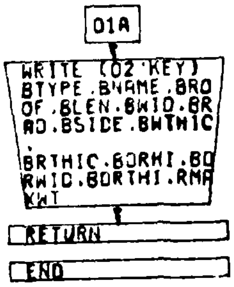
SUBROUTINE SEARCH



SUBROUTINE CLOSE



SUBROUTINE RITE



Appendix D

SBDB Utility Program Source Listing

```
*#RUN * = /OBJECT/SBDBUP (NOGO)
C       STANDARD BUILDING DATA BASE UTILITY PROGRAM                 23 NOV 79
***************************************************************************
*                                                                         *
*           SBDBUP MAIN                                                   *
*                                                                         *
***************************************************************************
*                                                                         *
*************                  PROGRAM IDENTIFICATION         *************
*                                                                         *
*         SBDBUP IS RESPONSIBLE FOR CREATING AND MAINTAINING RECORDS
*         IN THE STANDARD BUILDING DATA BASE
*                                                                         *
***************************************************************************
*                                                                         *
C               ENTER A ONE DIGIT VALUE AND HIT CARRIAGE RETURN
C                   1 - ADD
C                   2 - CHANGE
C                   3 - DELETE
C                   4 - DISPLAY
C                   5 - TERMINATE
*                                                                         *
C               IF "1" (ADD RECORD)
C                 FILL IN APPROPRIATE DATA ITEMS AS THEY ARE PRESENTED
C                           AND HIT THE CARRIAGE RETURN
*                                                                         *
C               IF "2" (CHANGE RECORD)
C                 ENTER TWO DIGIT BUILDING TYPE OF RECORD TO BE CHANGED
C                     AND HIT THE CARRIAGE RETURN
*                                                                         *
C                 THEN ENTER TWO DIGIT ITEM NUMBER TO BE CHANGED
C                       AND HIT THE CARRIAGE RETURN
*                                                                         *
C                    ITEM  ITEM                                 INPUT FORMAT
C                     NR
C                     02 - NAME                                 AAAAAA
C                     03 - ROOF (ROUND=RND, FLAT=FLT)           AAA
C                     04 - LENGTH (IN FEET)                     999.99
C                     05 - WIDTH (IN FEET)                      999.99
C                     06 - RADIUS (IN FEET)                     999.99
C                     07 - SIDE WALL HEIGHT (IN FEET)           99.99
C                     08 - WALL THICKNESS (IN FEET)             9.99
C                     09 - ROOF THICKNESS (IN FEET)             9.99
C                     10 - ENTRANCE HEIGHT (IN FEET)            99.99
C                     11 - ENTRANCE WIDTH (IN FEET)             99.99
C                     12 - DOOR THICKNESS (IN FEET)             9.99
C                     13 - MAXIMUM ALLOWABLE WEIGHT (IN TONS)   9999.99
*                                                                         *
C                     14 - FINISHED CURRENT TRANSACTION
```

```
*                                                                          *
C           IF "3" (DELETE RECORD)
C             ENTER TWO DIGIT "99" BUILDING TYPE OF RECORD TO BE DELETED
C                  AND HIT THE CARRIAGE RETURN
*                                                                          *
C           IF "4" (DISPLAY RECORD)
C             ENTER TWO DIGIT "99" BUILDING TYPE OF RECORD TO BE DISPLAYED
C                  AND HIT THE CARRIAGE RETURN
*                                                                          *
C           IF "5" TERMINATE THE EXECUTION OF THIS PROGRAM
*                                                                          *
****************************************************************************
*                                                                          *
*************              VARIABLE IDENTIFICATION            *************
*                                                                          *
*     SW1 - INDICATOR SWITCH FOR PROPER MODE OF OPERATION (ASCII)
*                                                                          *
****************                SUBROUTINE NAMES           *****************
*                                                                          *
*     CALLED BY:  NONE
*                                                                          *
*     CALLS :
*                 START - CHECKS MODE OF OPERATION
*                 OPEN - OPENS NECESSARY FILES
*                 INITIAL - INITIALIZES THE WORK FILE
*                 ACTION - PERFORMS APPROPRIATE ACTIONS ON DATA BASE
*                                                                          *
****************************************************************************
*                                                                          *
C
C
C
C
      COMMON /PT1/BTYPE,BNAME,BROOF,BLEN,BWID,BRAD,BSIDE,BWTHIC,
     &  BRTHIC,BDRHI,BDRWID,BDRTHI,RMAXWT
      COMMON /PT2/UPTOT,DELTOT,ADDTOT,BTOTAL
      COMMON /PT3/TYPE,TOTKEY,KEY
      CHARACTER BNAME*6,BROOF*3
      INTEGER TYPE*2,BTYPE*2
      INTEGER SW1,UPTOT,DELTOT,ADDTOT,BTOTAL,TOTKEY,KEY
      CALL START (SW1)
      IF (SW1.NE.0) GO TO 1
      CALL OPEN
      CALL INITIAL
      CALL ACTION
    1 STOP
      END
*                                                                          *
****************                  END MAIN                 *****************
```

```
*************************************************************************
*                                                                       *
      SUBROUTINE START (SW1)
*                                                                       *
*************************************************************************
*                                                                       *
*************             PROGRAM IDENTIFICATION             *************
*                                                                       *
*      THIS ROUTINE VERIFIES PROPER MODE OF OPERATION (ASCII)
*                                                                       *
*************************************************************************
*                                                                       *
*************************************************************************
*                                                                       *
*************             VARIABLE IDENTIFICATION            *************
*                                                                       *
*     MODE(2) - SYSTEM VARIABLE; 0 - BCD, 1 - ASCII
*     SW1 - INDICATOR SWITCH FOR PROPER MODE OF OPERATION (ASCII)
*                                                                       *
****************               SUBROUTINE NAMES             ******************
*                                                                       *
*     CALLED BY:  MAIN
*                                                                       *
*     CALLS:  NONE
*                                                                       *
*************************************************************************
*                                                                       *
        INTEGER SW1*1
        IF (MODE(2).NE.0) GO TO 1
        SW1 = 1
        WRITE (6,2)
*                                                                       *
    2   FORMAT (5X,"PLEASE RESTART USING 'RUN'")
*                                                                       *
    1 RETURN
      END
*                                                                       *
****************                  END START                 ******************
```

```
139***********************************************************************
140*                                                                     *
141      SUBROUTINE OPEN
142*                                                                     *
143***********************************************************************
144*                                                                     *
145*************             PROGRAM IDENTIFICATION          *************
146*                                                                     *
147*      THIS ROUTINE IS USED TO OPEN EXTERNAL FILES FOR PROGRAM
148*      CONTROL
149*                                                                     *
150***********************************************************************
151*                                                                     *
152***********************************************************************
153*                                                                     *
154*************             VARIABLE IDENTIFICATION         *************
155*                                                                     *
156*      ISTAT1 - FILE STATUS VARIABLE FOR PERMANENT FILE
157*      ISTAT2 - FILE STATUS VARIABLE FOR WORK FILE
158*                                                                     *
159****************              SUBROUTINE NAMES         ******************
160*                                                                     *
161*      CALLED BY:  MAIN
162*                                                                     *
163*      CALLS:
164*              ATTACH - OPENS PERMANENT FILE
165*              CREATE - CREATES TEMPORARY WORK FILE
166*              RANSIZ - INITIALIZES THE WORK FILE AS RANDOM
167*                       AND SPECIFIES RECORD LENGTH
168*                                                                     *
169***********************************************************************
170*                                                                     *
171***    CARD 173 WILL HAVE TO BE CHANGED FOR NEW USERS                ***
172*                                                                     *
173       CALL ATTACH (01,"79C06/DATA/SBDB;",3,0,ISTAT1, )
174       CALL CREATE (02,200,1,ISTAT2)
175       CALL RANSIZ (02,20,0)
176      RETURN
177      END
178*                                                                     *
179****************                  END OPEN             ******************
```

```
***************************************************************************
*                                                                         *
      SUBROUTINE INITIAL
*                                                                         *
***************************************************************************
*                                                                         *
*************             PROGRAM IDENTIFICATION              *************
*                                                                         *
*      THIS ROUTINE OPENS PERMANENT DATA BASE FILE ON TO THE
*      TEMPORARY WORK FILE
*                                                                         *
*************             VARIABLE IDENTIFICATION             *************
*                                                                         *
*      BDRHI - BUILDING DOOR HEIGHT
*      BDRTHI - BUILDING DOOR THICKNESS
*      BDRWID - BUILDING DOOR WIDTH
*      BLEN - BUILDING LENGTH
*      BNAME - BUILDING NAME
*      BRAD - BUILDING ROOF RADIUS
*      BROOF - BUILDING ROOF TYPE
*      BRTHIC - BUILDING ROOF THICKNESS
*      BSIDE - BUILDING SIDE WALL HEIGHT
*      BTYPE - STANDARD BUILDING TYPE
*      BWID - BUILDING WIDTH
*      BWTHIC - BUILDING WALL THICKNESS
*      KEY - INDEX KEY FOR STANDARD BUILDING WORK FILE
*      RMAXWT - BUILDING MAXIMUM STORAGE WEIGHT
*      TOTKEY - NUMBER OF RECORDS IN DATA BASE
*                                                                         *
****************              SUBROUTINE NAMES              ******************
*                                                                         *
*      CALLED BY:  MAIN
*                                                                         *
*      CALLS:  NONE
*                                                                         *
***************************************************************************
*                                                                         *
       COMMON /PT1/BTYPE,BNAME,BROOF,BLEN,BWID,BRAD,BSIDE,BWTHIC,
     &   BRTHIC,BDRHI,BDRWID,BDRTHI,RMAXWT
       COMMON /PT3/TYPE,TOTKEY,KEY
       INTEGER BTYPE*2
       INTEGER KEY,TOTKEY
       CHARACTER BNAME*6,BROOF*3
       KEY = 1
   1   READ (01,2,END=3) BTYPE,BNAME,BROOF,BLEN,BWID,BRAD,BSIDE,BWTHIC,
     &   BRTHIC,BDRHI,BDRWID,BDRTHI,RMAXWT
       WRITE (02'KEY) BTYPE,BNAME,BROOF,BLEN,BWID,BRAD,BSIDE,BWTHIC,
     &   BRTHIC,BDRHI,BDRWID,BDRTHI,RMAXWT
       KEY = KEY + 1
       GO TO 1
   3   TOTKEY = KEY - 1
*                                                                         *
   2   FORMAT (I2,A6,A3,3F6.2,F5.2,2F4.2,2F5.2,F4.2.F7.2)
*                                                                         *
       RETURN
       END
*                                                                         *
****************                END INITIAL                ******************
```

```
************************************************************************
*                                                                      *
      SUBROUTINE ACTION
*                                                                      *
************************************************************************
*                                                                      *
*************              PROGRAM IDENTIFICATION             *************
*                                                                      *
*      DRIVER ROUTINE THAT SELECTS THE APPROPRIATE ACTION TO BE
*      ACCOMPLISHED
*                                                                      *
************************************************************************
*                                                                      *
*************              VARIABLE IDENTIFICATION            *************
*                                                                      *
*      ACT - TYPE OF ACTION (VALUE - 1 TO 5)
*      REC - CONTROL SWITCH:  1 - FOUND, 0 - NOT FOUND
*      SW2 - FIRST PASS SWITCH:  0 - FIRST PASS, 1 - SUBSEQUENT PASSES
*                                                                      *
****************               SUBROUTINE NAMES          ******************
*                                                                      *
*      CALLED BY:  MAIN
*                                                                      *
*      CALLS:
*              MESSAG - PRINTS APPROPRIATE MESSAGE
*              ITEM - ADDS A NEW RECORD OR CHANGES AN EXISTING RECORD
*              DELREC - DELETES SPECIFIED RECORD
*              SEARCH - SEARCHES FOR SPECIFIED RECORD
*              CLOSE - TERMINATES THE PROGRAM
*                                                                      *
************************************************************************
*                                                                      *
       INTEGER ACT*1,SW2*1,REC*1
       SW2 = 0
       CALL MESSAG (SW2,ACT,REC)
 2     ACT = 0
       CALL MESSAG (SW2,ACT,REC)
       READ ,ACT
       IF (ACT.LT.1.OR.ACT.GT.5) GO TO 8
       GO TO (1,1,3,4,5),ACT
 1     CALL ITEM (SW2,ACT,REC)
       CALL MESSAG (SW2,ACT,REC)
       GO TO 2
 3     CALL DELREC (ACT,SW2)
       GO TO 2
 4     CALL SEARCH (REC,ACT)
       CALL MESSAG (SW2,ACT,REC)
       GO TO 2
 8     ACT = 8
       CALL MESSAG (SW2,ACT,REC)
       GO TO 2
 5     CALL CLOSE
       CALL MESSAG (SW2,ACT,REC)
     RETURN
     END
*                                                                      *
****************                  END ACTION             ******************
```

```
*****************************************************************************
*                                                                           *
      SUBROUTINE MESSAG (SW2,ACT,REC)
*                                                                           *
*****************************************************************************
*                                                                           *
*************             PROGRAM IDENTIFICATION                *************
*                                                                           *
*      PRINTS THE APPROPRIATE MESSAGES
*                                                                           *
*****************************************************************************
*                                                                           *
*****************************************************************************
*                                                                           *
*************             VARIABLE IDENTIFICATION               *************
*                                                                           *
*      ACT - TYPE OF ACTION BEING PERFORMED
*      ADDTOT - NUMBER OF RECORDS ADDED
*      BDRHI - BUILDING DOOR HEIGHT
*      BDRTHI - BUILDING DOOR THICKNESS
*      BDRWID - BUILDING DOOR WIDTH
*      BLEN - BUILDING LENGTH
*      BNAME - BUILDING NAME
*      BRAD - BUILDING ROOF RADIUS
*      BROOF - BUILDING ROOF TYPE
*      BRTHIC - BUILDING ROOF THICKNESS
*      BSIDE - BUILDING SIDE WALL HEIGHT
*      BTOTAL - NUMBER OF RECORDS IN DATA BASE
*      BTYPE - STANDARD BUILDING TYPE
*      BWID - BUILDING WIDTH
*      BWTHIC - BUILDING WALL THICKNESS
*      DELTOT - NUMBER OF RECORDS DELETED
*      REC - FOUND/NOT FOUND SWITCH
*      RMAXWT - BUILDING MAXIMUM STORAGE WEIGHT
*      SW2 - FIRST PASS SWITCH:  0 - FIRST PASS, 1 - SUBSEQUENT PASSES
*      TYPE - STANDARD BUILDING TYPE
*      UPTOT - NUMBER OF RECORDS UPDATED
*                                                                           *
****************                SUBROUTINE NAMES               *****************
*                                                                           *
*      CALLED BY:
*                   ACTION
*                   ITEM
*                   DELREC
*                                                                           *
*      CALLS:  NONE
*                                                                           *
*****************************************************************************
*                                                                           *
       COMMON /PT1/BTYPE,BNAME,BROOF,BLEN,BWID,BRAD,BSIDE,BWTHIC,
     &   BRTHIC,BDRHI,BDRWID,BDRTHI,RMAXWT
       COMMON /PT2/UPTOT,DELTOT,ADDTOT,BTOTAL
       COMMON /PT3/TYPE,TOTKEY,KEY
       CHARACTER BNAME*6,BROOF*3
       INTEGER TYPE*2,TOTKEY,KEY,BTYPE*2
       INTEGER ACT*1,SW2*1,REC*1,UPTOT,DELTOT,ADDTOT,BTOTAL
       IF (SW2.NE.0) GO TO 1
       SW2 = 1
```

```
      WRITE (6,2)
      GO TO 3
    1 IF (REC.EQ.0) GO TO 4
      IF (REC.EQ.2) GO TO 5
      WRITE (6,6) TYPE
      REC = 0
      GO TO 3
    5 WRITE (6,7) TYPE
      REC = 0
      GO TO 3
    4 IF (ACT.NE.8) GO TO 15
      WRITE (6,9)
      GO TO 3
   15 IF (ACT.NE.0) GO TO 8
      WRITE (6,16)
      GO TO 3
    8 IF (ACT.NE.3) GO TO 10
      WRITE (6,11) TYPE
      GO TO 3
   10 IF (ACT.NE.5) GO TO 17
      WRITE (6,13) ADDTOT, UPTOT,DELTOT,BTOTAL
      GO TO 3
   17 IF (ACT.NE.9) GO TO 12
      WRITE (6,18)
      GO TO 3
   12 WRITE (6,14) BTYPE,BNAME,BROOF,BLEN,BWID,BRAD,BSIDE,BWTHIC,
     &   BRTHIC,BDRHI,BDRWID,BDRTHI,RMAXWT
*                                                                     *
    2 FORMAT (//5X,"WELCOME TO THE STANDARD BUILDING DATA BASE"//)
*                                                                     *
    9 FORMAT (10X,"OPTIONS:",/
     &   25X,"1 - ADD RECORD"/,25X,"2 - CHANGE RECORD"/,25X,
     &   "3 - DELETE RECORD"/,25X,"4 - DISPLAY RECORD"/,25X,
     &   "5 - TERMINATE JOB"//)
*                                                                     *
    6 FORMAT (5X,"RECORD ",I2," DOES NOT EXIST"//)
*                                                                     *
    7 FORMAT (5X,"RECORD ",I2," ALREADY EXISTS"//)
*                                                                     *
   11 FORMAT (5X,"RECORD ",I2," HAS BEEN DELETED FROM THE DATA BASE"//)
*                                                                     *
   13 FORMAT (5X,"YOU ARE NOW EXITING THE UPDATE PROGRAM"//
     &     15X,"ADDED - ",I3/,15X,"CHANGED - ",I3/,15X,"DELETED - ",
     &     I3/,15X,"TOTAL NUMBER OF RECORDS IN DATA BASE - ",I3//)
*                                                                     *
   14 FORMAT (5X,"BUILDING ID - ",I2,/,10X,"NAME - ",A6,/,10X,"ROOF - ",
     & A3,/,10X,"LENGTH - ",F6.2," FT"/,10X,"WIDTH - ",F6.2," FT"/,
     & 10X,"RADIUS - ",F6.2," FT"/,10X,"SIDE WALL HEIGHT - ",
     & F5.2," FT"/,10X,"WALL THICKNESS - ",F4.2," FT"/,10X,"ROOF ",
     &   "THICKNESS - ",F4.2," FT"/,
     & 10X,"ENTRANCE HEIGHT - ",F5.2," FT"/,10X,"ENTRANCE WIDTH - ",
     &  F5.2," FT"/,10X,"DOOR THICKNESS - ",F4.2," FT"/,
     &  10X,"MAXIMUM WEIGHT - ",F7.2," TONS"//)
*                                                                     *
   16 FORMAT (5X,"ENTER THE ONE DIGIT TRANSACTION DESIRED:"//)
*                                                                     *
   18 FORMAT (5X,"OPTIONS:",/20X "02 - NAME (6A)",/20X,
```

```
     &    "03 - ROOF (3A) 'RND' - ROUND OR 'FLT' - FLAT"/,
     &    20X,"04 - LENGTH (999.99) IN FT",/20X,
     &    "05 - WIDTH (999.99) IN FT",/20X,
     &    "06 - RADIUS (999.99) IN FT",/20X,
     &    "07 - SIDE WALL HEIGHT (99.99) IN FT",/20X,
     &    "08 - WALL THICKNESS (9.99) IN FT",/20X,
     &    "09 - ROOF THICKNESS (9.99) IN FT",/20X,
     &    "10 - ENTRANCE HEIGHT (99.99) IN FT",/20X,
     &    "11 - ENTRANCE WIDTH (99.99) IN FT",/20X,
     &    "12 - DOOR THICKNESS (9.99) IN FT",/20X,
     &    "13 - MAX WEIGHT (9999.99) IN TONS",//20X,
     &    "14 - FINISHED THIS TRANSACTION"//)
*                                                                            *
    3 RETURN
      END
*                                                                            *
****************                 END MESSAG             ******************
```

```
*************************************************************************
*                                                                       *
      SUBROUTINE ITEM (SW2,ACT,REC)
*                                                                       *
*************************************************************************
*                                                                       *
*************             PROGRAM IDENTIFICATION            *************
*                                                                       *
*      THIS ROUTINE WILL EITHER CREATE A NEW RECORD OR UPDATE
*      SPECIFIED ITEMS OF AN EXISTING RECORD
*                                                                       *
*************************************************************************
*                                                                       *
*************************************************************************
*                                                                       *
*************             VARIABLE IDENTIFICATION           *************
*                                                                       *
*      ACT - TYPE OF ACTION IN PROGRESS
*      ADDTOT - NUMBER OF RECORDS ADDED
*      BDRHI - BUILDING DOOR HEIGHT
*      BDRTHI - BUILDING DOOR THICKNESS
*      BDRWID - BUILDING DOOR WIDTH
*      BLEN - BUILDING LENGTH
*      BNAME - BUILDING NAME
*      BRAD - BUILDING ROOF RADIUS
*      BROOF - BUILDING ROOF TYPE
*      BRTHIC - BUILDING ROOF THICKNESS
*      BSIDE - BUILDING SIDE WALL HEIGHT
*      BTYPE - STANDARD BUILDING TYPE
*      BWID - BUILDING WIDTH
*      BWTHIC - BUILDING WALL THICKNESS
*      IT - ARRAY CONTAINING ITEM NAMES TO BE PROCESSED
*      ITEMNR - INDEX FOR IT ARRAY
*      RMAXWT - BUILDING MAXIMUM STORAGE WEIGHT
*      TOTKEY - NUMBER OF RECORDS IN DATA BASE
*      UPTOT - NUMBER OF RECORDS UPDATED
*                                                                       *
****************               SUBROUTINE NAMES          ****************
*                                                                       *
*      CALLED BY:  ACTION
*                                                                       *
*      CALLS:
*              SEARCH - SEARCHES FOR SPECIFIED RECORD
*              MESSAG - PRINTS SPECIFIED MESSAGE
*              RITE - WRITES SPECIFIED RECORD
*                                                                       *
*************************************************************************
*                                                                       *
       COMMON /PT1/BTYPE,BNAME,BROOF,BLEN,BWID,BRAD,BSIDE,BWTHIC,
     &   BRTHIC,BDRHI,BDRWID,BDRTHI,RMAXWT
       COMMON /PT2/UPTOT,DELTOT,ADDTOT,BTOTAL
       COMMON /PT3/TYPE,TOTKEY,KEY
       INTEGER TYPE*2,SW2*1,BTYPE*2
       INTEGER ACT*1,ITEMNR*2,REC*1,ADDTOT,UPTOT,TOTKEY,KEY
       CHARACTER BNAME*6,BROOF*3
       CHARACTER IT*40(13)/"BUILDING TYPE (99)",
     &   "BUILDING NAME (XXXXXX)",
     &   "ROOF 'RND' - IGLOO OR 'FLT' - REGULAR",
```

```
     &    "BUILDING LENGTH (999.99) IN FT",
     &    "BUILDING WIDTH (999.99) IN FT",
     &    "RADIUS OF ROOF (999.99) IN FT",
     &    "SIDE WALL HEIGHT (99.99) IN FT"
     &    "WALL THICKNESS (9.99) IN FT",
     &    "ROOF THICKNESS (9.99) IN FT",
     &    "DOOR HEIGHT (99.99) IN FT",
     &    "DOOR WIDTH (99.99) IN FT",
     &    "DOOR THICKNESS (9.99) IN FT",
     &    "MAX WEIGHT (9999.99) IN TONS"/
      CALL SEARCH (REC,ACT)
      IF (REC.EQ.0.AND.ACT.EQ.1) GO TO 1
      IF (REC.EQ.1.AND.ACT.EQ.2) GO TO 16
      IF (REC.EQ.1.AND.ACT.EQ.1) GO TO 27
 19   WRITE (6,26) TYPE
      READ ,ITEMNR
      IF (ITEMNR.GE.2.AND.ITEMNR.LE.15) GO TO 30
      ITEMNR = 1
      ACT = 9
      CALL MESSAG (SW2,ACT,REC)
      ACT = 2
 30   GO TO (19,2,3,4,5,6,7,8,9,10,11,12,13,14),ITEMNR
 27   ITEMNR = 2
      BTYPE = TYPE
  2   WRITE (6,23) IT(ITEMNR)
      READ ,BNAME
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
  3   WRITE (6,23) IT(ITEMNR)
      READ ,BROOF
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
  4   WRITE (6,23) IT(ITEMNR)
      READ ,BLEN
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
  5   WRITE (6,23) IT(ITEMNR)
      READ ,BWID
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
  6   WRITE (6,23) IT(ITEMNR)
      READ ,BRAD
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
  7   WRITE (6,23) IT(ITEMNR)
      READ ,BSIDE
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
  8   WRITE (6,23) IT(ITEMNR)
      READ ,BWTHIC
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
  9   WRITE (6,23) IT(ITEMNR)
      READ ,BRTHIC
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
 10   WRITE (6,23) IT(ITEMNR)
      READ ,BDRHI
```

```
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
 11   WRITE (6,23) IT(ITEMNR)
      READ ,BDRWID
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
 12   WRITE (6,23) IT(ITEMNR)
      READ ,BDRTHI
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
 13   WRITE (6,23) IT(ITEMNR)
      READ ,RMAXWT
      IF (ACT.EQ.2) GO TO 19
 14   CALL RITE
      REC = 0
      IF (ACT.NE.1) GO TO 28
      ADDTOT = ADDTOT + 1
      TOTKEY = TOTKEY + 1
 28   IF (ACT.EQ.2) UPTOT=UPTOT+1
      GO TO 16
  1   REC = 2
*                                                                        *
 23   FORMAT (5X,"ENTER THE INFORMATION FOR THE ",A40)
*                                                                        *
 26   FORMAT (5X,"ENTER THE 2 DIGIT ITEM NUMBER OF THE ITEM TO BE ",
    &     /10X,"CHANGED FOR BUILDING TYPE ",I2//)
*                                                                        *
 16 RETURN
    END
*                                                                        *
****************                 END ITEM                *****************
```

```
************************************************************************
*                                                                      *
      SUBROUTINE DELREC (ACT,SW2)
*                                                                      *
************************************************************************
*                                                                      *
*************              PROGRAM IDENTIFICATION          *************
*                                                                      *
*       THIS ROUTINE WILL DELETE THE SPECIFIED EXISTING RECORD FROM
*       THE DATA BASE
*                                                                      *
************************************************************************
*                                                                      *
************************************************************************
*                                                                      *
*************              VARIABLE IDENTIFICATION         *************
*                                                                      *
*       ACT - TYPE OF ACTION IN PROGRESS
*       BNAME - BUILDING NAME
*       BTYPE - STANDARD BUILDING TYPE
*       DELTOT - NUMBER OF RECORDS DELETED
*       REC - FOUND/NOT FOUND SWITCH
*       SW2 - FIRST PASS SWITCH: 0 - FIRST PASS, 1 - SUBSEQUENT PASSES
*       TYPE - STANDARD BUILDING TYPE
*                                                                      *
****************              SUBROUTINE NAMES          ****************
*                                                                      *
*       CALLED BY:  ACTION
*                                                                      *
*       CALLS:
*               SEARCH - SEARCHES FOR SPECIFIED RECORD
*               RITE - WRITES DELETED IDENTIFIER FOR SPECIFIED RECORD
*               MESSAG - PRINTS SPECIFIED MESSAGE
*                                                                      *
************************************************************************
*                                                                      *
        COMMON /PT1/BTYPE,BNAME,BROOF,BLEN,BWID,BRAD,BSIDE,BWTHIC,
     &    BRTHIC,BDRHI,BDRWID,BDRTHI,RMAXWT
        COMMON /PT2/UPTOT,DELTOT,ADDTOT,BTOTAL
        COMMON /PT3/TYPE,TOTKEY,KEY
        INTEGER TYPE*2,TOTKEY,KEY,BTYPE*2
        CHARACTER BNAME*6,BROOF*3
        INTEGER SW2*1,ACT*1,REC*1,DELTOT
        CALL SEARCH (REC,ACT)
        IF (REC.EQ.1) GO TO 1
        TYPE = BTYPE
        DELTOT = DELTOT + 1
        BNAME = "****"
        CALL RITE
    1   CALL MESSAG (SW2,ACT,REC)
      RETURN
      END
*                                                                      *
****************                 END DELREC             ****************
```

```
****************************************************************************
*                                                                          *
      SUBROUTINE SEARCH (REC,ACT)
*                                                                          *
****************************************************************************
*                                                                          *
*************              PROGRAM IDENTIFICATION              *************
*                                                                          *
*       THIS ROUTINE SEARCHES FOR THE SPECIFIED RECORD
*                                                                          *
****************************************************************************
*                                                                          *
****************************************************************************
*                                                                          *
*************              VARIABLE IDENTIFICATION             *************
*                                                                          *
*       ACT - TYPE OF ACTION IN PROGRESS
*       BDRHI - BUILDING DOOR HEIGHT
*       BDRTHI - BUILDING DOOR THICKNESS
*       BDRWID - BUILDING DOOR WIDTH
*       BLEN - BUILDING LENGTH
*       BNAME - BUILDING NAME
*       BRAD - BUILDING ROOF RADIUS
*       BROOF - BUILDING ROOF TYPE
*       BRTHIC - BUILDING ROOF THICKNESS
*       BSIDE - BUILDING SIDE WALL HEIGHT
*       BTYPE - STANDARD BUILDING TYPE
*       BWID - BUILDING WIDTH
*       BWTHIC - BUILDING WALL THICKNESS
*       KEY - INDEX KEY FOR STANDARD BUILDING WORK FILE
*       REC - FOUND/NOT FOUND SWITCH
*       RMAXWT - BUILDING MAXIMUM STORAGE WEIGHT
*       TOTKEY - NUMBER OF RECORDS IN DATA BASE
*       TYPE - STANDARD BUILDING TYPE
*       WRD - ARRAY CONTAINING TYPE OF ACTION IN PROGRESS
*                                                                          *
****************              SUBROUTINE NAMES            ******************
*                                                                          *
*       CALLED BY:
*                   ACTION
*                   ITEM
*                   DELREC
*                                                                          *
*       CALLS:  NONE
*                                                                          *
****************************************************************************
*                                                                          *
        COMMON /PT1/BTYPE,BNAME,BROOF,BLEN,BWID,BRAD,BSIDE,BWTHIC,
     &    BRTHIC,BDRHI,BDRWID,BDRTHI,RMAXWT
        COMMON /PT3/TYPE,TOTKEY,KEY
        INTEGER REC*1,ACT*1,BTYPE*2,TOTKEY,KEY,TYPE*2
        CHARACTER WRD*9(4)/"ADDED ","CHANGED","DELETED","DISPLAYED"/,
     &    BNAME*6,BROOF*3
        WRITE (6,1) WRD(ACT)
        READ ,TYPE
        KEY = 1
    4   READ (02'KEY) BTYPE,BNAME,BROOF,BLEN,BWID,BRAD,BSIDE,BWTHIC,
     &    BRTHIC,BDRHI,BDRWID,BDRTHI,RMAXWT
```

```
      IF (TYPE.EQ.BTYPE) GO TO 5
      KEY = KEY + 1
      IF (TOTKEY.GE.KEY) GO TO 4
      GO TO 6
 5    IF (BNAME.EQ."****") GO TO 6
      REC = 0
      GO TO 7
*                                                                     *
 1    FORMAT (5X,"ENTER THE TWO DIGIT BUILDING TYPE OF RECORD TO BE ",A9)
*                                                                     *
 6    REC = 1
 7  RETURN
    END
*                                                                     *
****************                 END SEARCH          ******************
```

```
************************************************************************
*                                                                      *
      SUBROUTINE CLOSE
*                                                                      *
************************************************************************
*                                                                      *
*************            PROGRAM IDENTIFICATION             *************
*                                                                      *
*     THIS ROUTINE WRITES THE UPDATED DATA BASE BACK TO THE
*     PERMANENT FILE
*                                                                      *
************************************************************************
*                                                                      *
************************************************************************
*                                                                      *
*************            VARIABLE IDENTIFICATION            *************
*                                                                      *
*     BDRHI - BUILDING DOOR HEIGHT
*     BDRTHI - BUILDING DOOR THICKNESS
*     BDRWID - BUILDING DOOR WIDTH
*     BLEN - BUILDING LENGTH
*     BNAME - BUILDING NAME
*     BRAD - BUILDING ROOF RADIUS
*     BROOF - BUILDING ROOF TYPE
*     BRTHIC - BUILDING ROOF THICKNESS
*     BSIDE - BUILDING SIDE WALL HEIGHT
*     BTOTAL - NUMBER OF RECORDS IN DATA BASE
*     BTYPE - STANDARD BUILDING TYPE
*     BWID - BUILDING WIDTH
*     BWTHIC - BUILDING WALL THICKNESS
*     KEY - INDEX KEY FOR STANDARD BUILDING WORK FILE
*     RMAXWT - BUILDING MAXIMUM STORAGE WEIGHT
*     TOTKEY - NUMBER OF RECORDS IN DATA BASE
*                                                                      *
****************             SUBROUTINE NAMES            ****************
*                                                                      *
*     CALLED BY:  ACTION
*                                                                      *
*     CALLS:  DETACH - RELEASES SPECIFIED FILE FROM PROGRAM CONTROL
*                                                                      *
************************************************************************
*                                                                      *
      COMMON /PT1/BTYPE,BNAME,BROOF,BLEN,BWID,BRAD,BSIDE,BWTHIC,
     &  BRTHIC,BDRHI,BDRWID,BDRTHI,RMAXWT
      COMMON /PT2/UPTOT,DELTOT,ADDTOT,BTOTAL
      COMMON /PT3/TYPE,TOTKEY,KEY
      INTEGER BTOTAL,TYPE,BTYPE*2,TOTKEY,KEY
      CHARACTER BNAME*6,BROOF*3
      KEY = 1
      REWIND 01
    1 READ (02'KEY) BTYPE,BNAME,BROOF,BLEN,BWID,BRAD,BSIDE,BWTHIC,
     &  BRTHIC,BDRHI,BDRWID,BDRTHI,RMAXWT
      IF (BNAME.NE."****") GO TO 5
      KEY = KEY + 1
      IF (KEY.GT.TOTKEY) GO TO 4
      GO TO 1
    5 WRITE (01,2) BTYPE,BNAME,BROOF,BLEN,BWID,BRAD,BSIDE,BWTHIC,
     &  BRTHIC,BDRHI,BDRWID,BDRTHI,RMAXWT
```

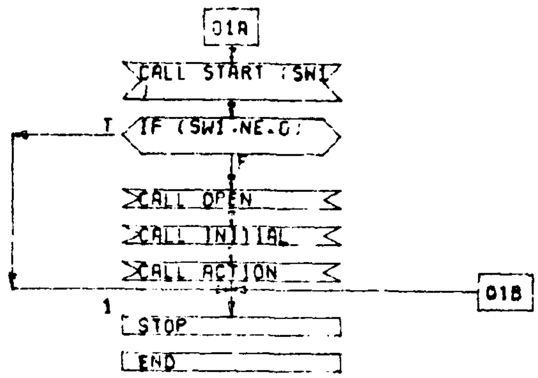
```
      BTOTAL = BTOTAL + 1
      KEY = KEY + 1
      IF (KEY.GT.TOTKEY) GO TO 4
      GO TO 1
*                                                                      *
    2 FORMAT (I2,A6,A3,3F6.2,F5.2,2F4.2,2F5.2,F4.2,F7.2)
*                                                                      *
    4 CALL DETACH (01,ISTAT, )
      CALL DETACH (02,ISTAT2, )
    RETURN
    END
*                                                                      *
****************                 END CLOSE           ******************
```

```
***************************************************************************
*                                                                         *
      SUBROUTINE RITE
*                                                                         *
***************************************************************************
*                                                                         *
*************             PROGRAM IDENTIFICATION              *************
*                                                                         *
*      THIS ROUTINE WRITES A RECORD TO THE STANDARD BUILDING WORK FILE
*                                                                         *
***************************************************************************
*                                                                         *
***************************************************************************
*                                                                         *
*************             VARIABLE IDENTIFICATION             *************
*                                                                         *
*      BDRHI - BUILDING DOOR HEIGHT
*      BDRTHI - BUILDING DOOR THICKNESS
*      BDRWID - BUILDING DOOR WIDTH
*      BLEN - BUILDING LENGTH
*      BNAME - BUILDING NAME
*      BRAD - BUILDING ROOF RADIUS
*      BROOF - BUILDING ROOF TYPE
*      BRTHIC - BUILDING ROOF THICKNESS
*      BSIDE - BUILDING SIDE WALL HEIGHT
*      BTYPE - STANDARD BUILDING TYPE
*      BWID - BUILDING WIDTH
*      BWTHIC - BUILDING WALL THICKNESS
*      KEY - INDEX KEY FOR STANDARD BUILDING WORK FILE
*      RMAXWT - BUILDING MAXIMUM STORAGE WEIGHT
*      TOTKEY - NUMBER OF RECORDS IN DATA BASE
*                                                                         *
****************             SUBROUTINE NAMES             ******************
*                                                                         *
*      CALLED BY:
*                  DELREC
*                  ITEM
*                                                                         *
*      CALLS:  NONE
*                                                                         *
***************************************************************************
*                                                                         *
       COMMON /PT1/BTYPE,BNAME,BROOF,BLEN,BWID,BRAD,BSIDE,BWTHIC,
     &   BRTHIC,BDRHI,BDRWID,BDRTHI,RMAXWT
       COMMON /PT3/TYPE,TOTKEY,KEY
       INTEGER BTYPE*2
       CHARACTER BNAME*6,BROOF*3
       WRITE (02'KEY) BTYPE,BNAME,BROOF,BLEN,BWID,BRAD,BSIDE,BWTHIC,
     &   BRTHIC,BDRHI,BDRWID,BDRTHI,RMAXWT
      RETURN
*                                                                         *
****************                END RITE                  ******************
      END
```

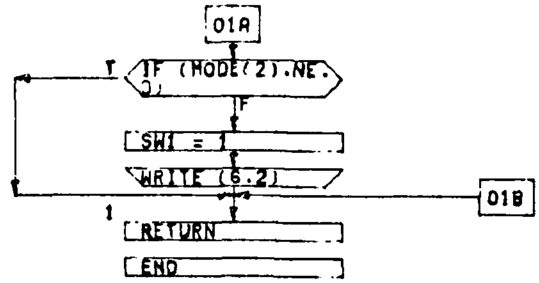
Appendix E

MSADB Utility Program Flow Chart

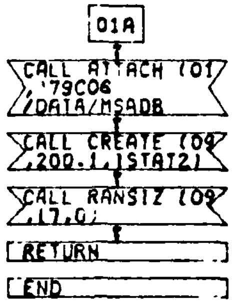
MSADBUP


01A
CALL START (SWI)
IF (SWI.NE.0)
T
F
CALL OPEN
CALL INITIAL
CALL ACTION
01B
1
STOP
END

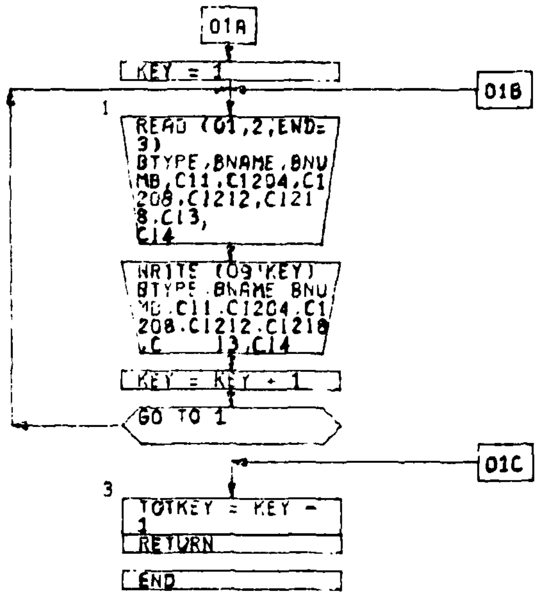
SUBROUTINE START



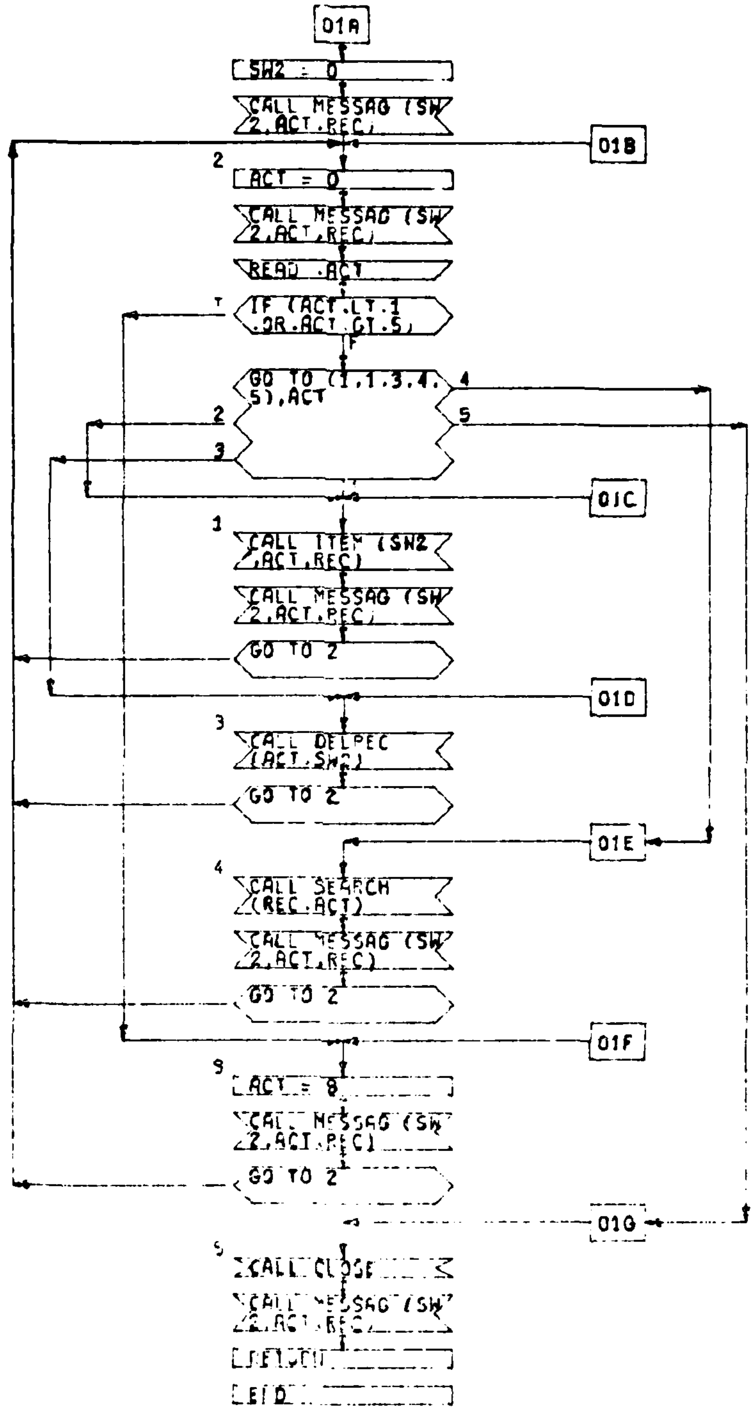
SUBROUTINE OPEN



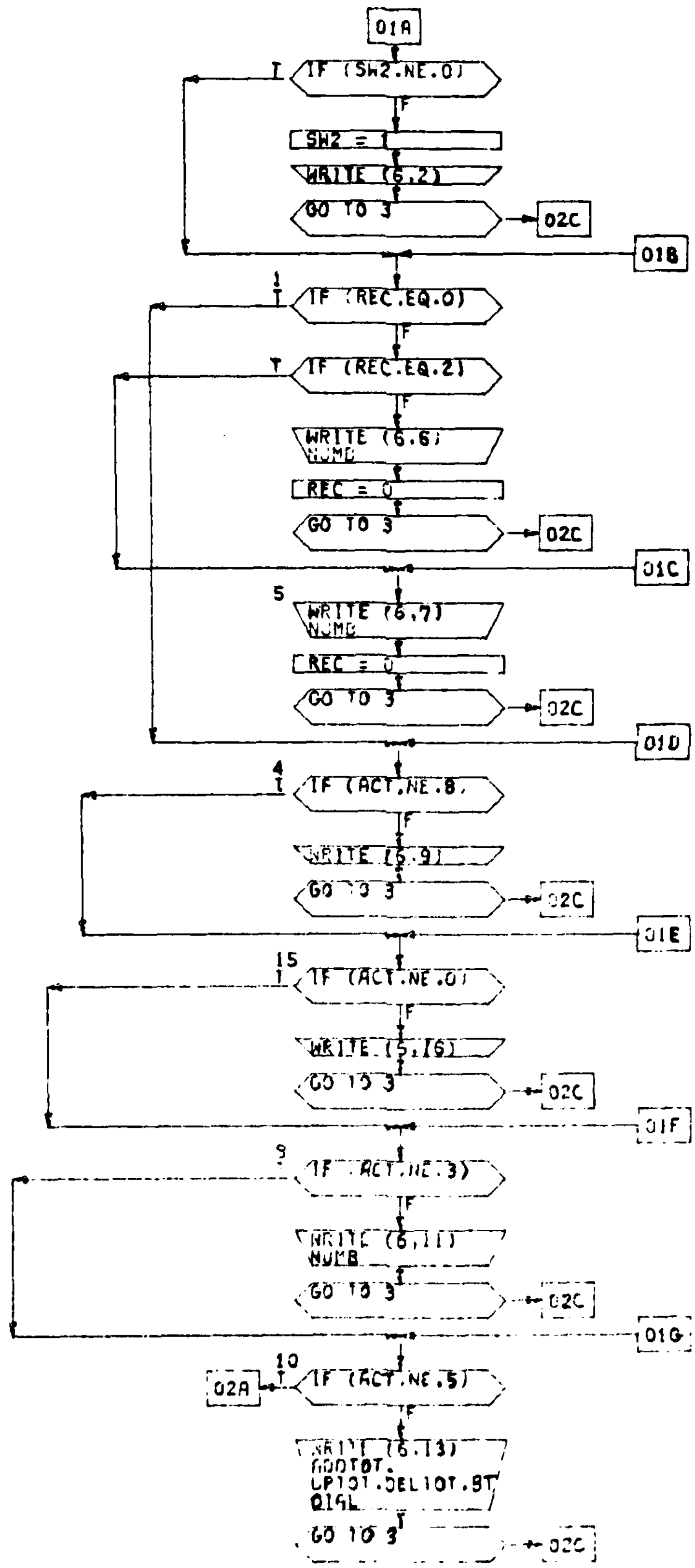
SUBROUTINE INITIAL



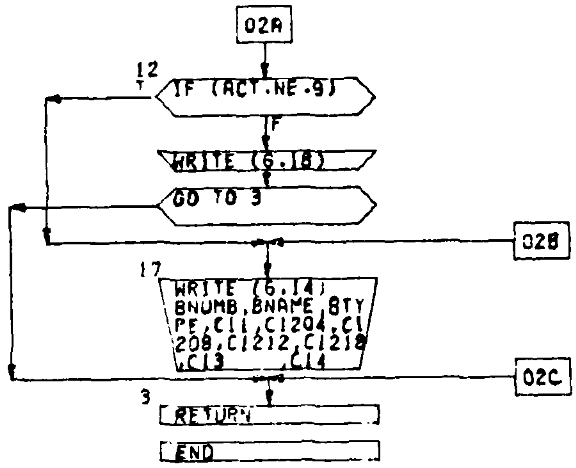
SUBROUTINE ACTION



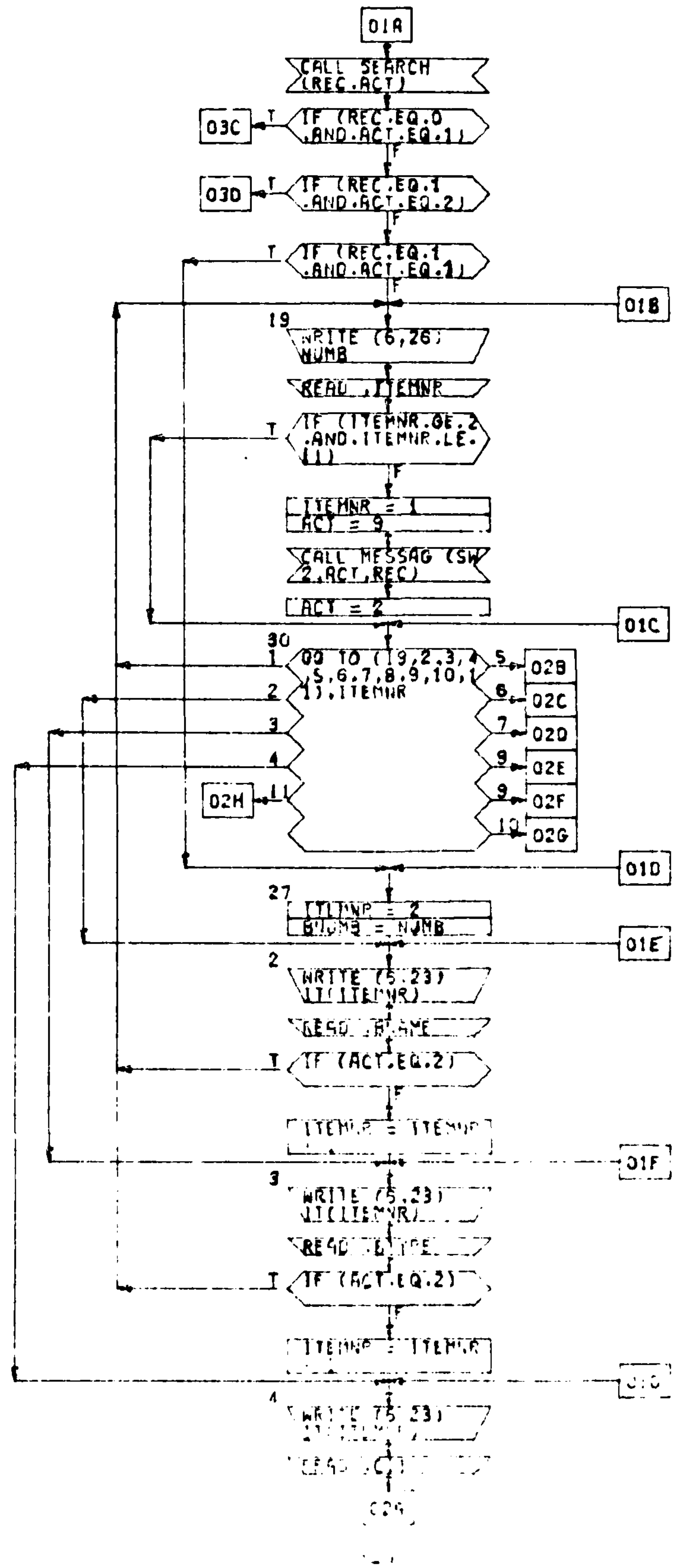
SUBROUTINE MESSAG



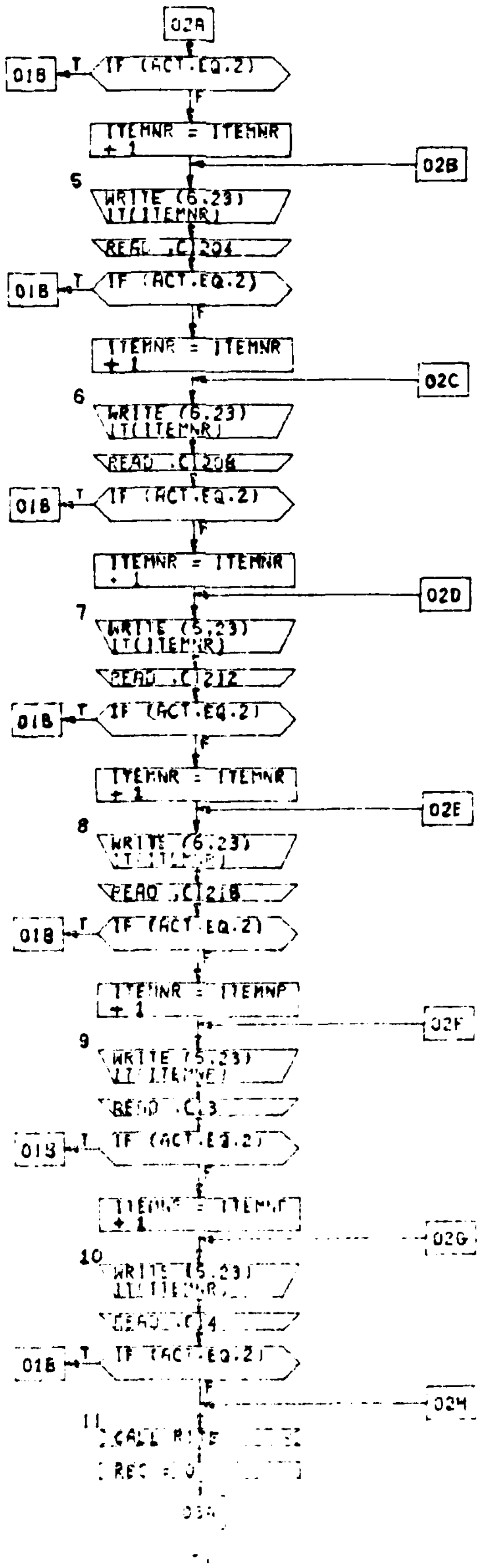
SUBROUTINE MESSAG



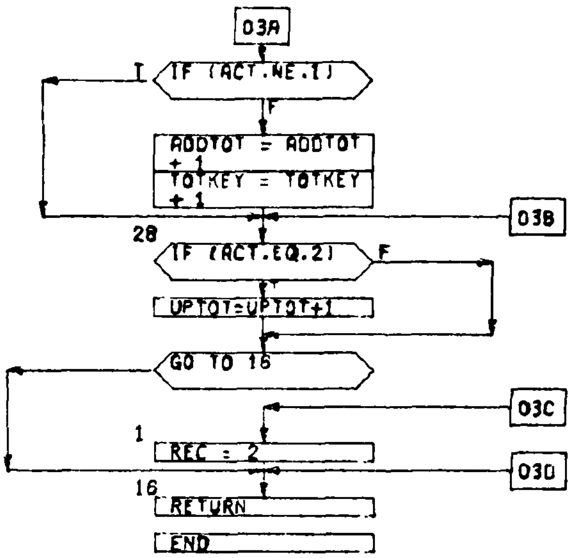
SUBROUTINE ITEM



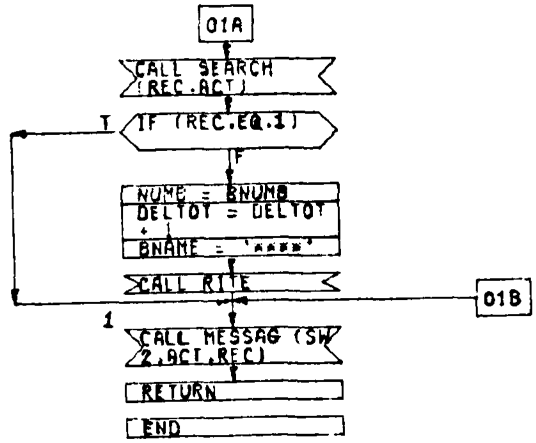
UTINE ITEM


02A
01B
T
IF (ACT.EQ.2)
F
ITEMNR = ITEMNR + 1
02B
5
WRITE (6,23) IT(ITEMNR)
READ ,C)204
01B
T
IF (ACT.EQ.2)
F
ITEMNR = ITEMNR + 1
02C
6
WRITE (6,23) IT(ITEMNR)
READ ,C)208
01B
T
IF (ACT.EQ.2)
F
ITEMNR = ITEMNR + 1
02D
7
WRITE (6,23) IT(ITEMNR)
READ ,C)212
01B
T
IF (ACT.EQ.2)
F
ITEMNR = ITEMNR + 1
02E
8
WRITE (6,23)
READ ,C)216
01B
T
IF (ACT.EQ.2)
F
02F
9
01B
T
02G
10
01B
T
IF (ACT.EQ.2)
F
02H
11

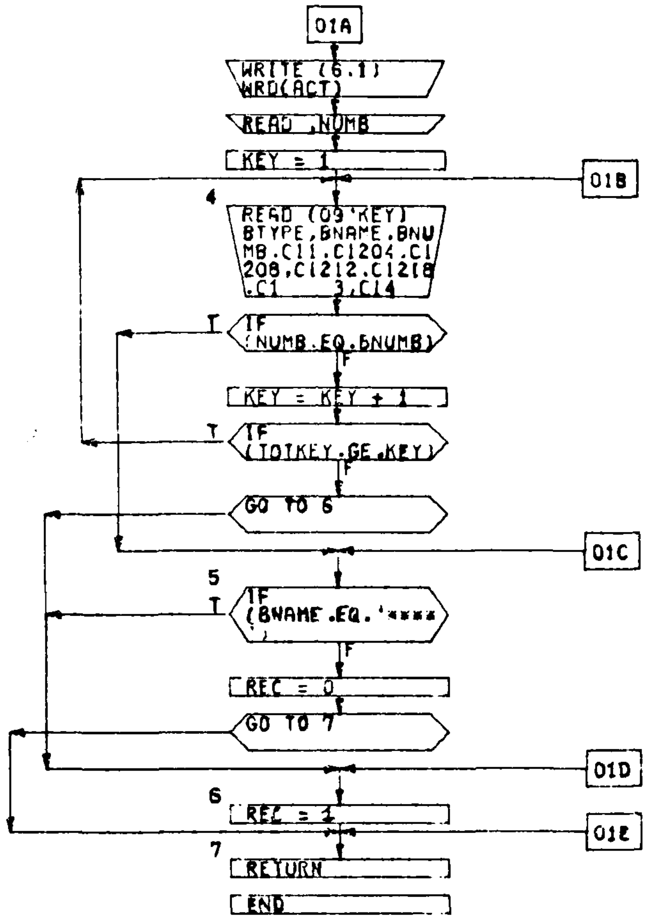
·ΟUTINE ITEM


03A
IF (ACT.NE.1)
F
ADDTOT = ADDTOT + 1
TOTKEY = TOTKEY + 1
03B
28
IF (ACT.EQ.2)
F
UPTOT=UPTOT+1
GO TO 16
03C
1
REC = 2
03D
16
RETURN
END

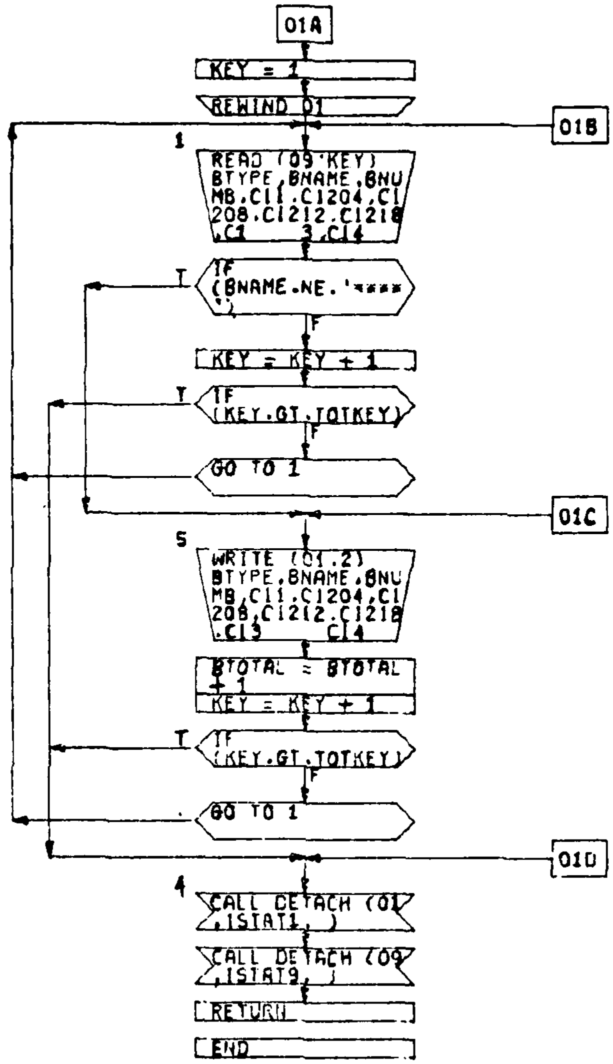
SUBROUTINE DELREC



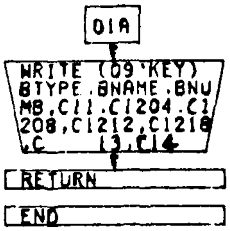
SUBROUTINE SEARCH



SUBROUTINE CLOSE

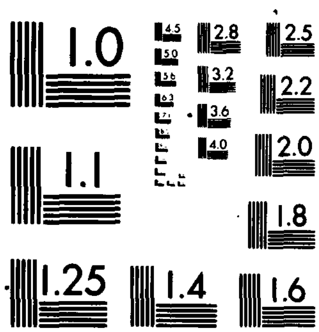


SUBROUTINE RITE



AD-A083 708 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/G 15/5
OPTIMIZATION OF MUNITIONS STORAGE.(U)
DEC 79 B A BOGGS, L M GUSMUS
UNCLASSIFIED AFIT/GSM/SM/79D-15 NL

3 of 4
AD
A083708



MICROCOPY RESOLUTION TEST CHART

Appendix F

MSADB Utility Program Source Listing

```
*#RUN * = /OBJECT/MSADBUP (NOGO)
C       MUNITION STORAGE AREA DATA BASE UTILITY PROGRAM        23 NOV 79
C
C
C
C
************************************************************************
*                                                                      *
*         MSADBUP MAIN                                                 *
*                                                                      *
************************************************************************
*                                                                      *
*************             PROGRAM IDENTIFICATION            *************
*                                                                      *
*      MSADBUP IS RESPONSIBLE FOR CREATING AND MAINTAINING RECORDS IN
*      MUNITION STORAGE AREA DATA BASE
*                                                                      *
************************************************************************
*                                                                      *
C            ENTER A ONE DIGIT VALUE AND HIT CARRIAGE RETURN
C                1 - ADD
C                2 - CHANGE
C                3 - DELETE
C                4 - DISPLAY
C                5 - TERMINATE
*                                                                      *
C            IF "1" (ADD RECORD)
C              FILL IN APPROPRIATE DATA ITEMS AS THEY ARE PRESENTED
C                           AND HIT THE CARRIAGE RETURN
*                                                                      *
C            IF "2" (CHANGE RECORD)
C              ENTER 6 CHARACTER BUILDING NUMBER OF RECORD TO BE CHANGED
C                  AND HIT THE CARRIAGE RETURN
*                                                                      *
C              THEN ENTER TWO DIGIT ITEM NUMBER TO BE CHANGED
C                    AND HIT THE CARRIAGE RETURN
*                                                                      *
C                 ITEM  ITEM                                  INPUT FORMAT
C                  NR
C                  02 - NAME (6A)                             AAAAAA
C                  03 - TYPE (2N)                             99
C                  04 - CLASS/DIV 1.1 NEW (IN POUNDS)         9999999
C                  05 - CLASS/DIV/CAT 1.2 04 NEW (IN POUNDS)  9999999
C                  06 - CLASS/DIV/CAT 1.2 08 NEW (IN POUNDS)  9999999
C                  07 - CLASS/DIV/CAT 1.2 12 NEW (IN POUNDS)  9999999
C                  08 - CLASS/DIV/CAT 1.2 18 NEW (IN POUNDS)  9999999
C                  09 - CLASS/DIV 1.3 NEW (IN POUNDS)         9999999
C                  10 - CLASS/DIV 1.4 NEW (IN POUNDS)         9999999
*                                                                      *
```

```
C                  11 - FINISHED CURRENT TRANSACTION
*                                                                          *
C            IF "3" (DELETE RECORD)
C              ENTER 6 CHARACTER BUILDING NUMBER OF RECORD TO BE DELETED
C                   AND HIT THE CARRIAGE RETURN
*                                                                          *
C            IF "4" (DISPLAY RECORD)
C              ENTER 6 CHARACTER BUILDING NUMBER OF RECORD TO BE DISPLAYED
C                   AND HIT THE CARRIAGE RETURN
*                                                                          *
C            IF "5" TERMINATE THE EXECUTION OF THIS PROGRAM
*                                                                          *
****************************************************************************
*                                                                          *
*************             VARIABLE IDENTIFICATION            *************
*                                                                          *
*      SW1 - INDICATOR SWITCH FOR PROPER MODE OF OPERATION (ASCII)
*                                                                          *
****************               SUBROUTINE NAMES            *****************
*                                                                          *
*      CALLED BY:  NONE
*                                                                          *
*      CALLS:
*              START - VERIFIES PROPER MODE OF OPERATION (ASCII)
*              OPEN - OPENS NECESSARY FILES USED BY MSADBUP
*              INITIAL - CREATES MUNITION STORAGE AREA WORK FILE
*              ACTION - PERFORMS REQUESTED ACTIONS ON DATA BASE
*                                                                          *
****************************************************************************
*                                                                          *
      COMMON /PT1/BNUMB,BNAME,BTYPE,C11,C1204,C1208,C1212,C1218,C13,C14
      COMMON /PT2/UPTOT,DELTOT,ADDTOT,BTOTAL
      COMMON /PT3/NUMB,TOTKEY,KEY
      CHARACTER BNAME*6,BNUMB*6,NUMB*6
      INTEGER BTYPE*2
      INTEGER *7 C11,C1204,C1208,C1212,C1218,C13,C14
      INTEGER SW1,UPTOT,DELTOT,ADDTOT,BTOTAL,TOTKEY,KEY
      CALL START (SW1)
      IF (SW1.NE.0) GO TO 1
      CALL OPEN
      CALL INITIAL
      CALL ACTION
    1 STOP
      END
*                                                                          *
****************               END MAIN                    *****************
```

```
****************************************************************************
*                                                                          *
      SUBROUTINE START (SW1)
*                                                                          *
****************************************************************************
*                                                                          *
*************              PROGRAM IDENTIFICATION              *************
*                                                                          *
*      THIS ROUTINE VERIFIES PROPER MODE OF OPERATION (ASCII)
*                                                                          *
****************************************************************************
*                                                                          *
****************************************************************************
*                                                                          *
*************              VARIABLE IDENTIFICATION             *************
*                                                                          *
*      MODE(2) - SYSTEM VARIABLE:  0 - BCD, 1 - ASCII
*      SW2 - INDICATOR SWITCH FOR PROPER MODE OF OPERATION (ASCII)
*                                                                          *
****************                SUBROUTINE NAMES             *****************
*                                                                          *
*      CALLED BY:  MAIN
*                                                                          *
*      CALLS:  NONE
*                                                                          *
****************************************************************************
*                                                                          *
       INTEGER SW1*1
       IF (MODE(2).NE.0) GO TO 1
       SW1 = 1
       WRITE (6,2)
*                                                                          *
    2  FORMAT (5X,"PLEASE RESTART USING 'RUN'")
*                                                                          *
    1 RETURN
      END
*                                                                          *
****************                 END START                   *****************
```

```
****************************************************************************
*                                                                          *
      SUBROUTINE OPEN
*                                                                          *
****************************************************************************
*                                                                          *
*************              PROGRAM IDENTIFICATION             *************
*                                                                          *
*     THIS ROUTINE IS USED TO OPEN EXTERNAL FILES FOR PROGRAM CONTROL
*                                                                          *
****************************************************************************
*                                                                          *
****************************************************************************
*                                                                          *
*************              VARIABLE IDENTIFICATION            *************
*                                                                          *
*     ISTAT1 - FILE STATUS VARIABLE FOR PERMANENT FILE
*     ISTAT9 - FILE STATUS VARIABLE FOR WORK FILE
*                                                                          *
****************               SUBROUTINE NAMES          ******************
*                                                                          *
*     CALLED BY:  MAIN
*                                                                          *
*     CALLS:
*             ATTACH - OPENS PERMANENT FILE
*             CREATE - CREATES A TEMPORARY WORK FILE
*             RANSIZ - SPECIFIES THE RECORD SIZE OF THE TEMPORARY
*                      WORK FILE
*                                                                          *
****************************************************************************
*                                                                          *
*                                                                          *
***     CARD 999 WILL HAVE TO BE CHANGED FOR NEW USERS                   ***
*                                                                          *
        CALL ATTACH (01,"79C06/DATA/MSADB;",3,0,ISTAT1, )
        CALL CREATE (09,200,1,ISTAT2)
        CALL RANSIZ (09,17,0)
        RETURN
      END
*                                                                          *
****************               END OPEN                  ******************
```

```
*************************************************************************
*                                                                       *
      SUBROUTINE INITIAL
*                                                                       *
*************************************************************************
*                                                                       *
*************             PROGRAM IDENTIFICATION            *************
*                                                                       *
*      THIS ROUTINE COPIES PERMANENT DATA BASE FILE ON TO THE TEMPORARY
*      WORK FILE
*                                                                       *
*************             VARIABLE IDENTIFICATION           *************
*                                                                       *
*      BNAME - BUILDING NAME
*      BNUMB - BUILDING NUMBER
*      BTYPE - STANDARD BUILDING TYPE
*      C11 -NET EXPLOSIVE WEIGHT (NEW) FOR MUNITION CLASS/DIVISION 1.1
*      C1204 - NEW FOR CLASS/DIVISION/CAT 1.2 04
*      C1208 - NEW FOR CLASS/DIVISION/CAT 1.2 08
*      C1212 - NEW FOR CLASS/DIVISION/CAT 1.2 12
*      C1218 - NEW FOR CLASS/DIVISION/CAT 1.2 18
*      C13 - NEW FOR CLASS/DIVISION 1.3
*      C14 - NEW FOR CLASS/DIVISION 1.4
*      KEY - INDEX KEY FOR MUNITION STORAGE AREA WORK FILE
*      TOTKEY - NUMBER OF RECORDS IN DATA BASE
*                                                                       *
****************              SUBROUTINE NAMES           ****************
*                                                                       *
*      CALLED BY:  MAIN
*      CALLS:  NONE
*                                                                       *
*************************************************************************
*                                                                       *
       COMMON /PT1/BNUMB,BNAME,BTYPE,C11,C1204,C1208,C1212,C1218,C13,C14
       COMMON /PT3/NUMB,TOTKEY,KEY
       CHARACTER BNAME*6,BNUMB*6,NUMB*6
       INTEGER BTYPE*2
       INTEGER *7 C11,C1204,C1208,C1212,C1218,C13,C14
       INTEGER TOTKEY,KEY
       KEY = 1
   1   READ (01,2,END=3) BTYPE,BNAME,BNUMB,C11,C1204,C1208,C1212,C1218,C13,
    &    C14
       WRITE (09'KEY) BTYPE,BNAME,BNUMB,C11,C1204,C1208,C1212,C1218,C13,C14
       KEY = KEY + 1
       GO TO 1
   3   TOTKEY = KEY - 1
*                                                                       *
   2   FORMAT (I2,2A6,7I7)
*                                                                       *
       RETURN
       END
*                                                                       *
****************                END INITIAL              ****************
```

```
**********************************************************************
*                                                                    *
      SUBROUTINE ACTION
*                                                                    *
*************            PROGRAM IDENTIFICATION           *************
*                                                                    *
*      DRIVER ROUTINE THAT SELECTS THE APPROPRIATE ACTION TO BE
*      ACCOMPLISHED
**********************************************************************
*                                                                    *
*************            VARIABLE IDENTIFICATION          *************
*                                                                    *
*      ACT - TYPE OF ACTION (VALUE - 1 TO 5)
*      REC - CONTROL SWITCH: 0 - NOT FOUND, 1 - FOUND
*      SW2 - FIRST PASS SWITCH:  0 - FIRST PASS, 1 - SUBSEQUENT PASSES
*                                                                    *
***************            SUBROUTINE NAMES            ****************
*                                                                    *
*      CALLED BY:  MAIN
*                                                                    *
*      CALLS:
*              MESSAG - PRINTS APPROPRIATE MESSAGE
*              ITEM - ADDS A NEW RECORD OR CHANGES AN EXISTING RECORD
*              DELREC - DELETES SPECIFIED RECORD FROM DATA BASE
*              SEARCH - SEARCHES WORK FILE FOR SPECIFIED RECORD
*              CLOSE - TERMINATES THE PROGRAM
*                                                                    *
**********************************************************************
*                                                                    *
       INTEGER ACT*1,SW2*1,REC*1
       SW2 = 0
       CALL MESSAG (SW2,ACT,REC)
    2  ACT = 0
       CALL MESSAG (SW2,ACT,REC)
       READ ,ACT
       IF (ACT.LT.1.OR.ACT.GT.5) GO TO 8
       GO TO (1,1,3,4,5),ACT
    1  CALL ITEM (SW2,ACT,REC)
       CALL MESSAG (SW2,ACT,REC)
       GO TO 2
    3  CALL DELREC (ACT,SW2)
       GO TO 2
    4  CALL SEARCH (REC,ACT)
       CALL MESSAG (SW2,ACT,REC)
       GO TO 2
    8  ACT = 8
       CALL MESSAG (SW2,ACT,REC)
       GO TO 2
    5  CALL CLOSE
       CALL MESSAG (SW2,ACT,REC)
*                                                                    *
    7  FORMAT (I1)
*                                                                    *
      RETURN
      END
*                                                                    *
***************              END ACTION                ****************
```

```
*************************************************************************
*                                                                       *
      SUBROUTINE MESSAG (SW2,ACT,REC)
*                                                                       *
*************************************************************************
*                                                                       *
*************             PROGRAM IDENTIFICATION            *************
*                                                                       *
*     PRINTS THE APPROPRIATE MESSAGES
*                                                                       *
*************************************************************************
*                                                                       *
*************************************************************************
*                                                                       *
*************             VARIABLE IDENTIFICATION           *************
*                                                                       *
*     ACT - TYPE OF ACTION BEING PERFORMED
*     ADDTOT - NUMBER OF RECORDS ADDED
*     BNAME - BUILDING NAME
*     BNUMB - BUILDING NUMBER
*     BTYPE - STANDARD BUILDING TYPE
*     BTOTAL - NUMBER OF RECORDS IN DATA BASE
*     C11 -NET EXPLOSIVE WEIGHT (NEW) FOR MUNITION CLASS/DIVISION 1.1
*     C1204 - NEW FOR CLASS/DIVISION/CAT 1.2 04
*     C1208 - NEW FOR CLASS/DIVISION/CAT 1.2 08
*     C1212 - NEW FOR CLASS/DIVISION/CAT 1.2 12
*     C1218 - NEW FOR CLASS/DIVISION/CAT 1.2 18
*     C13 - NEW FOR CLASS/DIVISION 1.3
*     C14 - NEW FOR CLASS/DIVISION 1.4
*     DELTOT - NUMBER OF RECORDS DELETED FROM DATA BASE
*     NUMB - BUILDING NUMBER
*     REC - FOUND/NOT FOUND SWITCH
*     SW2 - FIRST PASS SWITCH: 0 - FIRST PASS, 1 - SUBSEQUENT PASSES
*     UPTOT - NUMBER OF RECORDS UPDATED
*                                                                       *
****************             SUBROUTINE NAMES          ******************
*                                                                       *
*     CALLED BY:
*                 ACTION
*                 ITEM
*                 DELREC
*                                                                       *
*     CALLS:  NONE
*                                                                       *
*************************************************************************
*                                                                       *
        COMMON /PT1/BNUMB,BNAME,BTYPE,C11,C1204,C1208,C1212,C1218,C13,C14
      COMMON /PT2/UPTOT,DELTOT,ADDTOT,BTOTAL
      COMMON /PT3/NUMB,TOTKEY,KEY
      CHARACTER BNAME*6,BNUMB*6,NUMB*6
        INTEGER BTYPE*2
        INTEGER *7 C11,C1204,C1208,C1212,C1218,C13,C14
        INTEGER ACT*1,SW2*1,REC*1,UPTOT,DELTOT,ADDTOT,BTOTAL,TOTKEY,KEY
        IF (SW2.NE.0) GO TO 1
        SW2 = 1
        WRITE (6,2)
        GO TO 3
    1   IF (REC.EQ.0) GO TO 4
```

```
      IF (REC.EQ.2) GO TO 5
      WRITE (6,6) NUMB
      REC = 0
      GO TO 3
    5 WRITE (6,7) NUMB
      REC = 0
      GO TO 3
    4 IF (ACT.NE.8) GO TO 15
      WRITE (6,9)
      GO TO 3
   15 IF (ACT.NE.0) GO TO 8
      WRITE (6,16)
      GO TO 3
    8 IF (ACT.NE.3) GO TO 10
      WRITE (6,11) NUMB
      GO TO 3
   10 IF (ACT.NE.5) GO TO 12
      WRITE (6,13) ADDTOT, UPTOT,DELTOT,BTOTAL
      GO TO 3
   12 IF (ACT.NE.9) GO TO 17
      WRITE (6,18)
      GO TO 3
   17 WRITE (6,14) BNUMB,BNAME,BTYPE,C11,C1204,C1208,C1212,C1218,C13,C14
*                                                                      *
    2 FORMAT (//5X,"WELCOME TO THE MUNITION STORAGE AREA DATA BASE"//)
*                                                                      *
    9 FORMAT (5X,"OPTIONS:",
     &   /25X,"1 - ADD RECORD"/,25X,"2 - CHANGE RECORD"/,25X,
     &   "3 - DELETE RECORD"/,25X,"4 - DISPLAY RECORD"/,25X,
     &   "5 - TERMINATE JOB"//)
*                                                                      *
    6 FORMAT (5X,"RECORD ",A6," DOES NOT EXIST"//)
*                                                                      *
    7 FORMAT (5X,"RECORD ",A6," ALREADY EXISTS"//)
*                                                                      *
   11 FORMAT (5X,"RECORD ",A6," HAS BEEN DELETED FROM THE DATA BASE"//)
*                                                                      *
   13 FORMAT (5X,"YOU ARE NOW EXITING THE UPDATE PROGRAM"//
     &     15X,"ADDED - ",I3/,15X,"CHANGED - ",I3/,15X,"DELETED - ",
     &     I3/,15X,"TOTAL NUMBER OF RECORDS IN DATA BASE - ",I3//)
*                                                                      *
   14 FORMAT (5X,"BUILDING NR - ",A6,/,10X,"NAME - ",A6,/,10X,"TYPE - ",
     & I2,/,10X,"CLASS/DIV 1.1 NEW - " I7," LBS"/,10X,
     & "CLASS/DIV/CAT 1.2 04 NEW - ",I7," LBS"/,10X,
     & "CLASS/DIV/CAT 1.2 08 NEW - ",I7," LBS"/,10X,
     & "CLASS/DIV/CAT 1.2 12 NEW - ",I7," LBS"/,10X,
     & "CLASS/DIV/CAT 1.2 18 NEW - ",I7," LBS"/,10X,
     & "CLASS/DIV 1.3 NEW - ",I7," LBS"/,10X,
     & "CLASS/DIV 1.4 NEW - ",I7," LBS"/)
*                                                                      *
   16 FORMAT (5X,"ENTER THE ONE DIGIT TRANSACTION DESIRED:"/)
*                                                                      *
   18 FORMAT (5X,"OPTIONS:",
     &     /10X,"02 - NAME (6A)",/10X,"03 - STD TYPE BLDG (99)",
     &   /10X."04 - CLASS/DIV 1.1 NEW (9999999) IN LBS"/,10X,
     &    "05 - CLASS/DIV/CAT 1.2 04 NEW (9999999) IN LBS"/,10X,
     &    "06 - CLASS/DIV/CAT 1.2 08 NEW (9999999) IN LBS"/,10X,
     &    "07 - CLASS/DIV/CAT 1.2 12 NEW (9999999) IN LBS"/,10X,
```

```
     &     "08 - CLASS/DIV/CAT 1.2 18 NEW (9999999) IN LBS"/,10X,
     &     "09 - CLASS/DIV 1.3 NEW (9999999) IN LBS"/,10X,
     &     "10 - CLASS/DIV 1.4 NEW (9999999) IN LBS"//,10X,
     &     "11 - FINISHED WITH THIS RECORD"//)
*                                                                          *
   3 RETURN
     END
*                                                                          *
*****************                END MESSAG                 ******************
```

```
*************************************************************************
*                                                                       *
      SUBROUTINE ITEM (SW2,ACT,REC)
*                                                                       *
*************************************************************************
*                                                                       *
*************             PROGRAM IDENTIFICATION             *************
*                                                                       *
*      THIS ROUTINE WILL EITHER CREATE A NEW RECORD OR
*      UPDATE SPECIFIED ITEMS OF AN EXISTING RECORD
*                                                                       *
*************************************************************************
*                                                                       *
*************************************************************************
*                                                                       *
*************             VARIABLE IDENTIFICATION            *************
*                                                                       *
*      ACT - TYPE OF ACTION IN PROGRESS
*      ADDTOT - NUMBER OF RECORDS ADDED TO DATA BASE
*      BNAME - BUILDING NAME
*      BNUMB - BUILDING NUMBER
*      BTYPE - STANDARD BUILDING TYPE
*      C11 -NET EXPLOSIVE WEIGHT (NEW) FOR MUNITION CLASS/DIVISION 1.1
*      C1204 - NEW FOR CLASS/DIVISION/CAT 1.2 04
*      C1208 - NEW FOR CLASS/DIVISION/CAT 1.2 08
*      C1212 - NEW FOR CLASS/DIVISION/CAT 1.2 12
*      C1218 - NEW FOR CLASS/DIVISION/CAT 1.2 18
*      C13 - NEW FOR CLASS/DIVISION 1.3
*      C14 - NEW FOR CLASS/DIVISION 1.4
*      IT - ARRAY CONTAINING ITEM NAMES TO BE PROCESSED
*      ITEMNR - INDEX FOR IT ARRAY
*      NUMB - BUILDING NUMBER
*      REC - FOUND/NOT FOUND SWITCH
*      TOTKEY - NUMBER OF RECORDS IN DATA BASE
*                                                                       *
***************                SUBROUTINE NAMES              *****************
*                                                                       *
*      CALLED BY:  ACTION
*                                                                       *
*      CALLS:
*              SEARCH - SEARCHES WORK FILE FOR SPECIFIED RECORD
*              MESSAG - PRINTS SPECIFIED MESSAGE
*              RITE - WRITES SPECIFIED RECORD
*                                                                       *
*************************************************************************
*                                                                       *
        COMMON /PT1/BNUMB,BNAME,BTYPE,C11,C1204,C1208,C1212,C1218,C13,C14
      COMMON /PT2/UPTOT,DELTOT,ADDTOT,BTOTAL
      COMMON /PT3/NUMB,TOTKEY,KEY
      CHARACTER BNAME*6,BNUMB*6,NUMB*6
        INTEGER BTYPE*2
        INTEGER *7 C11,C1204,C1208,C1212,C1218,C13,C14
        INTEGER ACT*1,ITEMNR*2,REC*1,ADDTOT,UPTOT,TOTKEY,KEY
         CHARACTER IT*41(10)/"BLDG NUMBER (XXXXXX)","NAME (AAAAAA)",
     &    "TYPE (99)","CLASS/DIV 1.1 NEW (9999999) IN LBS",
     &    "CLASS/DIV/CAT 1.2 04 NEW (9999999) IN LBS",
     &    "CLASS/DIV/CAT 1.2 08 NEW (9999999) IN LBS",
     &    "CLASS/DIV/CAT 1.2 12 NEW (9999999) IN LBS",
```

```
     &    "CLASS/DIV/CAT 1.2 18 NEW (9999999) IN LBS",
     &    "CLASS/DIV 1.3 NEW (9999999) IN LBS",
     &    "CLASS/DIV 1.4 NEW (9999999) IN LBS"/
      CALL SEARCH (REC,ACT)
      IF (REC.EQ.0.AND.ACT.EQ.1) GO TO 1
      IF (REC.EQ.1.AND.ACT.EQ.2) GO TO 16
      IF (REC.EQ.1.AND.ACT.EQ.1) GO TO 27
   19 WRITE (6,26) NUMB
      READ ,ITEMNR
      IF (ITEMNR.GE.2.AND.ITEMNR.LE.11)GO TO 30
      ITEMNR = 1
      ACT = 9
      CALL MESSAG (SW2,ACT,REC)
      ACT = 2
   30 GO TO (19,2,3,4,5,6,7,8,9,10,11),ITEMNR
   27 ITEMNR = 2
      BNUMB = NUMB
    2 WRITE (6,23) IT(ITEMNR)
      READ ,BNAME
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
    3 WRITE (6,23) IT(ITEMNR)
      READ ,BTYPE
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
    4 WRITE (6,23) IT(ITEMNR)
      READ ,C11
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
    5 WRITE (6,23) IT(ITEMNR)
      READ ,C1204
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
    6 WRITE (6,23) IT(ITEMNR)
      READ ,C1208
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
    7 WRITE (6,23) IT(ITEMNR)
      READ ,C1212
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
    8 WRITE (6,23) IT(ITEMNR)
      READ ,C1218
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
    9 WRITE (6,23) IT(ITEMNR)
      READ ,C13
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
   10 WRITE (6,23) IT(ITEMNR)
      READ ,C14
      IF (ACT.EQ.2) GO TO 19
   11 CALL RITE
      REC = 0
      IF (ACT.NE.1) GO TO 28
      ADDTOT = ADDTOT + 1
      TOTKEY = TOTKEY + 1
   28 IF (ACT.EQ.2) UPTOT=UPTOT+1
```

```
      GO TO 16
   1  REC = 2
*                                                                         *
  23  FORMAT (5X,"ENTER THE INFORMATION FOR THE ",A41)
*                                                                         *
  26  FORMAT (5X,"ENTER THE 2 DIGIT ITEM NUMBER OF THE ITEM TO BE ",
     &     /10X,"CHANGED FOR BUILDING NR ",A6//)
*                                                                         *
  16 RETURN
     END
*                                                                         *
****************                END ITEM               ******************
```

```
*************************************************************************
*                                                                       *
      SUBROUTINE DELREC (ACT,SW2)
*                                                                       *
*************************************************************************
*                                                                       *
*************             PROGRAM IDENTIFICATION            *************
*                                                                       *
*       THIS ROUTINE WILL DELETE THE SPECIFIED EXISTING RECORD FROM
*       THE DATA BASE
*                                                                       *
*************************************************************************
*                                                                       *
*************             VARIABLE IDENTIFICATION           *************
*                                                                       *
*       ACT - TYPE OF ACTION IN PROGRESS
*       BNAME - BUILDING NAME
*       BNUMB - BUILDING NUMBER
*       NUMB - BUILDING NUMBER
*       DELTOT - NUMBER OF RECORDS DELETED
*       REC - FOUND/NOT FOUND SWITCH
*       SW2 - FIRST PASS SWITCH: 0 - FIRST PASS, 1 - SUBSEQUENT PASSES
*                                                                       *
****************             SUBROUTINE NAMES             ******************
*                                                                       *
*       CALLED BY:  ACTION
*                                                                       *
*       CALLS:
*               SEARCH - SEARCHES WORK FILE FOR SPECIFIED RECORD
*               RITE - WRITES DELETED IDENTIFIER FOR SPECIFIED RECORD
*               MESSAG - PRINTS SPECIFIED MESSAGE
*                                                                       *
*************************************************************************
*                                                                       *
        COMMON /PT1/BNUMB,BNAME,BTYPE,C11,C1204,C1208,C1212,C1218,C13,C14
        COMMON /PT2/UPTOT,DELTOT,ADDTOT,BTOTAL
        COMMON /PT3/NUMB,TOTKEY,KEY
        CHARACTER BNAME*6,BNUMB*6,NUMB*6
        INTEGER BTYPE*2
        INTEGER *7 C11,C1204,C1208,C1212,C1218,C13,C14
        INTEGER SW2*1,ACT*1,REC*1,DELTOT,TOTKEY,KEY
        CALL SEARCH (REC,ACT)
        IF (REC.EQ.1) GO TO 1
        NUMB = BNUMB
        DELTOT = DELTOT + 1
        BNAME = "****"
        CALL RITE
    1   CALL MESSAG (SW2,ACT,REC)
      RETURN
      END
*                                                                       *
****************                END DELREC                ******************
```

```
***************************************************************************
*                                                                         *
      SUBROUTINE SEARCH (REC,ACT)
*                                                                         *
***************************************************************************
*                                                                         *
*************             PROGRAM IDENTIFICATION              *************
*                                                                         *
*      THIS ROUTINE SEARCHES WORK FILE FOR SPECIFIED RECORD
*                                                                         *
***************************************************************************
*                                                                         *
***************************************************************************
*                                                                         *
*************             VARIABLE IDENTIFICATION             *************
*                                                                         *
*      ACT - TYPE OF ACTION IN PROGRESS
*      BNAME - BUILDING NAME
*      BNUMB - BUILDING NUMBER
*      BTYPE - STANDARD BUILDING TYPE
*      C11 -NET EXPLOSIVE WEIGHT (NEW) FOR MUNITION CLASS/DIVISION 1.1
*      C1204 - NEW FOR CLASS/DIVISION/CAT 1.2 04
*      C1208 - NEW FOR CLASS/DIVISION/CAT 1.2 08
*      C1212 - NEW FOR CLASS/DIVISION/CAT 1.2 12
*      C1218 - NEW FOR CLASS/DIVISION/CAT 1.2 18
*      C13 - NEW FOR CLASS/DIVISION 1.3
*      C14 - NEW FOR CLASS/DIVISION 1.4
*      KEY - INDEX KEY FOR MUNITION STORAGE AREA WORK FILE
*      NUMB - BUILDING NUMBER
*      REC - FOUND/NOT FOUND SWITCH
*      TOTKEY - NUMBER OF RECORDS IN DATA BASE
*      WRD - ARRAY CONTAINING TYPE OF ACTION IN PROGRESS
*                                                                         *
****************               SUBROUTINE NAMES           *****************
*                                                                         *
*      CALLED BY:
*                  ACTION
*                  ITEM
*                  DELREC
*                                                                         *
*      CALLS:  NONE
*                                                                         *
***************************************************************************
*                                                                         *
      COMMON /PT1/BNUMB,BNAME,BTYPE,C11,C1204,C1208,C1212,C1218,C13,C14
      COMMON /PT3/NUMB,TOTKEY,KEY
      CHARACTER BNAME*6,BNUMB*6,NUMB*6
      INTEGER BTYPE*2
      INTEGER *7 C11,C1204,C1208,C1212,C1218,C13,C14
      INTEGER REC*1,TOTKEY,KEY,ACT*1
      CHARACTER WRD*9(4)/"ADDED ","CHANGED","DELETED","DISPLAYED"/
      WRITE (6,1) WRD(ACT)
      READ ,NUMB
      KEY = 1
    4 READ (09'KEY) BTYPE,BNAME,BNUMB,C11,C1204,C1208,C1212,C1218,C13,C14
      IF (NUMB.EQ.BNUMB) GO TO 5
      KEY = KEY + 1
      IF (TOTKEY.GE.KEY) GO TO 4
```

```
      GO TO 6
    5 IF (BNAME.EQ."****") GO TO 6
      REC = 0
      GO TO 7
*                                                                        *
    1 FORMAT (5X,"ENTER THE 6 CHARACTER BUILDING NUMBER OF RECORD TO BE ",
     &   A9)
*                                                                        *
    6 REC = 1
    7 RETURN
      END
*                                                                        *
****************          END SEARCH               ******************
```

```
*************************************************************************
*                                                                       *
      SUBROUTINE CLOSE
*                                                                       *
*************************************************************************
*                                                                       *
*************             PROGRAM IDENTIFICATION             *************
*                                                                       *
*       THIS ROUTINE WRITES THE UPDATED DATA BASE BACK TO THE
*       PERMANENT FILE
*                                                                       *
*************************************************************************
*                                                                       *
*************************************************************************
*                                                                       *
*************             VARIABLE IDENTIFICATION            *************
*                                                                       *
*       BNAME - BUILDING NAME
*       BNUMB - BUILDING NUMBER
*       BTYPE - STANDARD BUILDING TYPE
*       BTOTAL - NUMBER OF RECORDS IN DATA BASE
*       C11 -NET EXPLOSIVE WEIGHT (NEW) FOR MUNITION CLASS/DIVISION 1.1
*       C1204 - NEW FOR CLASS/DIVISION/CAT 1.2 04
*       C1208 - NEW FOR CLASS/DIVISION/CAT 1.2 08
*       C1212 - NEW FOR CLASS/DIVISION/CAT 1.2 12
*       C1218 - NEW FOR CLASS/DIVISION/CAT 1.2 18
*       C13 - NEW FOR CLASS/DIVISION 1.3
*       C14 - NEW FOR CLASS/DIVISION 1.4
*       KEY - INDEX KEY FOR MUNITION STORAGE AREA WORK FILE
*       TOTKEY - NUMBER OF RECORDS IN DATA BASE
*                                                                       *
****************              SUBROUTINE NAMES           ****************
*                                                                       *
*       CALLED BY:  ACTION
*                                                                       *
*       CALLS:  DETACH - RELEASES SPECIFIED FILE FROM PROGRAM CONTROL
*                                                                       *
*************************************************************************
*                                                                       *
        COMMON /PT1/BNUMB,BNAME,BTYPE,C11,C1204,C1208,C1212,C1218,C13,C14
        COMMON /PT2/UPTOT,DELTOT,ADDTOT,BTOTAL
        COMMON /PT3/NUMB,TOTKEY,KEY
        CHARACTER BNAME*6,BNUMB*6,NUMB*6
        INTEGER BTYPE*2
        INTEGER *7 C11,C1204,C1208,C1212,C1218,C13,C14
        INTEGER BTOTAL,TOTKEY,KEY
        KEY = 1
        REWIND 01
    1   READ (09'KEY) BTYPE,BNAME,BNUMB,C11,C1204,C1208,C1212,C1218,C13,C14
        IF (BNAME.NE."****") GO TO 5
        KEY = KEY + 1
        IF (KEY.GT.TOTKEY) GO TO 4
        GO TO 1
    5   WRITE (01,2) BTYPE,BNAME,BNUMB,C11,C1204,C1208,C1212,C1218,C13,C14
        BTOTAL = BTOTAL + 1
        KEY = KEY + 1
        IF (KEY.GT.TOTKEY) GO TO 4
        GO TO 1
```

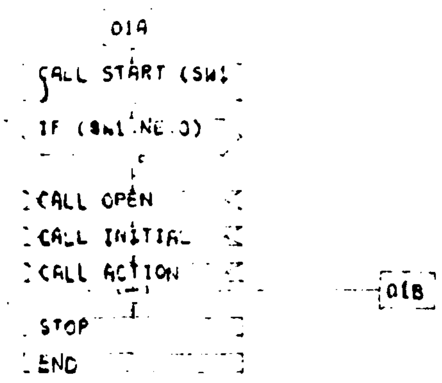
```
*                                                                        *
      2   FORMAT (I2,2A6,7I7)
*                                                                        *
      4   CALL DETACH (01,ISTAT1, )
          CALL DETACH (09,ISTAT9, )
        RETURN
        END
*                                                                        *
*****************                END CLOSE                ******************
```

```
*****************************************************************************
*                                                                           *
      SUBROUTINE RITE
*                                                                           *
*****************************************************************************
*                                                                           *
*************             PROGRAM IDENTIFICATION                *************
*                                                                           *
*      THIS ROUTINE WRITES A RECORD TO THE STANDARD BUILDING WORK FILE
*                                                                           *
*****************************************************************************
*                                                                           *
*****************************************************************************
*                                                                           *
*************             VARIABLE IDENTIFICATION               *************
*                                                                           *
*      BNAME - BUILDING NAME
*      BNUMB - BUILDING NUMBER
*      BTYPE - STANDARD BUILDING TYPE
*      C11 -NET EXPLOSIVE WEIGHT (NEW) FOR MUNITION CLASS/DIVISION 1.1
*      C1204 - NEW FOR CLASS/DIVISION/CAT 1.2 04
*      C1208 - NEW FOR CLASS/DIVISION/CAT 1.2 08
*      C1212 - NEW FOR CLASS/DIVISION/CAT 1.2 12
*      C1218 - NEW FOR CLASS/DIVISION/CAT 1.2 18
*      C13 - NEW FOR CLASS/DIVISION 1.3
*      C14 - NEW FOR CLASS/DIVISION 1.4
*      KEY - INDEX KEY FOR MUNITION STORAGE AREA WORK FILE
*                                                                           *
****************               SUBROUTINE NAMES              ******************
*                                                                           *
*      CALLED BY:
*                    DELREC
*                    ITEM
*                                                                           *
*      CALLS:  NONE
*                                                                           *
*****************************************************************************
*                                                                           *
       COMMON /PT1/BNUMB,BNAME,BTYPE,C11,C1204,C1208,C1212,C1218,C13,C14
       COMMON /PT3/NUMB,TOTKEY,KEY
       CHARACTER BNAME*6,BNUMB*6,NUMB*6
       INTEGER BTYPE*2
       INTEGER *7 C11,C1204,C1208,C1212,C1218,C13,C14
       INTEGER TOTKEY,KEY
       WRITE (09'KEY) BTYPE,BNAME,BNUMB,C11,C1204,C1208,C1212,C1218,C13,C14
      RETURN
*                                                                           *
****************                 END RITE                    ******************
      END
```

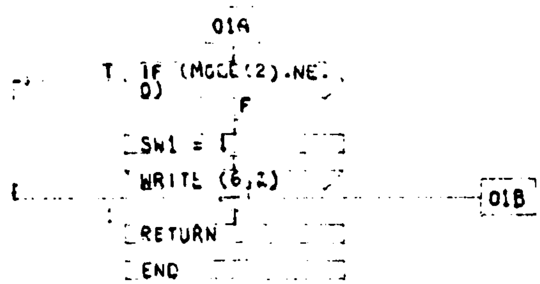
Appendix G

NSNDB Utility Program Flow Chart

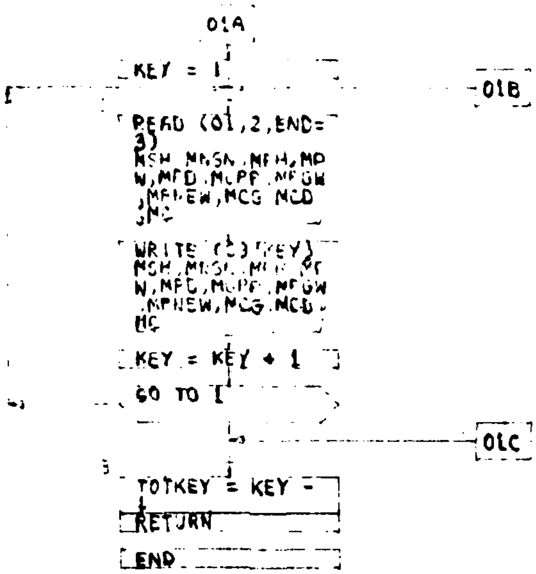
NSNDBUF

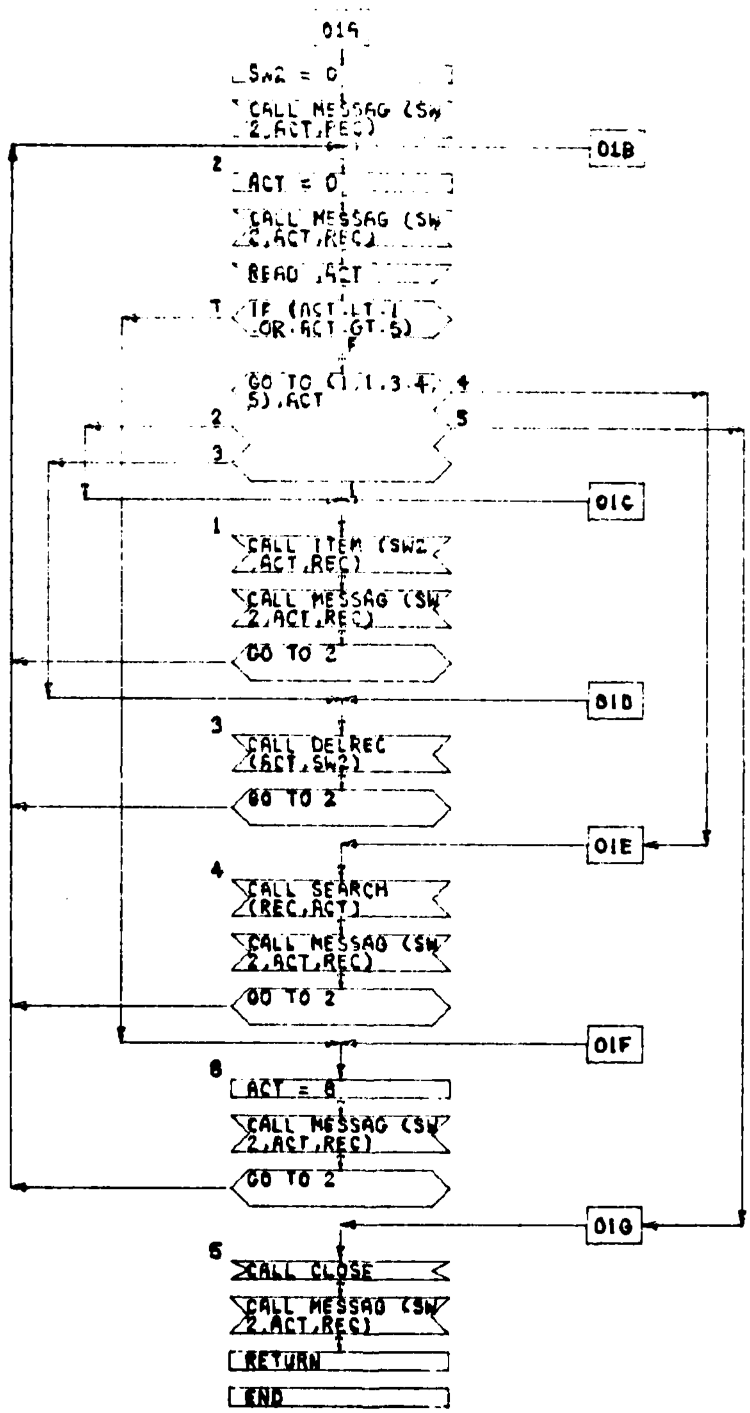


01A
CALL START (SW1
IF (SW1.NE.0)
CALL OPEN
CALL INITIAL
CALL ACTION
01B
STOP
END

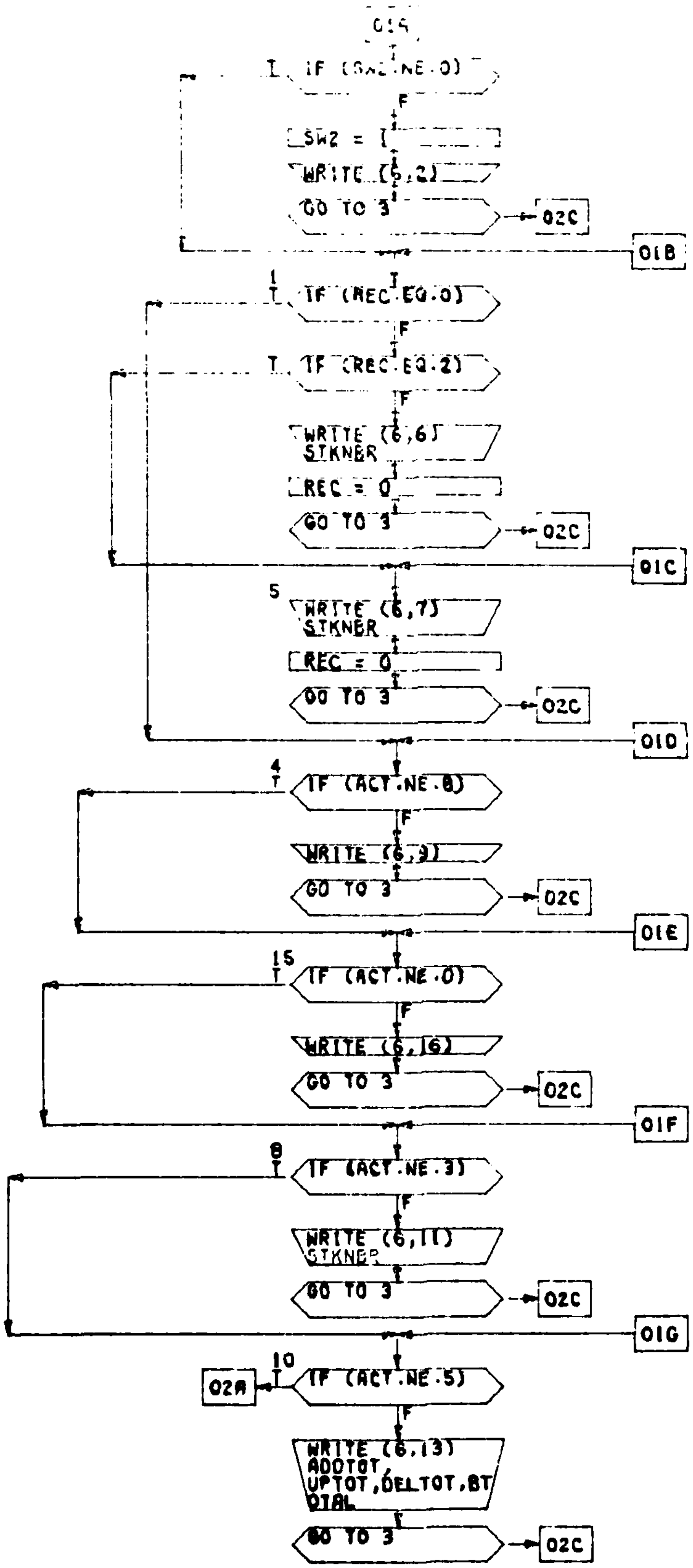

01A
T
IF (MODE(2).NE. 0)
F
SW1 = 1
WRITE (6,2)
RETURN
END
01B

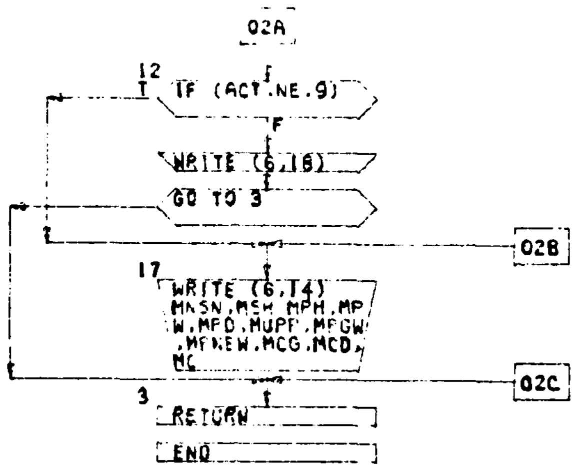
```
01A

CALL ATTACH (05
,'73CC6
/DATA/NSNDB

CALL CREATE (09
200,1,ISTAT2)

CALL RANSIZ (09
,20,0)

RETURN

END
```


01A
KEY = 1
01B
READ (01,2,END=3)
WRITE
KEY = KEY + 1
GO TO 1
01C
TOTKEY = KEY -
RETURN
END

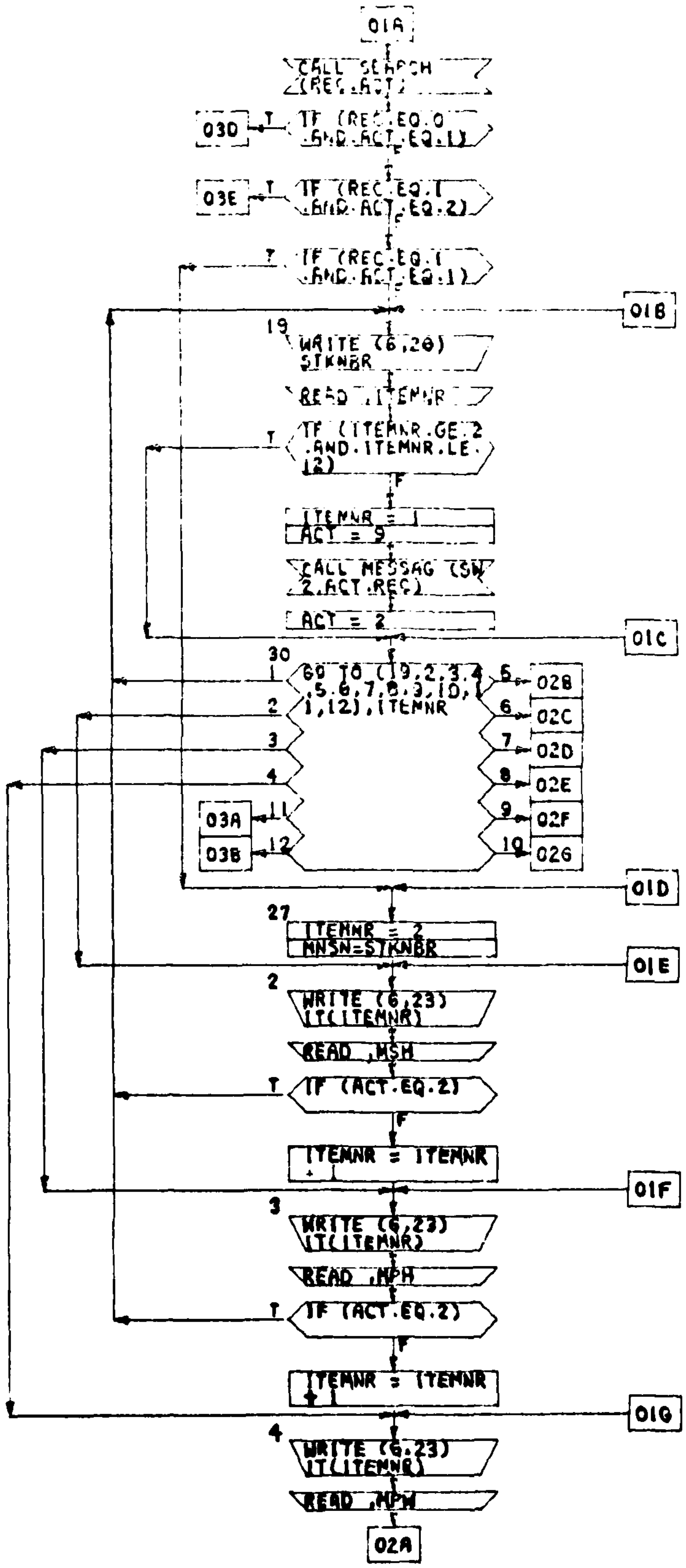


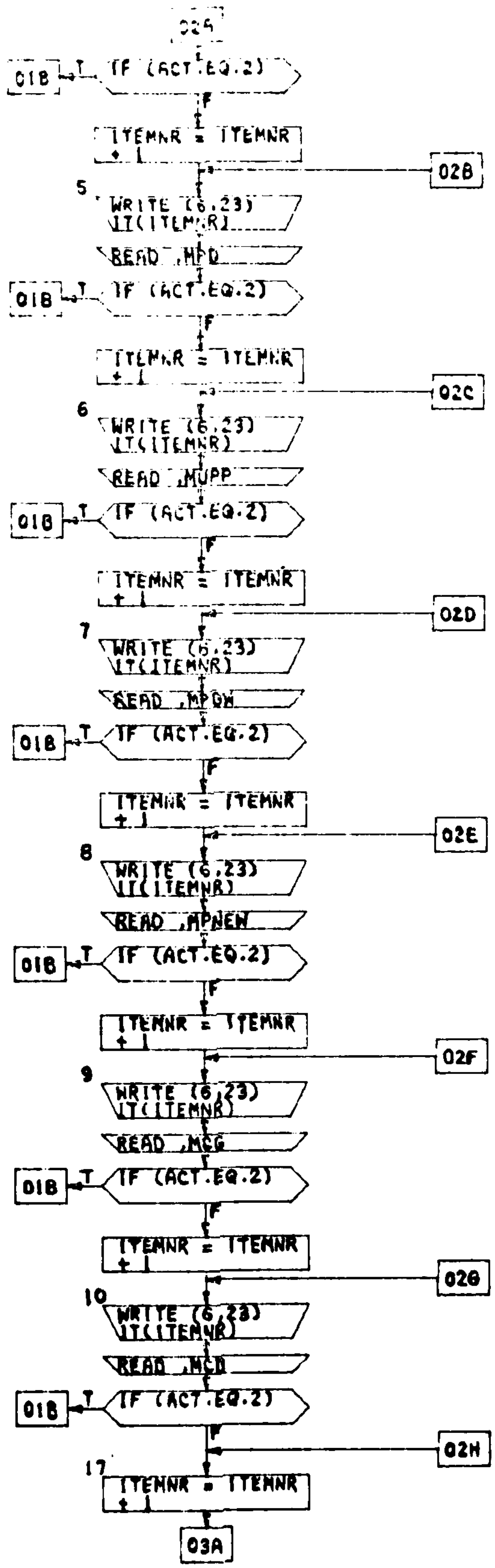
01A
SW2 = 0
CALL MESSAG (SW2,ACT,REC)
01B
2
ACT = 0
CALL MESSAG (SW2,ACT,REC)
READ ,ACT
IF (ACT.LT.1 .OR.ACT.GT.5)
T
F
GO TO (1,1,3,4,5),ACT
2
3
4
5
01C
1
CALL ITEM (SW2,ACT,REC)
CALL MESSAG (SW2,ACT,REC)
GO TO 2
01D
3
CALL DELREC (ACT,SW2)
GO TO 2
01E
4
CALL SEARCH (REC,ACT)
CALL MESSAG (SW2,ACT,REC)
GO TO 2
01F
8
ACT = 8
CALL MESSAG (SW2,ACT,REC)
GO TO 2
01G
5
CALL CLOSE
CALL MESSAG (SW2,ACT,REC)
RETURN
END


01A
IF (SW2.NE.0)
SW2 = 1
WRITE (6,2)
GO TO 3
02C
01B
1
IF (REC.EQ.0)
IF (REC.EQ.2)
WRITE (6,6)
STKNBR
REC = 0
GO TO 3
02C
01C
5
WRITE (6,7)
STKNBR
REC = 0
GO TO 3
02C
01D
4
IF (ACT.NE.8)
WRITE (6,9)
GO TO 3
02C
01E
15
IF (ACT.NE.0)
WRITE (6,16)
GO TO 3
02C
01F
8
IF (ACT.NE.3)
WRITE (6,11)
STKNBR
GO TO 3
02C
01G
10
02A
IF (ACT.NE.5)
WRITE (6,13)
ADDTOT,
UPTOT,DELTOT,BT
OTAL
GO TO 3
02C

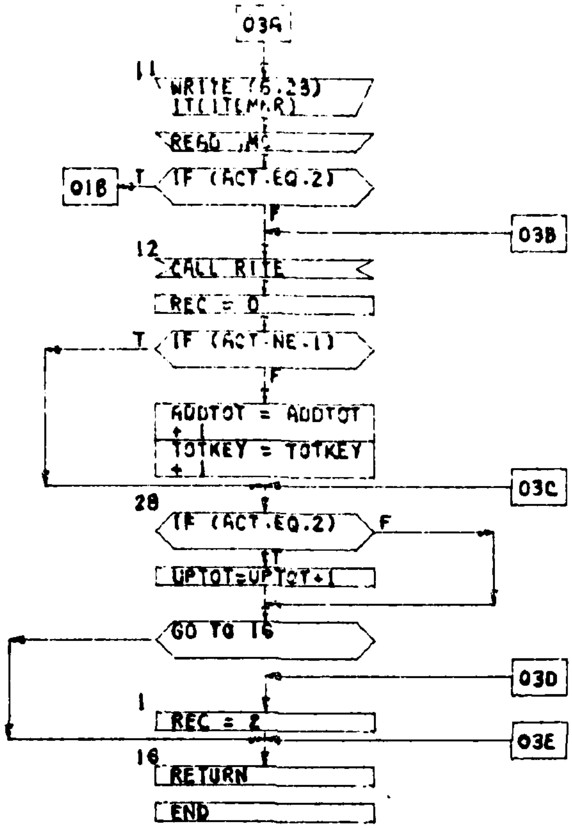

02A
12
T
IF (ACT.NE.9)
F
WRITE (6,18)
GO TO 3
02B
17
WRITE (6,14) MNSN,MSH,MPH,MPW,MPD,MUPP,MPGW,MPNEW,MCG,MCD,MC
02C
3
RETURN
END

SUBROUTINE ITEM

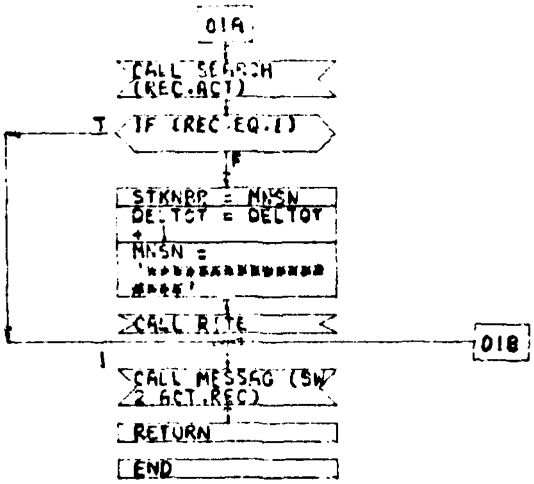



02A
01B
IF (ACT.EQ.2)
T
F
ITEMNR = ITEMNR + 1
02B
5
WRITE (6,23) IT(ITEMNR)
READ ,MPD
01B
IF (ACT.EQ.2)
ITEMNR = ITEMNR + 1
02C
6
WRITE (6,23) IT(ITEMNR)
READ ,MUPP
01B
IF (ACT.EQ.2)
ITEMNR = ITEMNR + 1
02D
7
WRITE (6,23) IT(ITEMNR)
READ ,MPOW
01B
IF (ACT.EQ.2)
ITEMNR = ITEMNR + 1
02E
8
WRITE (6,23) IT(ITEMNR)
READ ,MPNEW
01B
IF (ACT.EQ.2)
ITEMNR = ITEMNR + 1
02F
9
WRITE (6,23) IT(ITEMNR)
READ ,MCG
01B
IF (ACT.EQ.2)
ITEMNR = ITEMNR + 1
02G
10
WRITE (6,23) IT(ITEMNR)
READ ,MCD
01B
IF (ACT.EQ.2)
02H
17
ITEMNR = ITEMNR + 1
03A

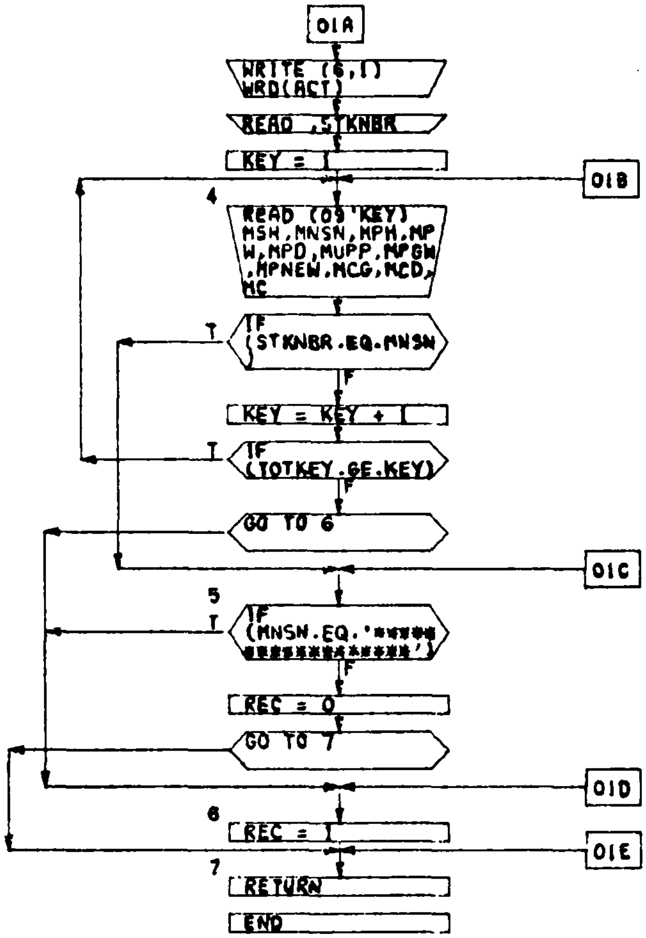
SUBROUTINE ITEM



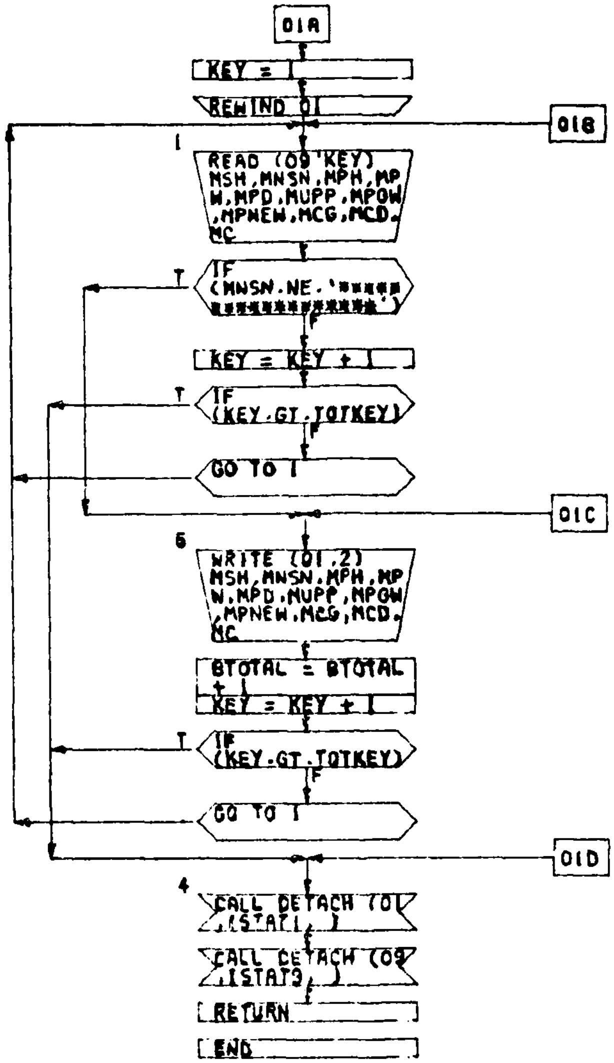
SUBROUTINE DELREC



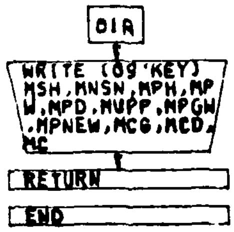
SUBROUTINE SEARCH



SUBROUTINE CLOSE



SUBROUTINE RITE



Appendix H

NSNDB Utility Program Source Listing

```
*#RUN * = /OBJECT/NSNDBUP (NOGO)
C       NATIONAL STOCK NUMBER DATA BASE UTILITY PROGRAM          23 NOV 79
C
C
C
C
*********************************************************************
*                                                                   *
*          NSNDBUP MAIN                                             *
*                                                                   *
*********************************************************************
*                                                                   *
*************            PROGRAM IDENTIFICATION          *************
*                                                                   *
*      NSNDBUP IS RESPONSIBLE FOR CREATING AND MAINTAINING RECORDS
*      IN THE NATIONAL STOCK NUMBER DATA BASE
*                                                                   *
*********************************************************************
*                                                                   *
C             ENTER A ONE DIGIT VALUE AND HIT CARRIAGE RETURN
C                 1 - ADD
C                 2 - CHANGE
C                 3 - DELETE
C                 4 - DISPLAY
C                 5 - TERMINATE
*                                                                   *
C             IF "1" (ADD RECORD)
C               FILL IN APPROPRIATE DATA ITEMS AS THEY ARE PRESENTED
C                            AND HIT THE CARRIAGE RETURN
*                                                                   *
C             IF "2" (CHANGE RECORD)
C               ENTER 18 DIGIT NATIONAL STOCK NUMBER OF RECORD TO BE CHANGED
C                   (REF AFTO 11A-1-46) AND HIT THE CARRIAGE RETURN
*                                                                   *
C               THEN ENTER TWO DIGIT ITEM NUMBER TO BE CHANGED
C                     AND HIT THE CARRIAGE RETURN
*                                                                   *
C                  ITEM  ITEM                           INPUT FORMAT
C                   NR
C                   01 - NATIONAL STOCK NUMBER          9999-99-999-9999AA
C                   02 - STACKING HEIGHT (# OF PACKAGES) 9999
C                   03 - PACKAGE HEIGHT (IN FEET)       999.9
C                   04 - PACKAGE WIDTH (IN FEET)        999.9
C                   05 - PACKAGE LENGTH (IN FEET)       999.9
C                   06 - UNITS PER PACKAGE              9999
C                   07 - GROSS WEIGHT (IN POUNDS)       99999.9999
C                   08 - NET EXPLOSIVE WEIGHT (IN POUNDS) 99999.9999
C                   09 - COMPATIBILITY GROUP            A
C                   10 - CLASS/DIVISION                 9.9
```

```
C                   11 - CATEGORY FOR 1.2                  99
*                                                                          *
C                   12 - FINISHED CURRENT TRANSACTION
*                                                                          *
C             IF "3" (DELETE RECORD)
C               ENTER 18 DIGIT STOCK NUMBER OF RECORD TO BE DELETED
C                    AND HIT THE CARRIAGE RETURN
*                                                                          *
C             IF "4" (DISPLAY RECORD)
C               ENTER 4 DIGIT STOCK NUMBER OF RECORD TO BE DISPLAYED
C                    AND HIT THE CARRIAGE RETURN
*                                                                          *
C             IF "5" TERMINATE THE EXECUTION OF THIS PROGRAM
*                                                                          *
***************************************************************************
*                                                                          *
*************              VARIABLE IDENTIFICATION              *************
*                                                                          *
*       SW1 - INDICATOR SWITCH FOR PROPER MODE OF OPERATION (ASCII)
*                                                                          *
****************                SUBROUTINE NAMES            *****************
*                                                                          *
*       CALLED BY:  NONE
*                                                                          *
*       CALLS:
*               START - VERIFIES PROPER MODE OF OPERATION (ASCII)
*               OPEN - OPENS NECESSARY FILES FOR THE PROGRAM
*               INITIAL - INITIALIZES THE NATIONAL STOCK NUMBER WORK FILE
*               ACTION - PERFORMS ALL THE REQUESTED ACTIONS ON THE DATA BASE
*                                                                          *
***************************************************************************
*                                                                          *
      COMMON /PT1/MNSN,MSH,MPH,MPW,MPD,MUPP,MPGW,MPNEW,MCG,MCD,MC
      COMMON /PT2/UPTOT,DELTOT,ADDTOT,BTOTAL
      COMMON /PT3/STKNBR,TOTKEY,KEY
      CHARACTER MNSN*18,STKNBR*18,MCG*1
      INTEGER MUPP*4,MC*2,MSH*4
      INTEGER SW1,UPTOT,DELTOT,ADDTOT,BTOTAL,TOTKEY,KEY
      REAL MPH,MPW,MPD,MPGW,MPNEW,MCD
      CALL START (SW1)
      IF (SW1.NE.0) GO TO 1
      CALL OPEN
      CALL INITIAL
      CALL ACTION
    1 STOP
      END
*                                                                          *
****************              END MAIN                      *****************
```

```
*************************************************************************
*                                                                       *
      SUBROUTINE START (SW1)
*                                                                       *
*************************************************************************
*                                                                       *
*************             PROGRAM IDENTIFICATION            *************
*                                                                       *
*     THIS ROUTINE VERIFIFS PROPER MODE OF OPERATION (ASCII)
*                                                                       *
*************************************************************************
*                                                                       *
*************************************************************************
*                                                                       *
*************             VARIABLE IDENTIFICATION           *************
*                                                                       *
*     MODE(2) - SYSTEM VARIABLE: 0 - BCD, 1 - ASCII
*     SW1 - INDICATOR SWITCH FOR PROPER MODE OF OPERATION (ASCII)
*                                                                       *
****************             SUBROUTINE NAMES            ****************
*                                                                       *
*     CALLED BY:  MAIN
*                                                                       *
*     CALLS:  NONE
*                                                                       *
*************************************************************************
*                                                                       *
       INTEGER SW1*1
       IF (MODE(2).NE.0) GO TO 1
       SW1 = 1
       WRITE (6,2)
*                                                                       *
    2  FORMAT (5X,"PLEASE RESTART USING 'RUN'")
*                                                                       *
    1 RETURN
      END
*                                                                       *
****************                END START                ****************
```

```
***************************************************************************
*                                                                         *
      SUBROUTINE OPEN
*                                                                         *
***************************************************************************
*                                                                         *
*************             PROGRAM IDENTIFICATION               *************
*                                                                         *
*       THIS ROUTINE INITIALIZES THE WORK FILE
*                                                                         *
***************************************************************************
*                                                                         *
***************************************************************************
*                                                                         *
*************             VARIABLE IDENTIFICATION              *************
*                                                                         *
*       ISTAT1 - FILE STATUS VARIABLE FOR PERMANENT FILE
*       ISTAT9 - FILE STATUS VARIABLE FOR WORK FILE
*                                                                         *
****************             SUBROUTINE NAMES              ******************
*                                                                         *
*       CALLED BY:  MAIN
*                                                                         *
*       CALLS:
*               ATTACH - OPENS PERMANENT FILE
*               CREATE - CREATES A TEMPORARY WORK FILE
*               RANSIZ - DEFINES THE RECORD SIZE FOR THE WORK FILE
*                                                                         *
***************************************************************************
*                                                                         *
***       CARD 999 WILL HAVE TO BE CHANGED FOR NEW USERS                ***
*                                                                         *
        CALL ATTACH (01,"79C06/DATA/NSNDB;",3,0,ISTAT1, )
        CALL CREATE (09,200,1,ISTAT2)
        CALL RANSIZ (09,20,0)
      RETURN
      END
*                                                                         *
****************                 END OPEN                  ******************
```

```
***************************************************************************
*                                                                         *
      SUBROUTINE INITIAL
*                                                                         *
***************************************************************************
*                                                                         *
*************            PROGRAM IDENTIFICATION              *************
*                                                                         *
*     THIS ROUTINE COPIES THE PERMANENT DATA BASE ON TO THE TEMPORARY
*     WORK FILE
*                                                                         *
***************************************************************************
*                                                                         *
***************************************************************************
*                                                                         *
*************            VARIABLE IDENTIFICATION             *************
*                                                                         *
*     KEY - INDEX KEY FOR NATIONAL STOCK NUMBER WORK FILE
*     MC - MUNITION CLASS/DIVISION
*     MCD - MUNITION CLASS/DIVISION
*     MCG - MUNITION COMPATIBILITY GROUP
*     MNSN - MUNITION NATIONAL STOCK NUMBER
*     MPD - MUNITION PACKAGE LENGTH
*     MPGW - MUNITION PACKAGE GROSS WEIGHT
*     MPH - MUNITION PACKAGE HEIGHT
*     MPNEW - MUNITION PACKAGE NET EXPLOSIVE WEIGHT
*     MPW - MUNITION PACKAGE WEIGHT
*     MSH - MUNITION STACKING HEIGHT (IN PACKAGES)
*     MUPP - NUMBER OF UNITS PER PACKAGE
*     TOTKEY - NUMBER OF RECORDS IN DATA BASE
*                                                                         *
****************               SUBROUTINE NAMES            ****************
*                                                                         *
*     CALLED BY:  MAIN
*                                                                         *
*     CALLS:  NONE
*                                                                         *
***************************************************************************
*                                                                         *
      COMMON /PT1/MNSN,MSH,MPH,MPW,MPD,MUPP,MPGW,MPNEW,MCG,MCD,MC
      COMMON /PT3/STKNBR,TOTKEY,KEY
      CHARACTER MNSN*18,STKNBR*18,MCG*1
      INTEGER MUPP*4,MC*2,MSH*4
      INTEGER TOTKEY,KEY
      REAL MPH,MPW,MPD,MPGW,MPNEW,MCD
      KEY = 1
    1 READ (01,2,END=3) MSH,MNSN,MPH,MPW,MPD,MUPP,MPGW,MPNEW,MCG,MCD,MC
      WRITE (09'KEY) MSH,MNSN,MPH,MPW,MPD,MUPP,MPGW,MPNEW,MCG,MCD,MC
      KEY = KEY + 1
      GO TO 1
    3 TOTKEY = KEY - 1
*                                                                         *
    2 FORMAT (I4,A18,3F5.1,I4,2F10.4,A1,F3.1,I2)
*                                                                         *
      RETURN
      END
*                                                                         *
****************                 END INITIAL               ****************
```

```
**************************************************************************
*                                                                        *
      SUBROUTINE ACTION
*                                                                        *
*************             PROGRAM IDENTIFICATION           *************
*                                                                        *
*       DRIVER ROUTINE THAT SELECTS THE APPROPRIATE ACTION TO BE
*       ACCOMPLISHED
*                                                                        *
**************************************************************************
*************             VARIABLE IDENTIFICATION          *************
*                                                                        *
*       ACT - TYPE OF ACTION (VALUE - 1 TO 5)
*       REC - CONTROL SWITCH: 0 - NOT FOUND, 1 - FOUND
*       SW2 - FIRST PASS SWITCH:  0 - FIRST PASS, 1 - SUBSEQUENT PASSES
*                                                                        *
****************              SUBROUTINE NAMES         ******************
*                                                                        *
*       CALLED BY:  MAIN
*                                                                        *
*       CALLS:
*               MESSAG - PRINTS APPROPRIATE MESSAGE
*               ITEM - ADDS A NEW RECORD OR CHANGES AN EXISTING RECORD
*               DELREC - DELETES SPECIFIED RECORD FROM DATA BASE
*               SEARCH - SEARCHES THE WORK FILE FOR SPECIFIED RECORD
*               CLOSE - TERMINATES THE PROGRAM
*                                                                        *
**************************************************************************
*                                                                        *
        INTEGER ACT*1,SW2*1,REC*1
        SW2 = 0
        CALL MESSAG (SW2,ACT,REC)
    2   ACT = 0
        CALL MESSAG (SW2,ACT,REC)
        READ ,ACT
        IF (ACT.LT.1.OR.ACT.GT.5) GO TO 8
        GO TO (1,1,3,4,5),ACT
    1   CALL ITEM (SW2,ACT,REC)
        CALL MESSAG (SW2,ACT,REC)
        GO TO 2
    3   CALL DELREC (ACT,SW2)
        GO TO 2
    4   CALL SEARCH (REC,ACT)
        CALL MESSAG (SW2,ACT,REC)
        GO TO 2
    8   ACT = 8
        CALL MESSAG (SW2,ACT,REC)
        GO TO 2
    5   CALL CLOSE
        CALL MESSAG (SW2,ACT,REC)
*                                                                        *
    7   FORMAT (I1)
*                                                                        *
      RETURN
      END
*                                                                        *
****************              END ACTION               ******************
```

```
****************************************************************************
*                                                                          *
      SUBROUTINE MESSAG (SW2,ACT,REC)
*                                                                          *
****************************************************************************
*                                                                          *
*************             PROGRAM IDENTIFICATION              *************
*                                                                          *
*      THIS ROUTINE PRINTS THE APPROPRIATE MESSAGES
*                                                                          *
****************************************************************************
*                                                                          *
****************************************************************************
*                                                                          *
*************             VARIABLE IDENTIFICATION             *************
*                                                                          *
*      ACT - TYPE OF ACTION BEING PERFORMED
*      ADDTOT - NUMBER OF RECORDS ADDED TO DATA BASE
*      BTOTAL - NUMBER OF RECORDS IN DATA BASE
*      DELTOT - NUMBER OF RECORDS DELETED FROM DATA BASE
*      MC - MUNITION CLASS/DIVISION
*      MCD - MUNITION CLASS/DIVISION
*      MCG - MUNITION COMPATIBILITY GROUP
*      MNSN - MUNITION NATIONAL STOCK NUMBER
*      MPD - MUNITION PACKAGE LENGTH
*      MPGW - MUNITION PACKAGE GROSS WEIGHT
*      MPH - MUNITION PACKAGE HEIGHT
*      MPNEW - MUNITION PACKAGE NET EXPLOSIVE WEIGHT
*      MPW - MUNITION PACKAGE WEIGHT
*      MSH - MUNITION STACKING HEIGHT (IN PACKAGES)
*      MUPP - NUMBER OF UNITS PER PACKAGE
*      REC - FOUND/NOT FOUND SWITCH
*      STKNBR - MUNITION NATIONAL STOCK NUMBER
*      SW2 - FIRST PASS SWITCH:  0 - FIRST PASS, 1 - SUBSEQUENT PASSES
*      UPTOT - NUMBER OF RECORDS UPDATED
*                                                                          *
****************              SUBROUTINE NAMES           ******************
*                                                                          *
*      CALLED BY:
*                  ACTION
*                  ITEM
*                  DELREC
*                                                                          *
*      CALLS:  NONE
*                                                                          *
****************************************************************************
*                                                                          *
       COMMON /PT1/MNSN,MSH,MPH,MPW,MPD,MUPP,MPGW,MPNEW,MCG,MCD,MC
       COMMON /PT2/UPTOT,DELTOT,ADDTOT,BTOTAL
       COMMON /PT3/STKNBR,TOTKEY,KEY
       CHARACTER MNSN*18,STKNBR*18,MCG*1
       INTEGER MUPP*4,MC*2,MSH*4
       INTEGER ACT*1,SW2*1,REC*1,UPTOT,DELTOT,ADDTOT,BTOTAL
       REAL MPH,MPW,MPD,MPGW,MPNEW,MCD
       IF (SW2.NE.0) GO TO 1
       SW2 = 1
       WRITE (6,2)
       GO TO 3
```

```
 1    IF (REC.EQ.0) GO TO 4
      IF (REC.EQ.2) GO TO 5
      WRITE (6,6) STKNBR
      REC = 0
      GO TO 3
 5    WRITE (6,7) STKNBR
      REC = 0
      GO TO 3
 4    IF (ACT.NE.8) GO TO 15
      WRITE (6,9)
      GO TO 3
15    IF (ACT.NE.0) GO TO 8
      WRITE (6,16)
      GO TO 3
 8    IF (ACT.NE.3) GO TO 10
      WRITE (6,11) STKNBR
      GO TO 3
10    IF (ACT.NE.5) GO TO 12
      WRITE (6,13) ADDTOT, UPTOT,DELTOT,BTOTAL
      GO TO 3
12    IF (ACT.NE.9) GO TO 17
      WRITE (6,18)
      GO TO 3
17    WRITE (6,14) MNSN,MSH,MPH,MPW,MPD,MUPP,MPGW,MPNEW,MCG,MCD,MC
*                                                                      *
 2    FORMAT (//5X,"WELCOME TO THE MUNITION NATIONAL STOCK NUMBER ",
     &  "DATA BASE"//)
*                                                                      *
 9    FORMAT (5X,"OPTIONS:",
     &  /25X,"1 - ADD RECORD"/,25X,"2 - CHANGE RECORD"/,25X,
     &  "3 - DELETE RECORD"/,25X,"4 - DISPLAY RECORD"/,25X,
     &  "5 - TERMINATE JOB"//)
*                                                                      *
 6    FORMAT (5X,"RECORD ",A18," DOES NOT EXIST"//)
*                                                                      *
 7    FORMAT (5X,"RECORD ",A18," ALREADY EXISTS"//)
*                                                                      *
11    FORMAT (5X,"RECORD ",A18," HAS BEEN DELETED FROM THE DATA BASE"//)
*                                                                      *
13    FORMAT (5X,"YOU ARE NOW EXITING THE UPDATE PROGRAM"//
     &    15X,"ADDED - ",I3/,15X,"CHANGED - ",I3/,15X,"DELETED - ",
     &    I3/,15X,"TOTAL NUMBER OF RECORDS IN DATA BASE - ",I3//)
*                                                                      *
14    FORMAT (5X,"NATIONAL STK NR - ",A18/,10X,"STACKING HEIGHT - ",I4,
     & " PACKAGES",/10X,
     & "PACKAGE HEIGHT - ",F5.1," FT"/,10X,"PACKAGE WIDTH - ",
     & F5.1," FT"/,10X,"PACKAGE LENGTH - ",F5.1," FT"/,10X,
     &  "UNITS PER PACKAGE - ",I4/
     &   ,10X,"PACKAGE GROSS WT - ",F10.4, " LBS",/10X,"PACKAGE NEW - ",
     &  F10.4," LBS",/10X,"COMPATIBILITY GROUP - ",A1/,10X,"CLASS/DIVI",
     &  "SION - ",F3.1/,10X,"CATEGORY - ",I2//)
*                                                                      *
16    FORMAT (5X,"ENTER THE ONE DIGIT TRANSACTION DESIRED:"/)
*                                                                      *
18    FORMAT (5X,"OPTIONS:",/10X,
     &  "02 - STACKING HEIGHT (9999) IN PACKAGES",/10X,
     &  "03 - PACKAGE HEIGHT (999.9) IN FT",/10X,
     &  "04 - PACKAGE WIDTH (999.9) IN FT",/10X,
```

```
      &     "05 - PACKAGE LENGTH (999.9) IN FT",/10X,
      &     "06 - UNITS PER PACKAGE (9999)",/10X,
      &     "07 - PACKAGE GROSS WT (99999.9999) IN LBS",/10X,
      &     "08 - PACKAGE NEW (99999.9999) IN LBS",/10X,
      &     "09 - COMPATIBILITY GROUP (A)",/10X,
      &     "10 - CLASS/DIVISION (9.9)",/10X,
      &     "11 - CATEGORY (99)",//10X,
      &     "12 - FINISHED WITH THIS RECORD"//)
*                                                                          *
    3 RETURN
      END
*                                                                          *
*****************                END MESSAG                *******************
```

```
*************************************************************************
*                                                                       *
      SUBROUTINE ITEM (SW2,ACT,REC)
*                                                                       *
*************************************************************************
*                                                                       *
*************             PROGRAM IDENTIFICATION             *************
*                                                                       *
*      THIS ROUTINE WILL EITHER CREATE A NEW RECORD OR UPDATE
*      SPECIFIED ITEMS OF AN EXISTING RECORD
*                                                                       *
*************************************************************************
*                                                                       *
*************************************************************************
*                                                                       *
*************             VARIABLE IDENTIFICATION            *************
*                                                                       *
*      ACT - TYPE OF ACTION IN PROGRESS
*      ADDTOT - NUMBER OF RECORDS ADDED TO DATA BASE
*      IT - ARRAY CONTAINING ITEM NAMES TO BE PROCESSED
*      ITEMNR - INDEX FOR IT ARRAY
*      MC - MUNITION CLASS/DIVISION
*      MCD - MUNITION CLASS/DIVISION
*      MCG - MUNITION COMPATIBILITY GROUP
*      MNSN - MUNITION NATIONAL STOCK NUMBER
*      MPD - MUNITION PACKAGE LENGTH
*      MPGW - MUNITION PACKAGE GROSS WEIGHT
*      MPH - MUNITION PACKAGE HEIGHT
*      MPNEW - MUNITION PACKAGE NET EXPLOSIVE WEIGHT
*      MPW - MUNITION PACKAGE WEIGHT
*      MSH - MUNITION STACKING HEIGHT (IN PACKAGES)
*      MUPP - NUMBER OF UNITS PER PACKAGE
*      REC - FOUND/NOT FOUND SWITCH
*      STKNBR - MUNITION NATIONAL STOCK NUMBER
*      TOTKEY - NUMBER OF RECORDS IN DATA BASE
*      UPTOT - NUMBER OF RECORDS UPDATED
*                                                                       *
****************              SUBROUTINE NAMES           *****************
*                                                                       *
*      CALLED BY:  ACTION
*                                                                       *
*      CALLS:
*              SEARCH - SEARCHES WORK FILE FOR SPECIFIED RECORD
*              MESSAG - PRINTS SPECIFIED MESSAGES
*              RITE - WRITES SPECIFIED RECORD
*                                                                       *
*************************************************************************
*                                                                       *
       COMMON /PT1/MNSN,MSH,MPH,MPW,MPD,MUPP,MPGW,MPNEW,MCG,MCD,MC
       COMMON /PT2/UPTOT,DELTOT,ADDTOT,BTOTAL
       COMMON /PT3/STKNBR,TOTKEY,KEY
       CHARACTER MNSN*18,STKNBR*18,MCG*1
       INTEGER MUPP*4,MC*2,MSH*4
       INTEGER ACT*1,ITEMNR*2,REC*1,ADDTOT,UPTOT,TOTKEY,KEY
       REAL MPH,MPW,MPD,MPGW,MPNEW,MCD
       CHARACTER IT*35(11)/"STOCK NR (9999-99-999-9999AA)",
     &   "STACK HEIGHT (9999) IN PACKS","PACKAGE HEIGHT (999.9) IN FT",
     &   "PACKAGE WIDTH (999.9) IN FT","PACKAGE LENGTH (999.9) IN FT",
```

```
     &   "UNITS PER PACK (9999)  ","PACKAGE GR WT (99999.9999) IN LBS",
     &   "PACKAGE NEW (99999.9999) IN LBS","COMPATBL GROUP (A)",
     &   "CLASS/DIVISION (9.9)","CATEGORY (99)"/
      CALL SEARCH (REC,ACT)
      IF (REC EQ.O.AND.ACT.EQ.1) GO TO 1
      IF (REC.EQ.1.AND.ACT.EQ.2) GO TO 16
      IF (REC.EQ 1.AND.ACT.EQ.1) GO TO 27
   19 WRITE (6,26) STKNBR
      READ ,ITEMNR
      IF (ITEMNR.GE.2.AND.ITEMNR.LE.12)GO TO 30
      ITEMNR = 1
      ACT = 9
      CALL MESSAG (SW2,ACT,REC)
      ACT = 2
   30 GO TO (19,2,3,4,5,6,7,8,9,10,11,12),ITEMNR
   27 ITEMNR = 2
      MNSN=STKNBR
    2 WRITE (6,23) IT(ITEMNR)
      READ ,MSH
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
    3 WRITE (6,23) IT(ITEMNR)
      READ ,MPH
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
    4 WRITE (6,23) IT(ITEMNR)
      READ ,MPW
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
    5 WRITE (6,23) IT(ITEMNR)
      READ ,MPD
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
    6 WRITE (6,23) IT(ITEMNR)
      READ ,MUPP
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
    7 WRITE (6,23) IT(ITEMNR)
      READ ,MPGW
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
    8 WRITE (6,23) IT(ITEMNR)
      READ ,MPNEW
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
    9 WRITE (6,23) IT(ITEMNR)
      READ ,MCG
      IF (ACT.EQ.2) GO TO 19
      ITEMNR = ITEMNR + 1
   10 WRITE (6,23) IT(ITEMNR)
      READ ,MCD
      IF (ACT.EQ.2) GO TO 19
   17 ITEMNR = ITEMNR + 1
   11 WRITE (6,23) IT(ITEMNR)
      READ ,MC
      IF (ACT.EQ.2) GO TO 19
   12 CALL RITE
      REC = 0
```

```
      IF (ACT.NE.1) GO TO 28
      ADDTOT = ADDTOT + 1
      TOTKEY = TOTKEY + 1
 28   IF (ACT.EQ.2) UPTOT=UPTOT+1
      GO TO 16
  1   REC = 2
*                                                                  *
 23   FORMAT (5X,"ENTER THE INFORMATION FOR THE ",A35)
*                                                                  *
 26   FORMAT (5X,"ENTER THE 2 DIGIT ITEM NUMBER OF THE ITEM TO BE ",
   &     /10X,"CHANGED FOR STOCK NR ",A18//)
*                                                                  *
 16 RETURN
    END
*                                                                  *
*****************            END ITEM            ******************
```

```
*************************************************************************
*                                                                       *
      SUBROUTINE DELREC (ACT,SW2)
*                                                                       *
*************************************************************************
*                                                                       *
*************             PROGRAM IDENTIFICATION            *************
*                                                                       *
*       THIS ROUTINE WILL DELETE THE SPECIFIED EXISTING RECORD
*       FROM THE DATA BASE
*                                                                       *
*************************************************************************
*                                                                       *
*************************************************************************
*                                                                       *
*************             VARIABLE IDENTIFICATION           *************
*                                                                       *
*        ACT - TYPE OF ACTION IN PROGRESS
*        DELTOT - NUMBER OF RECORDS DELETED FROM DATA BASE
*        MNSN - MUNITION NATIONAL STOCK NUMBER
*        REC - FOUND/NOT FOUND SWITCH
*        STKNBR - MUNITION NATIONAL STOCK NUMBER
*        SW2 - FIRST PASS SWITCH: 0 - FIRST PASS, 1 - SUBSEQUENT PASSES
*                                                                       *
****************              SUBROUTINE NAMES           ****************
*                                                                       *
*       CALLED BY:  ACTION
*                                                                       *
*       CALLS:
*               SEARCH - SEARCHES WORK FILE FOR SPECIFIED RECORD
*               RITE - WRITES DELETED IDENTIFIER FOR SPECIFIED RECORD
*               MESSAG - PRINTS SPECIFIED MESSAGE
*                                                                       *
*************************************************************************
*                                                                       *
        COMMON /PT1/MNSN,MSH,MPH,MPW,MPD MUPP,MPGW,MPNEW,MCG,MCD,MC
        COMMON /PT2/UPTOT,DELTOT,ADDTOT,BTOTAL
        COMMON /PT3/STKNBR,TOTKEY,KEY
        CHARACTER MNSN*18,STKNBR*18,MCG*1
        INTEGER MUPP*4,MC*2,MSH*4
        INTEGER SW2*1,ACT*1,REC*1,DELTOT,TOTKEY,KEY
        REAL MPH,MPW,MPD,MPGW,MPNEW,MCD
        CALL SEARCH (REC,ACT)
        IF (REC.EQ.1) GO TO 1
        STKNBR = MNSN
        DELTOT = DELTOT + 1
        MNSN = "******************"
        CALL RITE
    1   CALL MESSAG (SW2 ACT,REC)
      RETURN
      END
*                                                                       *
****************                 END DELREC              ****************
```

```
**************************************************************************
*                                                                        *
      SUBROUTINE SEARCH (REC,ACT)
*                                                                        *
**************************************************************************
*                                                                        *
*************             PROGRAM IDENTIFICATION             *************
*                                                                        *
*     THIS ROUTINE SEARCHES THE WORK FILE FOR THE SPECIFIED RECORD
*                                                                        *
**************************************************************************
*                                                                        *
**************************************************************************
*                                                                        *
*************             VARIABLE IDENTIFICATION            *************
*                                                                        *
*     ACT - TYPE OF ACTION IN PROGRESS
*     KEY - INDEX KEY FOR NATIONAL STOCK NUMBER WORK FILE
*     MC - MUNITION CLASS/DIVISION
*     MCD - MUNITION CLASS/DIVISION
*     MCG - MUNITION COMPATIBILITY GROUP
*     MNSN - MUNITION NATIONAL STOCK NUMBER
*     MPD - MUNITION PACKAGE LENGTH
*     MPGW - MUNITION PACKAGE GROSS WEIGHT
*     MPH - MUNITION PACKAGE HEIGHT
*     MPNEW - MUNITION PACKAGE NET EXPLOSIVE WEIGHT
*     MPW - MUNITION PACKAGE WEIGHT
*     MSH - MUNITION STACKING HEIGHT (IN PACKAGES)
*     MUPP - NUMBER OF UNITS PER PACKAGE
*     REC - FOUND/NOT FOUND SWITCH
*     STKNBR - MUNITION NATIONAL STOCK NUMBER
*     TOTKEY - NUMBER OF RECORDS IN DATA BASE
*     WRD - ARRAY CONTAINING TYPE OF ACTION IN PROGRESS
*                                                                        *
****************                SUBROUTINE NAMES          ****************
*                                                                        *
*     CALLED BY:
*                 ACTION
*                 ITEM
*                 DELREC
*                                                                        *
*     CALLS:  NONE
*                                                                        *
**************************************************************************
*                                                                        *
       COMMON /PT1/MNSN,MSH,MPH,MPW,MPD,MUPP,MPGW,MPNEW,MCG,MCD,MC
       COMMON /PT3/STKNBR,TOTKEY,KEY
       CHARACTER MNSN*18,STKNBR*18,MCG*1
       INTEGER MUPP*4,MC*2,MSH*4
       INTEGER REC*1,TOTKEY,KEY,ACT*1
       REAL MPH,MPW,MPD,MPGW,MPNEW,MCD
       CHARACTER WRD*9(4)/"ADDED ","CHANGED","DELETED","DISPLAYED"/
       WRITE (6,1) WRD(ACT)
       READ ,STKNBR
       KEY = 1
    4  READ (09'KEY) MSH,MNSN,MPH,MPW,MPD,MUPP,MPGW,MPNEW,MCG,MCD,MC
       IF (STKNBR.EQ.MNSN) GO TO 5
       KEY = KEY + 1
```

```
      IF (TOTKEY.GE.KEY) GO TO 4
      GO TO 6
5     IF (MNSN.EQ."*****************") GO TO 6
      REC = 0
      GO TO 7
*                                                                      *
1     FORMAT (5X,"ENTER THE 18 CHARACTER STOCK NUMBER OF RECORD TO BE ",A9)
*                                                                      *
6     REC = 1
7   RETURN
    END
*                                                                      *
****************                END SEARCH              ******************
```

```
***********************************************************************
*                                                                     *
      SUBROUTINE CLOSE
*                                                                     *
***********************************************************************
*                                                                     *
*************             PROGRAM IDENTIFICATION            *************
*                                                                     *
*       THIS ROUTINE WRITES THE UPDATED DATA BASE BACK TO THE
*       PERMANENT FILE
*                                                                     *
***********************************************************************
*                                                                     *
***********************************************************************
*                                                                     *
*************             VARIABLE IDENTIFICATION           *************
*                                                                     *
*       BTOTAL - NUMBER OF RECORDS IN DATA BASE
*       KEY - INDEX KEY FOR NATIONAL STOCK NUMBER WORK FILE
*       MC - MUNITION CLASS/DIVISION
*       MCD - MUNITION CLASS/DIVISION
*       MCG - MUNITION COMPATIBILITY GROUP
*       MNSN - MUNITION NATIONAL STOCK NUMBER
*       MPD - MUNITION PACKAGE LENGTH
*       MPGW - MUNITION PACKAGE GROSS WEIGHT
*       MPH - MUNITION PACKAGE HEIGHT
*       MPNEW - MUNITION PACKAGE NET EXPLOSIVE WEIGHT
*       MPW - MUNITION PACKAGE WEIGHT
*       MSH - MUNITION STACKING HEIGHT (IN PACKAGES)
*       MUPP - NUMBER OF UNITS PER PACKAGE
*       TOTKEY - NUMBER OF RECORDS IN DATA BASE
*                                                                     *
****************               SUBROUTINE NAMES            *****************
*                                                                     *
*       CALLED BY:  ACTION
*                                                                     *
*       CALLS:  DETACH - RELEASES SPECIFIED FILE FROM PROGRAM CONTROL
*                                                                     *
***********************************************************************
*                                                                     *
        COMMON /PT1/MNSN,MSH,MPH,MPW,MPD,MUPP,MPGW,MPNEW,MCG,MCD,MC
        COMMON /PT2/UPTOT,DELTOT,ADDTOT,BTOTAL
        COMMON /PT3/STKNBR,TOTKEY,KEY
        CHARACTER MNSN*18,STKNBR*18,MCG*1
        INTEGER MUPP*4,MC*2,MSH*4
        INTEGER BTOTAL,TOTKEY,KEY
        REAL MPH,MPW,MPD,MPGW,MPNEW,MCD
        KEY = 1
        REWIND 01
    1   READ (09'KEY) MSH,MNSN,MPH,MPW,MPD,MUPP,MPGW,MPNEW,MCG,MCD,MC
        IF (MNSN.NE."******************") GO TO 5
        KEY = KEY + 1
        IF (KEY.GT.TOTKEY) GO TO 4
        GO TO 1
    5   WRITE (01,2) MSH,MNSN,MPH,MPW,MPD,MUPP,MPGW,MPNEW,MCG,MCD,MC
        BTOTAL = BTOTAL + 1
        KEY = KEY + 1
        IF (KEY.GT.TOTKEY) GO TO 4
```

```
        GO TO 1
*                                                                      *
  2     FORMAT (I4,A18,3F5.1,I4,2F10.4,A1,F3.1,I2)
*                                                                      *
  4     CALL DETACH (01,ISTAT1, )
        CALL DETACH (09,ISTAT9, )
      RETURN
      END
*                                                                      *
****************              END CLOSE            ******************
```

```
************************************************************************
*                                                                      *
      SUBROUTINE RITE
*                                                                      *
************************************************************************
*                                                                      *
*************              PROGRAM IDENTIFICATION          *************
*                                                                      *
*      THIS ROUTINE WRITES A RECORD TO THE NATIONAL STOCK NUMBER
*      WORK FILE
*                                                                      *
************************************************************************
*                                                                      *
************************************************************************
*                                                                      *
*************              VARIABLE IDENTIFICATION         *************
*                                                                      *
*      KEY - INDEX KEY FOR NATIONAL STOCK NUMBER WORK FILE
*      MC - MUNITION CLASS/DIVISION
*      MCD - MUNITION CLASS/DIVISION
*      MCG - MUNITION COMPATIBILITY GROUP
*      MNSN - MUNITION NATIONAL STOCK NUMBER
*      MPD - MUNITION PACKAGE LENGTH
*      MPGW - MUNITION PACKAGE GROSS WEIGHT
*      MPH - MUNITION PACKAGE HEIGHT
*      MPNEW - MUNITION PACKAGE NET EXPLOSIVE WEIGHT
*      MPW - MUNITION PACKAGE WEIGHT
*      MSH - MUNITION STACKING HEIGHT (IN PACKAGES)
*      MUPP - NUMBER OF UNITS PER PACKAGE
*                                                                      *
***************               SUBROUTINE NAMES         *****************
*                                                                      *
*      CALLED BY:
*                    DELREC
*                    ITEM
*                                                                      *
*      CALLS:  NONE
*                                                                      *
************************************************************************
*                                                                      *
       COMMON /PT1/MNSN,MSH,MPH,MPW,MPD,MUPP,MPGW,MPNEW,MCG,MCD,MC
       COMMON /PT3/STKNBR,TOTKEY,KEY
       CHARACTER MNSN*18,STKNBR*18,MCG*1
       INTEGER MUPP*4,MC*2,MSH*4
       INTEGER TOTKEY,KEY
       REAL MPH,MPW,MPD,MPGW,MPNEW,MCD
       WRITE (09'KEY) MSH,MNSN,MPH,MPW,MPD,MUPP,MPGW,MPNEW,MCG,MCD,MC
      RETURN
*                                                                      *
***************                END RITE                *****************
      END
```

form
erat nd
te t t file
rcise amming age t lem"
se

the n i II con
erwise
n NS , th ss ref (CRSR
the file INFO)"
ate ing, L wo
d in vent numbe ach
num for lot"
er ento compa DB re
ensi or f
h n pa
rmat the file
sto ildi rom
ca vol of bu ng,
eigh W) a for
to ldi
g munit bui , wo
era fun th itio
bu str and
FO"
ce
erat nea amm
wn mmi

ENI

```
LEVEL 3

  LPGEN      MAIN

    PERFORM  START
    IF  "error switch-1 is not on"
      THEN
        PERFORM  OPEN
        PERFORM  MUNINV
        IF  "error switch-2 is not on"
          THEN
            PERFORM  STORE
            IF  "error switch-2 is not on"
              THEN
                PERFORM  FORM
                IF  "error switch-2 is not on"
                  THEN
                    PERFORM  CLOSE
                    IF  "error switch-2 is not on"
                      THEN
                        PERFORM  SPAWN
                      ELSE
                    ENDIF
                  ELSE
                ENDIF
              ELSE
            ENDIF
          ELSE
        ENDIF
      ELSE
    ENDIF

  END        LPGEN


LEVEL 3

  START      ROUTINE

    IF  "mode is ASCII"
      THEN  "print welcome message"
      ELSE  "turn on error switch-1 and
             print restart message"
    ENDIF

  END        START
```

```
LEVEL 3

  OPEN      ROUTINE

    "open files NSNDB,MSADB,SBDB,CRSREF, and LPINFO"
    "create munition, building, decision variable, and JCL work files"

  END        OPEN

LEVEL 3

  MUNINV      ROUTINE

    "read in the national stock number of munition
     to be entered into inventory"
    "search NSNDB to see if this stock number is valid"
    "if invalid print an error message "
    "if valid read in number of lots for this munition"
    "then read in number of packages for each lot"
    "retrieve necessary munition information for current stock
     number from NSNDB"
    "calculate the package volume and density factor (NEW/volume)
    "write information to munition work file for each
     lot of current munition"
    "repeat for remaining munitions to be entered into inventory"
    "if invalid stock numbers were found ask if user
     desires to stop execution"
    close "NSNDB"

  END        MUNINV

LEVEL 3

  STORE      ROUTINE

    "read in a record from MSADB"
    "search SBDB for a matching TYPE record"
    "if a match is not found turn on error switch-2"
    "if a match is found then:
     calculate the usable building volume; and,
     write a record to the building work file containing
     this information"
    "repeat for all buildings in MSADB"
    "close SBDB and MSADB"

  END        STORE
```

```
LEVEL 3

  FORM      ROUTINE

    "determine the objective function; number of constraints
     based on the number of munitions, number of storage buildings,
     and the number of different compatibility group/class combinations;
     and the number of decision variables
    "format the objective function, constraints, right hand side,
     and any other information needed by the LP/600 package"
    "write this information to LPINFO"

  END       FORM

LEVEL 3

  CLOSE     ROUTINE

    "print the cross reference lists"
    "close all open files except JCL work file"

  END       CLOSE

LEVEL 3

  SPAWN     ROUTINE

    "determine USERID and PROBLEM NUMBER"
    "calculate core and time limits"
    "generate the JCL cards"
    "spawn the LP/600 job"
    "close the JCL work file"

  END       SPAWN
```

LEVEL 4

```
MUNINV      ROUTINE

  PRINT  "message telling user to enter first stock number"
  READ  "first stock number for munition inventory"
  DO WHILE  "munition transactions are not complete"
    REWIND NSNDB (file code 01)
    READ  "first record from NSNDB"
    DO WHILE  "not end of NSNDB"
      IF  "a match is found"
        THEN  "turn error switch-2 off and exit this DO loop"
        ELSE  "turn error switch-2 on and read next record"
      ENDIF
    ENDDO
    IF  "error switch-2 is on (not found)"
      THEN  PRINT  "error message as warning using stock number"
      ELSE  PRINT  "message asking for number of lots
                    for this munition"
            READ  "number of lots"
            DO WHILE  "current number of lots not processed"
              PRINT  "message asking for number of packages"
              READ  "number of packages for current lot"
            ENDDO
            "calculate the volume of package"
            "calculate the density factor of this munition
             mlf = NEW / munition volume"
            DO WHILE  "all lots have not been processed"
              "WRITE  "appropriate information to munition work file"
               INCREMENT  "munition density factor counter by 1"
            ENDDO
            "identify this particular group/class/category
             combination as being in inventory (MGP array)"
            READ  "next stock number"
    ENDIF
  ENDDO
  CLOSE "NSNDB"

END        MUNINV
```

```
LEVEL 4

  STORE      ROUTINE

    WRITE  "message - program is now calculating
            information about storage facilities"
    REWIND  MSADB (file code 03)
    READ  "first record from MSADB"
    DO WHILE  "not end of MSADB file"
      REWIND  SBDB (file code 04)
      READ  "first record from SBDB"
      DO WHILE  "building type does not match or not end of file"
        IF  "building type does match"
          THEN  "indicate a match"
          ELSE
        ENDIF
        READ  "next record from SBDB"
      ENDDO
      IF  "no match"
        THEN  "print error message and
               turn error switch-2 on"
        ELSE  "calculate building volume"
              "write this and other needed information
               to building work file"
      ENDIF
      READ  "next record from MSADB"
    ENDDO
    CLOSE  "MSADB and SBDB"
  END        STORE


LEVEL 4

  FORM      ROUTINE

    PRINT  "message - program is generating objective
            function and constraints"
    WRITE  "file name on LPINFO (file code 08)"
    PEFORM  OBJCTV
    IF  "error switch-2 is not on"
      THEN
        PERFORM  MUNITN
        PERFORM  VOLUME
        PERFORM  SSET
        PERFORM  BLDNEW
        PERFORM  RHANDS
      ELSE
    ENDIF
    CLOSE  "munition and building work files, and LPINFO"
  END  FORM
```

LEVEL 4

```
OBJCTV      ROUTINE

  PRINT  "message - generating objective function"
  "set up the objective function id"
  REWIND  "munition work file (file code 02)"
  READ  "first munition from munition work file"
  DO WHILE  "not end of munition work file"
    REWIND  "building work file"
    READ  "first building from building work file"
    DO WHILE  "not end of building work file"
      SORT  "building NEW in descending order"
      READ  " most restrictive NEW"
      DO UNTIL  "current restrictive class(NEW) EQ class
                (munition NEW)"
        "select appropriate building density factor
         depending on current munition information"
        "calculate objective coefficient EQ
         building density factor / munition density factor"
        WRITE  "this information to LPINFO"
        WRITE  "this information to decision variable work file"
        READ  "next most restrictive NEW"
      ENDDO
      READ  "next building from building work file"
    ENDDO
    READ  "next munition from munition work file"
  ENDDO
  WRITE  "left over coefficient (0) to LPINFO"

END        OBJCTV
```

LEVEL 4

```
MUNITN      ROUTINE

  PRINT  "message - generating munition constraints"
  REWIND  "decision variable work file (file code 10)"
  READ  "first decision variable from decision variable work file"
  DO WHILE  "not end of decision variable work file"
    WRITE  "munition constraint"
    READ  "next decision variable from decision variable work file"
  ENDDO

END        MUNITN
```

LEVEL 4

```
VOLUME      ROUTINE

  PRINT  "message - generating building volume constraints"
  REWIND  "building work file (file code 07)"
  READ  "first building from building work file"
  DO WHILE  "not end of building work file"
    REWIND  "decision variable work file (file code 10)"
    "set up constraint id"
    READ  "first decision variable from decision variable work file"
    DO WHILE  "not end of decision variable work file"
      WRITE  "volume information to LPINFO"
      READ  "next decision variable from decision variable work file"
    ENDDO
    READ  "next building from building work file"
  ENDDO

END         VOLUME
```

LEVEL 4

```
SSET       ROUTINE

  REWIND  "building work file (file code 07)"
  READ  "first building from building work file"
  DO WHILE  "not end of building work file"
    WRITE  "special set identifier"
    READ  "first group from MGP"
    DO WHILE  "not end of groups"
      READ  "first class id from current group"
      DO WHILE  "not end of class ids in this group"
        IF  "this class is contained in inventory"
          THEN  WRITE  "decision set variable column id,
                subgroup constraint element for this set variable,
                and special set constraint element for this set variable"
          ELSE
        ENDIF
        READ  "next class id from current group"
      ENDDO
      READ  "next group from MGP"
    ENDDO
    READ  "next building from building work file"
  ENDDO

END         SSET
```

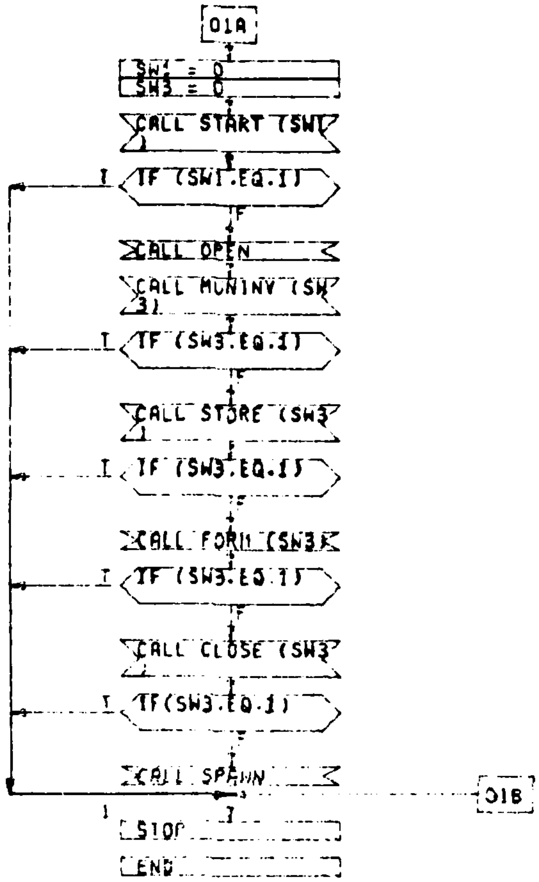
```
LEVEL 4

  BLDNEW      ROUTINE

    PRINT  "message - generating NEW constraints"
    REWIND  "building work file (file code 07)"
    READ  "first building from building work file"
    DO WHILE  "not end of building work file"
      PERFORM  NSORT  "to sort class NEW in descending order"
      READ  "first group to be processed"
      DO WHILE  "all groups have not been processed (MGP array)"
        READ  "first class from current group"
        DO WHILE  "all classes in this group are not processed"
          IF  "munitions of this class exist in inventory"
            THEN  "indicate that this class exists for group"
            ELSE
          ENDIF
          READ  "next class from current group"
        ENDDO
        IF  "current group does exist"
          THEN
            READ  "most restrictive NEW of current building"
            DO WHILE  "all restrictive NEW of current building
                       are not processed"
              READ  "first class in current group (MGP array)"
              DO WHILE  "all classes in current group have not
                         been processed"
                IF  "this class EQ current class associated with
                     current NEW of current building"
                   THEN  REWIND  "decision variable work file (fc 10)"
                     READ  "first decision variable from decision
                            variable work file"
                     DO WHILE  "not end of decision variable work file"
                       IF  "its group EQ current group and
                            its class EQ current class"
                         THEN  WRITE  "special set variable coefficient
                                       to LPINFO"
                               WRITE  "NEW information to LPINFO"
                         ELSE
                       ENDIF
                       READ  "next decision variable from decision"
                              variable work file"
                     ENDDO
                   ELSE
                ENDIF
                READ  "next class of current group"
              ENDDO
              READ  "next most restrictive NEW for
                     current building"
            ENDDO
          ELSE
        ENDIF
        READ  "next group to be processed (MGP array)"
      ENDDO
      READ  "next building from building work file"
    ENDDO

  END         BLDNEW
```

```
LEVEL 4

  NSORT       ROUTINE

    "sort the NEW of current building in descending order"

  END         NSORT

LEVEL 4

  RHANDS      ROUTINE

    PRINT  "message - generating right hand side values"
    REWIND  "munition work file (file code 02)"
    READ  "first munition from munition work file"
    DO WHILE  "not end of munition work file"
      WRITE  "number of packages to LPINFO"
      READ  "next munition from munition work file"
    ENDDO
    REWIND  "building work file (file code 07)"
    READ  "first building from building work file"
    DO WHILE  "not end of building work file"
      WRITE  "building volume to LPINFO"
      WRITE  "special set variable RHS = 1 to LPINFO"
      READ  "next building from building work file"
    ENDDO

  END         RHANDS
```

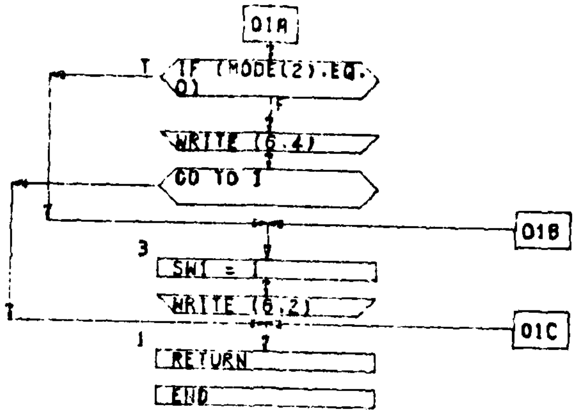
Appendix J

Format Generator Program Flow Chart

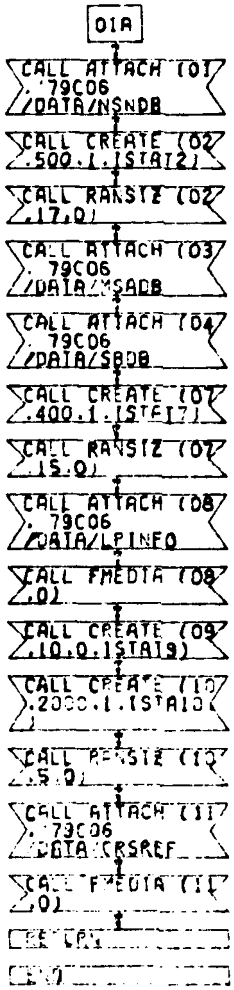
LPGEN


01A
SW1 = 0
SW3 = 0
CALL START (SW1)
IF (SW1.EQ.1)
CALL OPEN
CALL MUNINV (SW3)
IF (SW3.EQ.1)
CALL STORE (SW3)
IF (SW3.EQ.1)
CALL FORM (SW3)
IF (SW3.EQ.1)
CALL CLOSE (SW3)
IF(SW3.EQ.1)
CALL SPAWN
01B
STOP
END

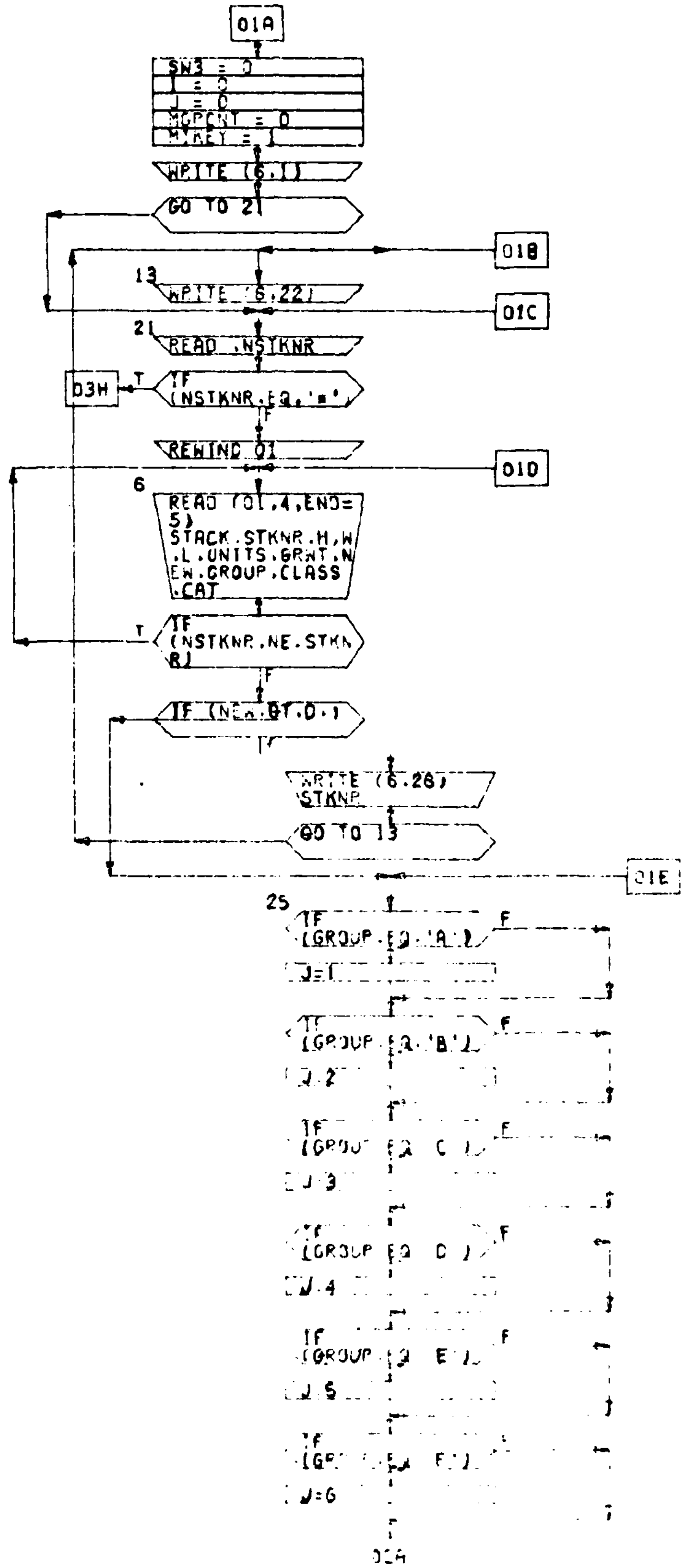
SUBROUTINE START



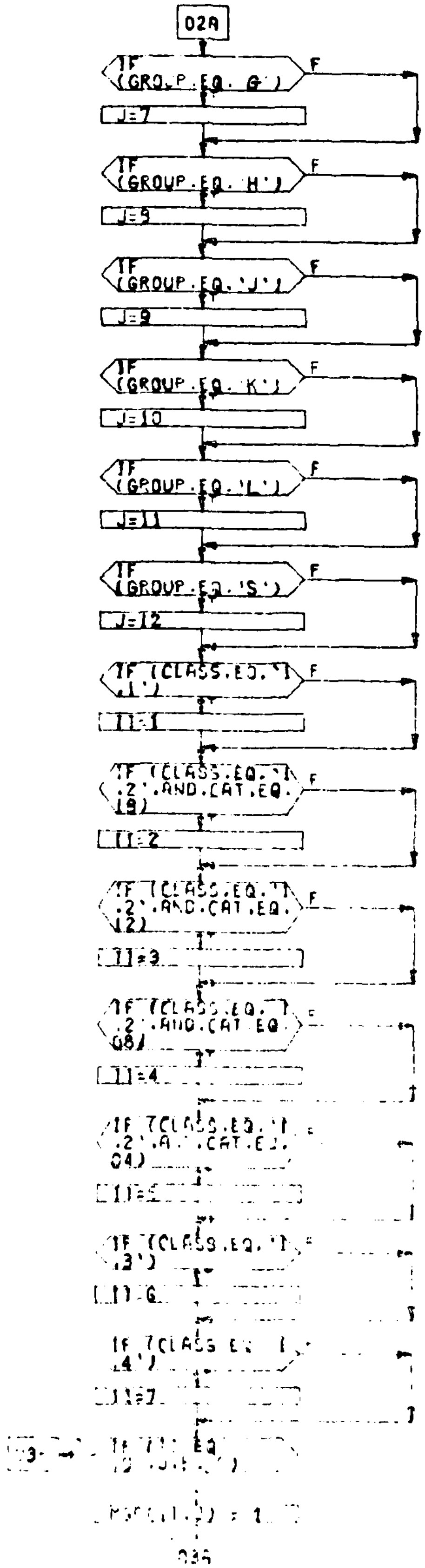
SUBROUTINE OPEN



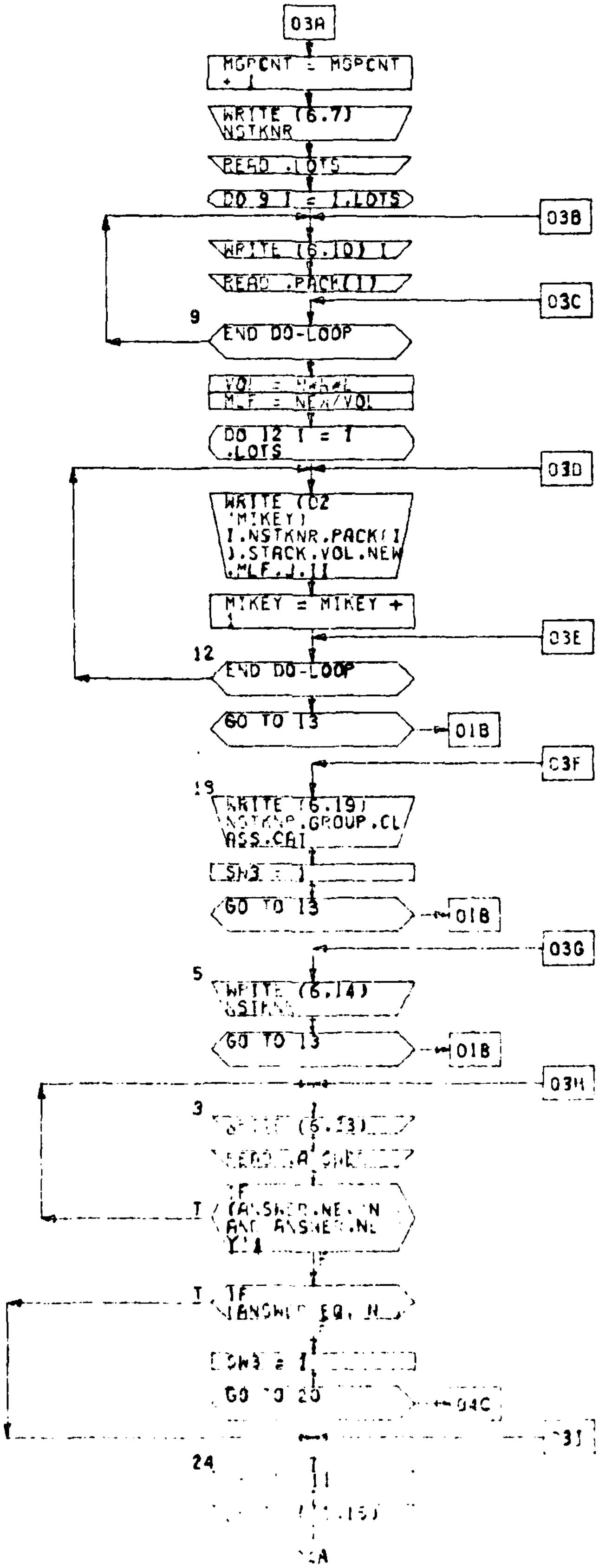
SUBROUTINE MUNINV



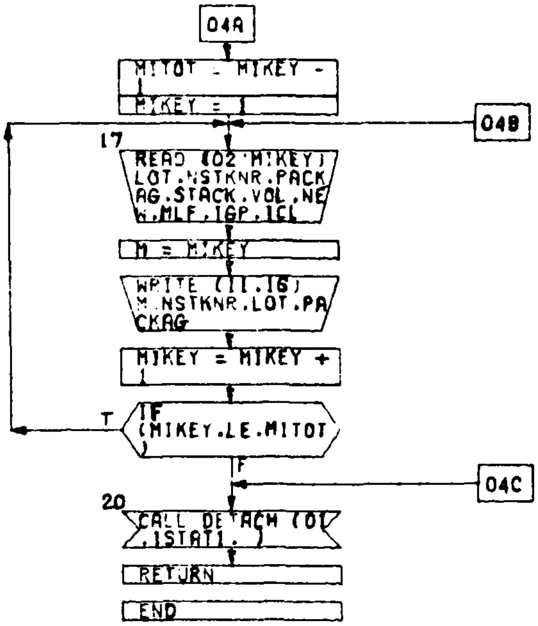
SUBROUTINE MUNINV



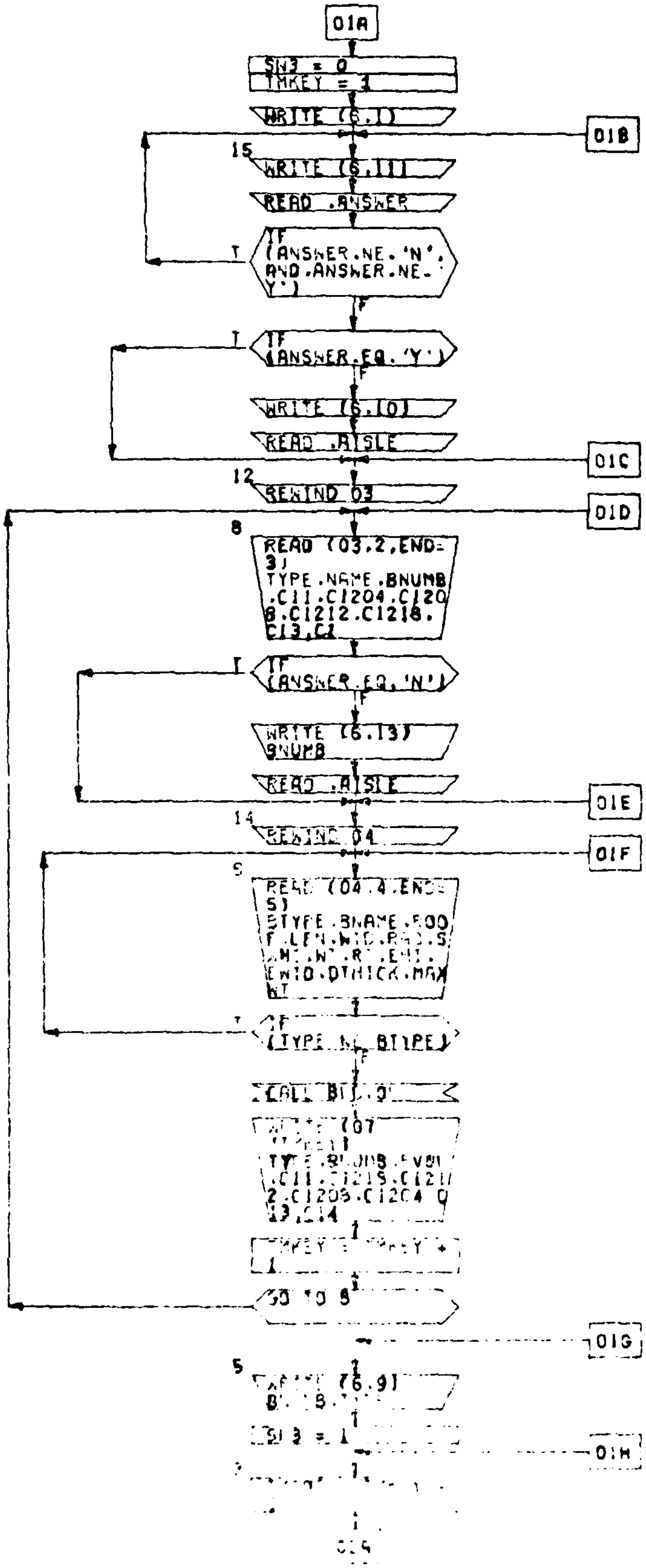
SUBROUTINE MUNINV



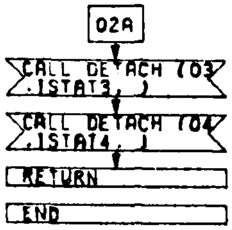
SUBROUTINE MUNINV



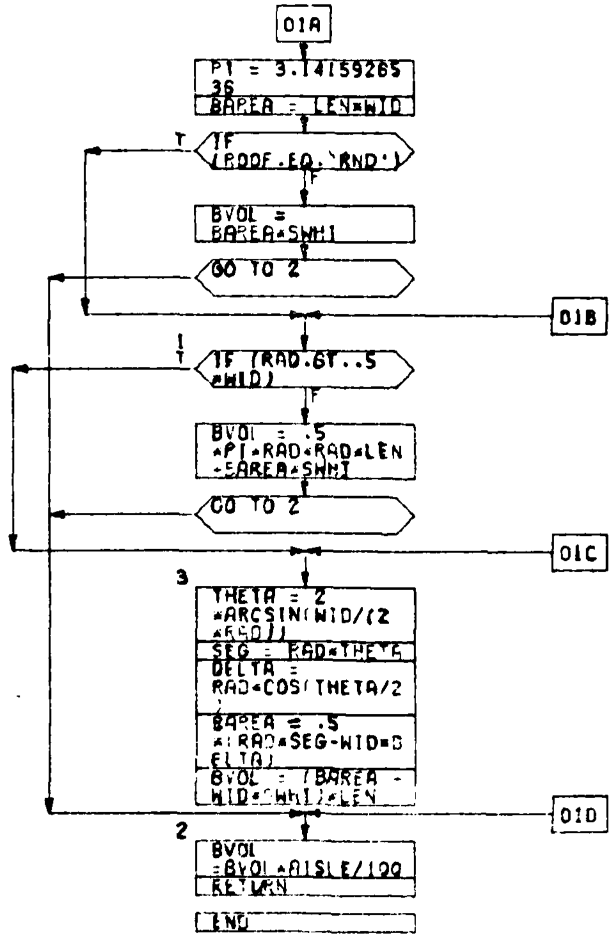
SUBROUTINE STORE



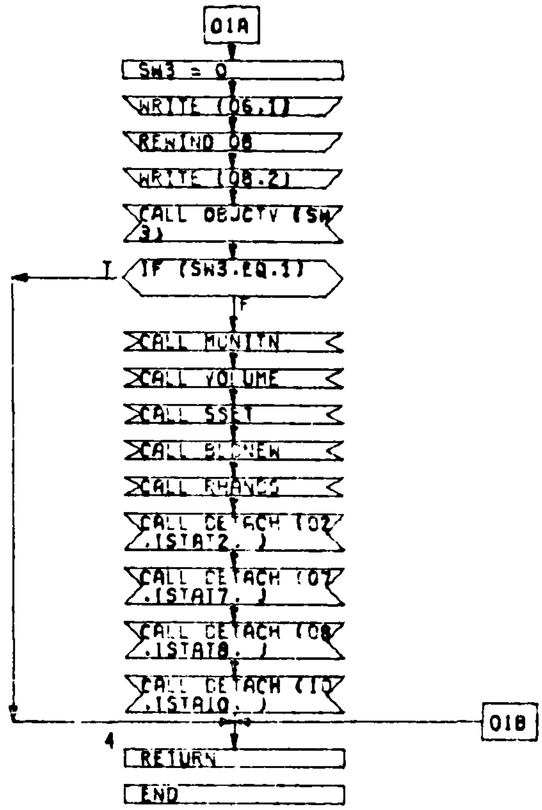
SUBROUTINE STORE



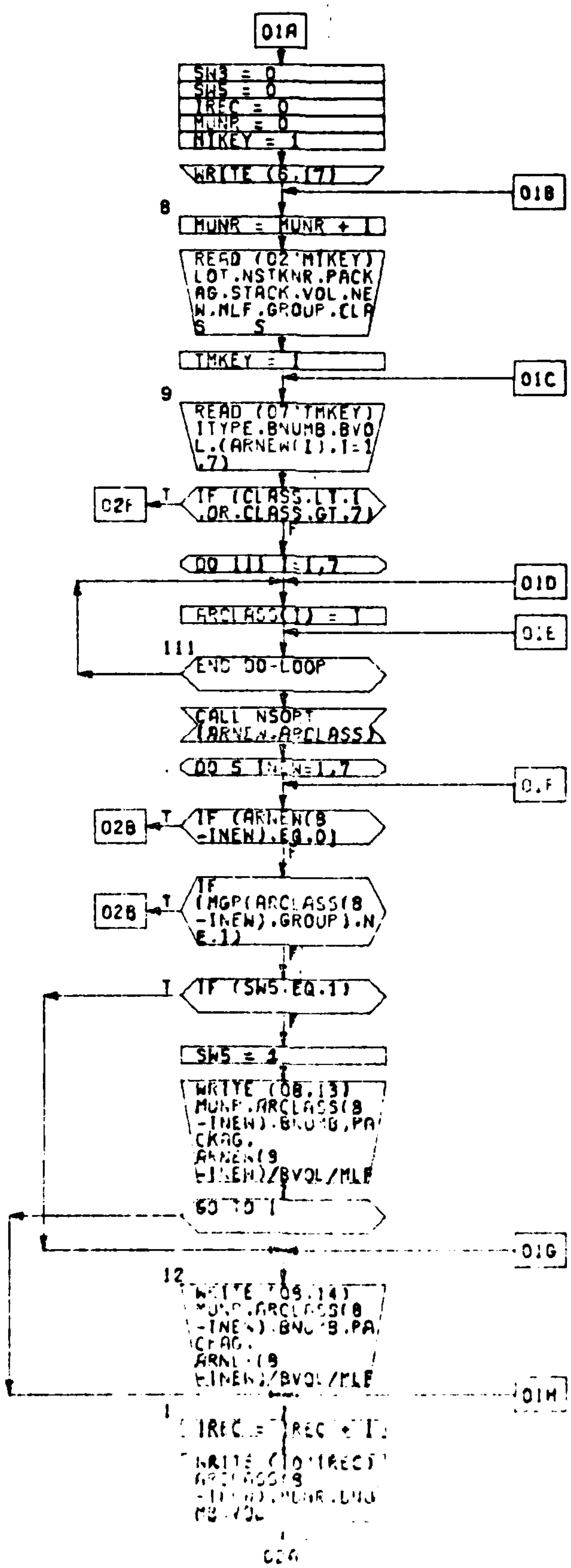
SUBROUTINE BLDVOL



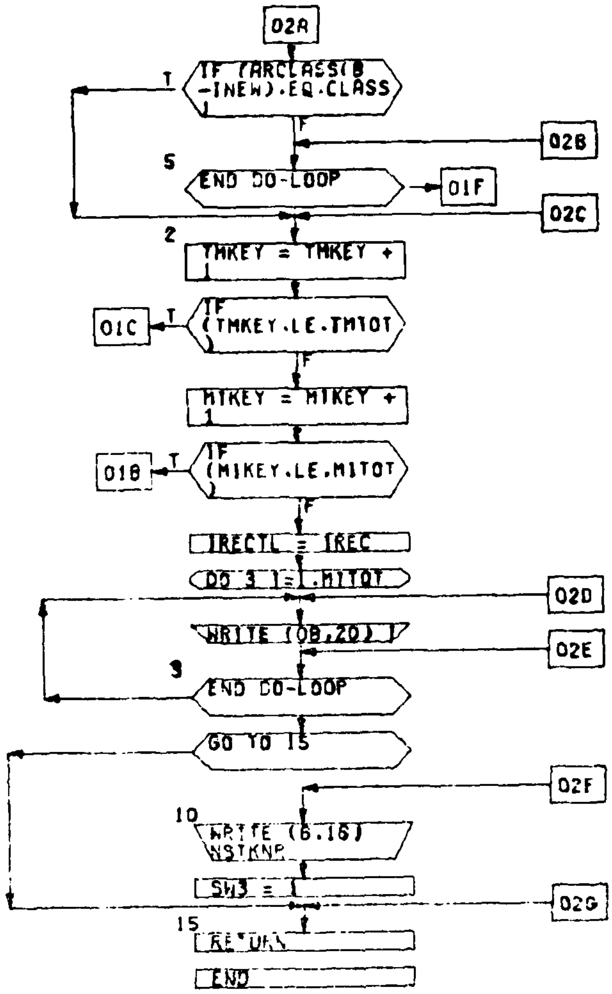
SUBROUTINE FORM



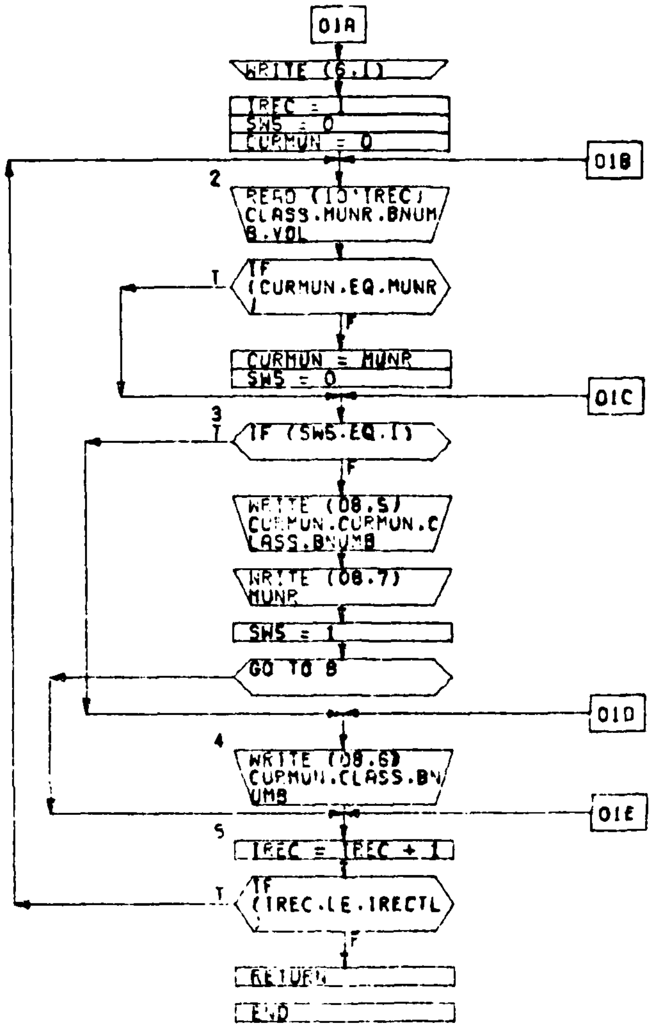
SUBROUTINE OBJCTV



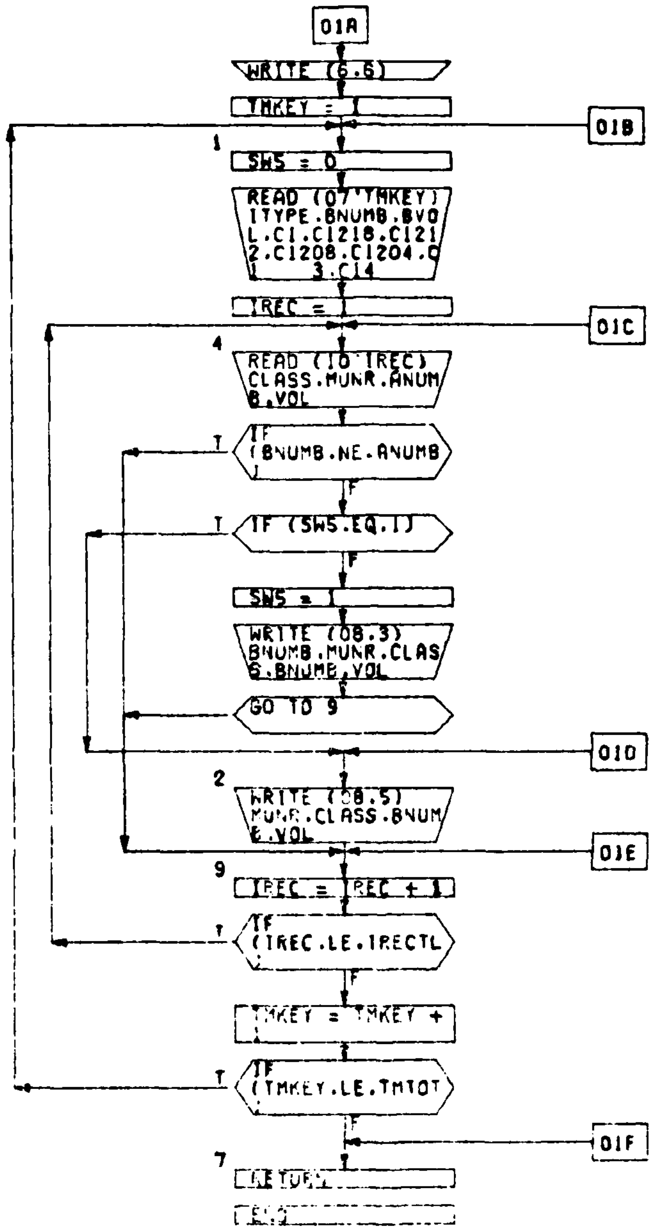
SUBROUTINE OBJCTV



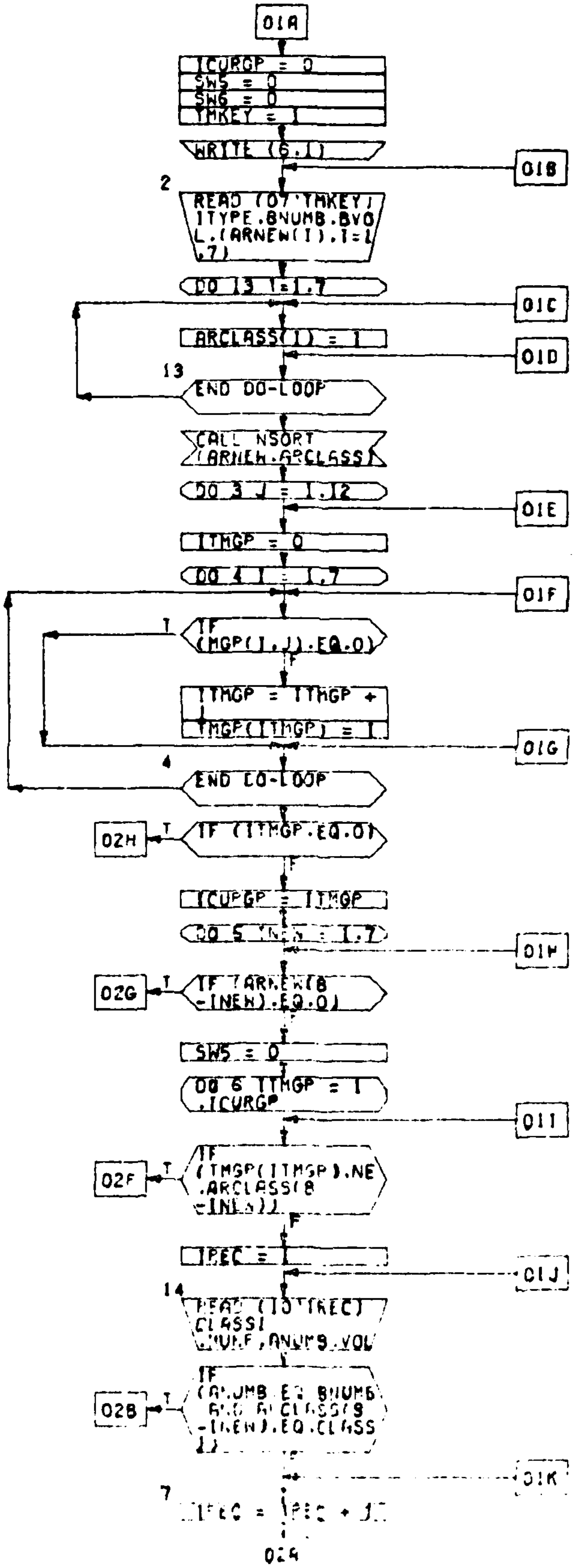
SUBROUTINE MUNITN



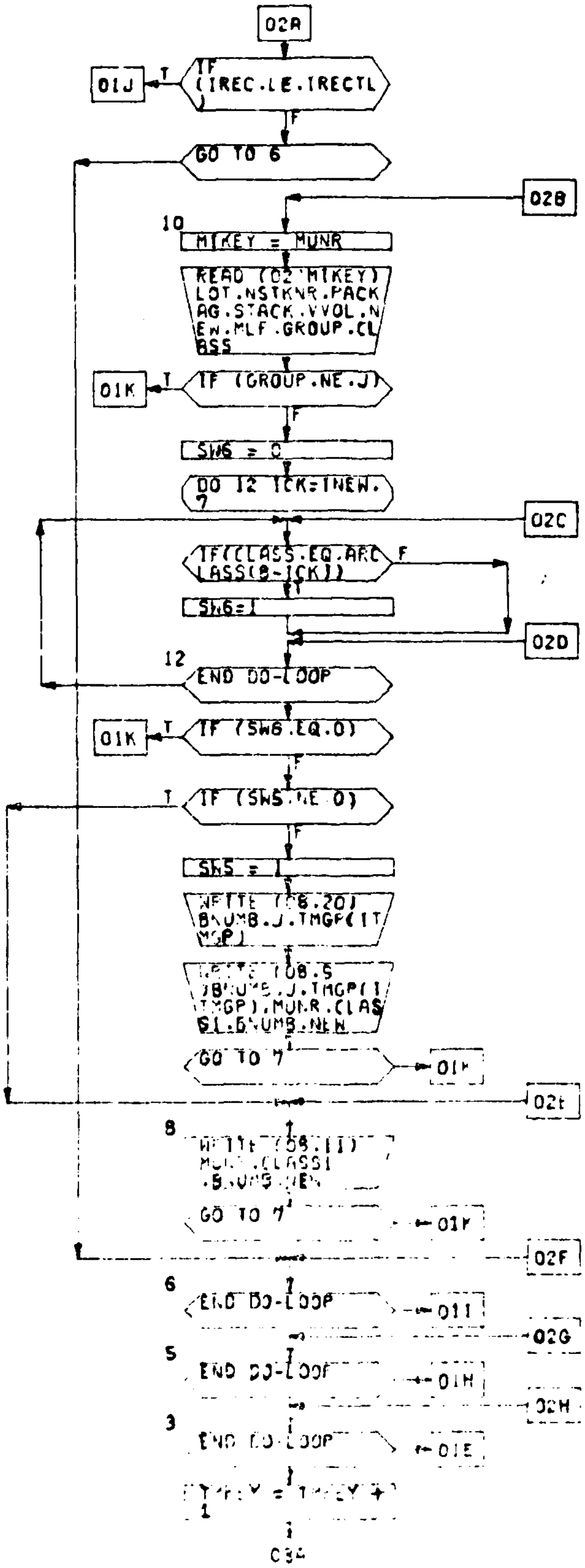
SUBROUTINE VOLUME



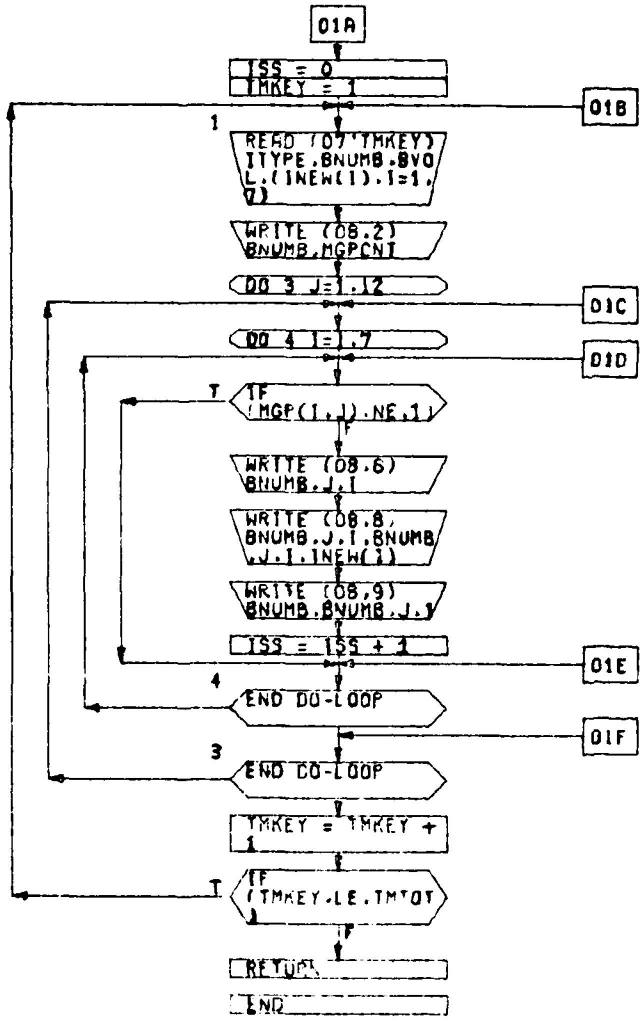
SUBROUTINE BLDNEW



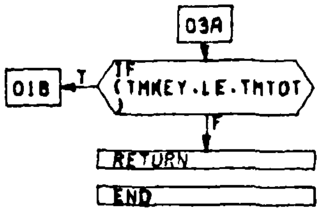
SUBROUTINE BLDNEW



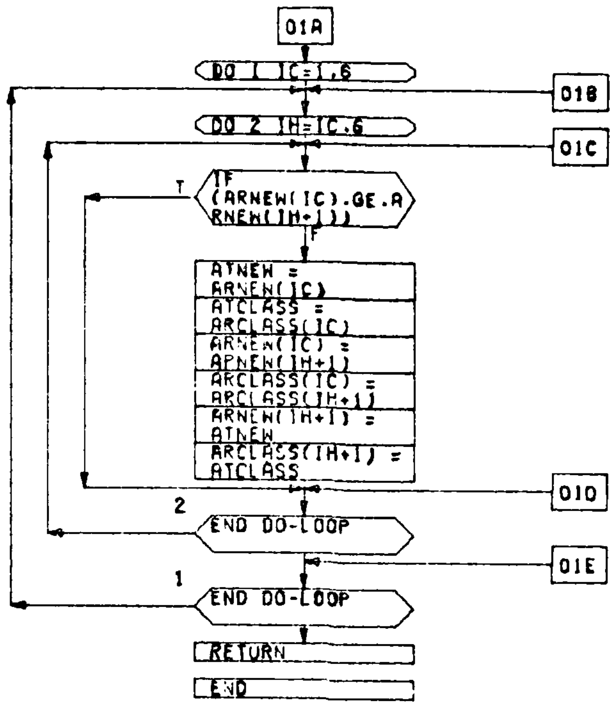
SUBROUTINE SSET



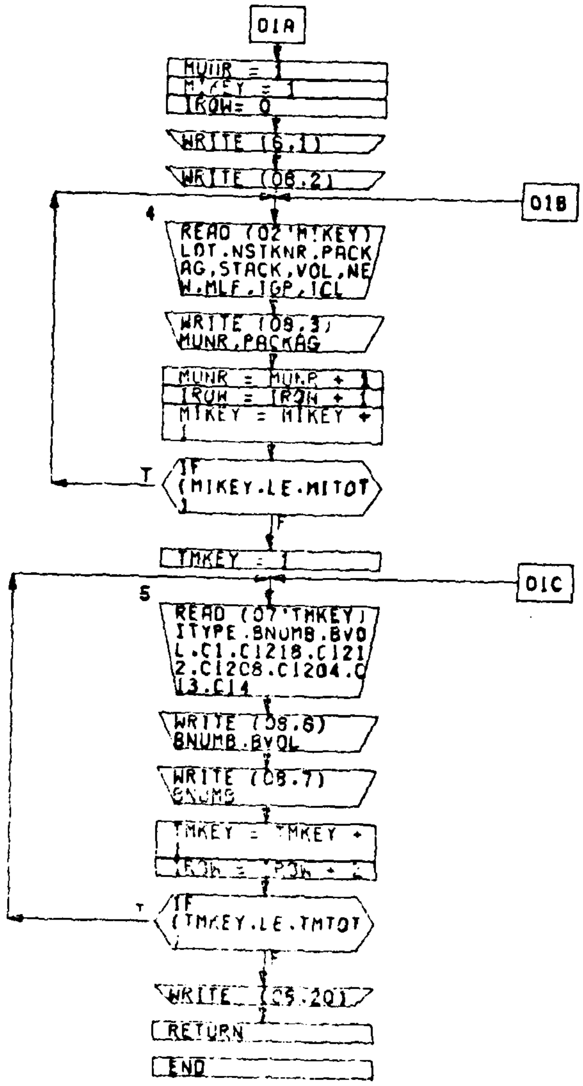
SUBROUTINE BLDNEW



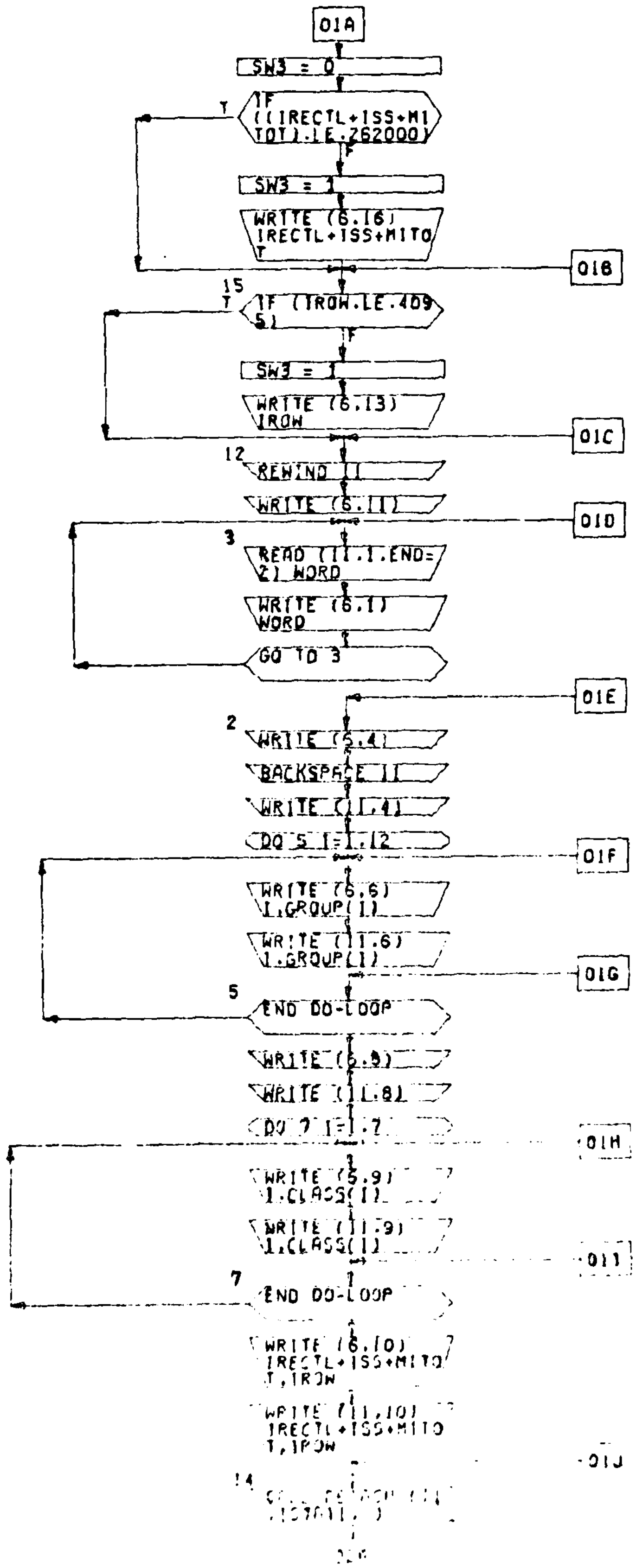
SUBROUTINE NSORT



SUBROUTINE RHANDS



SUBROUTINE CLOSE



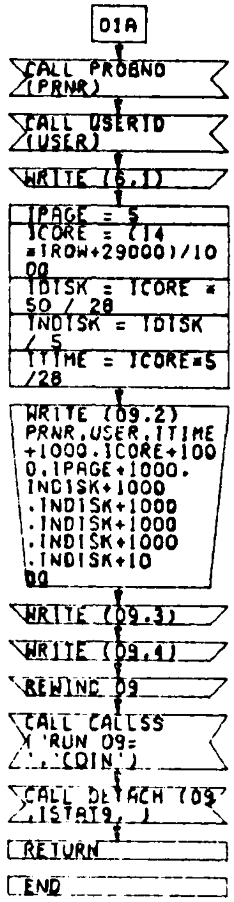
SUBROUTINE CLOSE

02A

RETURN

END

SUBROUTINE SPAWN



Appendix K

Format Generator Program Source Listing

```
*#RUN * = /OBJECT/LPGEN (NOGO)
C      MUNITION OPTIMIZATION FRONT END PROGRAM FOR LP/600    18 NOV 79
C
*****************************************************************************
*                                                                           *
*          LPGEN  MAIN                                                      *
*                                                                           *
*****************************************************************************
*                                                                           *
*************              PROGRAM IDENTIFICATION              *************
*                                                                           *
*          READS INFORMATION FROM THE STANDARD BUILDING DATA BASE,
*          THE MUNITION STORAGE AREA DATA BASE, AND THE NAITIONAL
*          STOCK NUMBER DATA BASE
*                                                                           *
*          READS IN PERCENTAGE OF USABLE VOLUME, ENTERED BY USER
*                                                                           *
*          CALCULATES THE OBJECTIVE COEFFICIENTS, THE MUNITIONS,
*          VOLUME, NET EXPLOSIVE WEIGHT (NEW) CONSTRAINTS,
*          AND RIGHT HAND SIDE VALUES
*                                                                           *
*          FORMATS THE DATA FOR INPUT INTO THE MIXED INTEGER
*          LINEAR PROGRAMMING LP/600 SOFTWARE PACKAGE
*                                                                           *
*          CREATES THE NECESSARY JOB CONTROL LANGUAGE
*                                                                           *
*                          PRODUCED BY
*                                                                           *
*                          MIKE GUSMUS
*                                                                           *
*****************************************************************************
*                                                                           *
*           THIS PROGRAM IS DESIGNED FOR OPERATION ON HONEYWELL 600
*           OR 6000 SERIES COMPUTER SYSTEMS THAT HAVE THE LP600
*           SYSTEM APPLICATIONS PACKAGE
*                                                                           *
*           REFERENCE HONEYWELL MANUALS BP50, BQ01, BQ19, BQ20,
*           BQ21,BQ22,DA87,DA88
*                                                                           *
*           THIS PROGRAM CONSISTS OF A MAIN DRIVER AND 15 SUBROUTINES
*                                                                           *
*****************************************************************************
*                                                                           *
```

```
*************             VARIABLE IDENTIFICATION            *************
*                                                                        *
*     SW1 - SWITCH THAT IDENTIFIES MODE OF OPERATION
*             1 - BCD     0 - ASCII
*                                                                        *
*     SW3 - ERROR IDENTIFICATION SWITCH
*            0 - NO ERROR     1 - ERROR
*                                                                        *
***************              SUBROUTINE NAMES               *****************
*                                                                        *
*     CALLED BY:  NONE
*                                                                        *
*     CALLS:
*             START - CHECKS MODE OF OPERATION
*                                                                        *
*             OPEN - OPENS NECESSARY FILES
*                                                                        *
*             MUNINV - READ IN THE NUMITION INVENTORY
*                                                                        *
*             STORE - CALCULATES BUILDING VOLUME
*                                                                        *
*             FORM - DRIVER THAT CONTROLS FORMULATION OF OBJECTIVE
*                    FUNCTION; MUNITION, VOLUME, AND NEW CONSTRAINTS;
*                    SPECIAL SET VARIABLES; AND RIGHT HAND SIDE VALUES
*                                                                        *
*             CLOSE - WRITES GROUP, CLASS, AND MUNTION CROSS
*                     REFERENCE LISTS
*                                                                        *
*             SPAWN - CREATES JCL FOR LP/600 AND SUBMITS THE JOB
*                     TO COMPUTER
*                                                                        *
**************************************************************************
*                                                                        *
      COMMON /PT2/ MITOT,MGP,MGPCNT
      COMMON /PT3/ AISLE,BVOL,ROOF,LEN,WID,RAD,SWHI
      COMMON /PT4/ TMTOT,IRECTL
      COMMON /PT5/ IROW,ISS
      CHARACTER ROOF*3
      INTEGER SW1*1,SW3*1,MITOT*4,TMTOT*4,MGP(7,12)
      REAL MLF,MLFARY(500)
      SW1 = 0
      SW3 = 0
      CALL START (SW1)
      IF (SW1.EQ.1) GO TO 1
      CALL OPEN
      CALL MUNINV (SW3)
      IF (SW3.EQ.1) GO TO 1
      CALL STORE (SW3)
      IF (SW3.EQ.1) GO TO 1
      CALL FORM (SW3)
      IF (SW3.EQ.1) GO TO 1
      CALL CLOSE (SW3)
      IF(SW3.EQ.1) GO TO 1
      CALL SPAWN
    1 STOP
      END
***************              END MAIN                   *****************
```

```
**************************************************************************
*                                                                        *
      SUBROUTINE START (SW1)
*                                                                        *
**************************************************************************
*                                                                        *
*************            PROGRAM IDENTIFICATION             *************
*                                                                        *
*      IF MODE OF OPERATION IS ASCII
*        THEN  PRINT WELCOME MESSAGE AND RETURN
*        ELSE  TURN ON ERROR SWITCH, PRINT RESTART MESSAGE AND RETURN
*                                                                        *
**************************************************************************
*                                                                        *
**************************************************************************
*                                                                        *
*************            VARIABLE IDENTIFICATION            *************
*                                                                        *
*     MODE(2) - SYSTEM  IDENTIFICATION OF MODE OF OPERATION
*     SW1 - SWITCH THAT IDENTIFIES THE MODE OF OPERATION TO PROGRAM
*                                                                        *
****************               SUBROUTINE NAMES          *****************
*                                                                        *
*      CALLED BY:  MAIN
*                                                                        *
*      CALLS:  NONE
*                                                                        *
**************************************************************************
*                                                                        *
        INTEGER SW1*1
        IF (MODE(2).EQ 0)GO TO 3
        WRITE (6,4)
        GO TO 1
    3     SW1 = 1
          WRITE (6,2)
*                                                                        *
    2   FORMAT (5X,"PLEASE RESTART USING 'RUN'")
    4   FORMAT (5X,"WELCOME TO THE INVENTORY PROGRAM"//)
    1 RETURN
      END
*                                                                        *
****************               END START                 *****************
```

```
************************************************************************
*                                                                      *
      SUBROUTINE OPEN
*                                                                      *
************************************************************************
*                                                                      *
*************             PROGRAM IDENTIFICATION            *************
*                                                                      *
*      THIS ROUTINE ATTACHES OR CREATES THE NECESSARY INPUT/OUTPUT FILES
*                                                                      *
*      ** THE CALL ROUTINES ARE SYSTEM DEPENDENT **
************************************************************************
*                                                                      *
************************************************************************
*                                                                      *
*************             VARIABLE IDENTIFICATION           *************
*                                                                      *
*     ISTAT1 - FILE 01 STATUS SWITCH
*     ISTAT2 - FILE 02 STATUS SWITCH
*     ISTAT3 - FILE 03 STATUS SWITCH
*     ISTAT4 - FILE 04 STATUS SWITCH
*     ISTAT7 - FILE 07 STATUS SWITCH
*     ISTAT8 - FILE 08 STATUS SWITCH
*     ISTAT9 - FILE 09 STATUS SWITCH
*     ISTA10 - FILE 10 STATUS SWITCH
*     ISTA11 - FILE 11 STATUS SWITCH
*                                                                      *
****************             SUBROUTINE NAMES          ******************
*                                                                      *
*      CALLED BY:  MAIN
*                                                                      *
*      CALLS:
*               ATTACH - ATTACHES PERMANENT FILE
*               CREATE - CREATES A TEMPORARY FILE
*               RANSIZ - DEFINES RECORD SIZE FOR RANDOM FILE
*               FMEDIA - DEFINES TYPE OF FILE
*                                                                      *
************************************************************************
*                                                                      *
$        CALL ATTACH (01,"79C06/DATA/NSNDB;",3,0,ISTAT1, )
         CALL CREATE (02,500,1,ISTAT2)
         CALL RANSIZ (02,17,0)
$        CALL ATTACH (03,"79C06/DATA/MSADB;",3,0,ISTAT3, )
$        CALL ATTACH (04,"79C06/DATA/SBDB;",3,0,ISTAT4, )
         CALL CREATE (07,400,1,ISTAT7)
         CALL RANSIZ (07,15,0)
$        CALL ATTACH (08,"79C06/DATA/LPINFO;",3,0,ISTAT8, )
         CALL FMEDIA (08,0)
         CALL CREATE (09,10,0,ISTAT9)
         CALL CREATE (10,2000,1,ISTA10, )
         CALL RANSIZ (10,5,0)
$        CALL ATTACH (11,"79C06/DATA/CRSREF;",3,0,ISTA11, )
         CALL FMEDIA (11,0)
      RETURN
      END
*                                                                      *
****************               END OPEN                ******************
```

```
*************************************************************************
*                                                                       *
      SUBROUTINE MUNINV (SW3)
*                                                                       *
*************************************************************************
*                                                                       *
*************             PROGRAM IDENTIFICATION            *************
*                                                                       *
*       READS MUNITION STOCK NUMBER FROM TERMINAL THEN SEARCHES
*       NSNDB TO VERIFY ITS EXISTANCE
*                                                                       *
*       USES DATA FROM NSNDB TO CALCULATE THE MUNITION DENSITY
*       FACTOR AND VOLUME FOR AN INDIVIDUAL PACKAGE OF CURRENT
*       MUNITION AND WRITES THIS DATA TO TEMPORARY WORK FILE
*                                                                       *
*       WRITES MUNITION CROSS REFERENCE LIST TO CRSREF FILE
*                                                                       *
*************************************************************************
*                                                                       *
*************************************************************************
*                                                                       *
*************             VARIABLE IDENTIFICATION           *************
*                                                                       *
*       CAT - MUNITION CATEGORY
*       CLASS - MUNITION CLASS
*       GROUP - MUNITION COMPATIBILITY GROUP
*       GRWT - PACKAGE GROSS WEIGHT
*       H - MUNITION PACKAGE HEIGHT
*       I - INDEX
*       II - INDEX
*       J - INDEX
*       L - MUNITION PACKAGE LENGTH
*       LOTS - NUMBER OF DIFFERENT LOTS PER MUNITION
*       MGP - MATRIX CONTAINING IDENTIFICATION OF DIFFERENT GROUP/
*             CLASS COMBINATIONS IN INVENTORY
*       MGPCNT - NUMBER OF GROUP/CLASS COMBINATIONS STORED IN MGP
*       MIKEY - INDEX KEY FOR INVENTORY WORK FILE
*       MITOT - NUMBER OF MUNITION/LOT COMBINATIONS IN INVENTORY
*       MLF - MUNITION DENSITY FACTOR (NEW/VOL)
*       NEW - MUNITION PACKAGE NET EXPLOSIVE WEIGHT
*       NSTKNR - MUNITION NATIONAL STOCK NUMBER
*       PACK - NUMBER OF PACKAGES IN CURRENT MUNITION/LOT COMBINATION
*       PACKAG - SAME AS ABOVE
*       STACK - NUMBER OF PACKAGES THAT CAN BE STACKED IN COLUMN
*       STKNR - MUNITION NATIONAL STOCK NUMBER
*       SW3 - ERROR SWITCH:  0 - NO ERROR, 1 - ERROR
*       UNITS - UNITS CONTAINED IN A PACKAGE
*       VOL - MUNITION PACKAGE VOLUME
*       W - MUNITION PACKAGE WIDTH
*                                                                       *
***************               SUBROUTINE NAMES           ***************
*                                                                       *
*       CALLED BY:  MAIN
*                                                                       *
*       CALLS:  DETACH - RELEASES SPECIFIED FILE FROM PROGRAM CONTROL
*                                                                       *
*************************************************************************
*                                                                       *
```

```
      INTEGER MIKEY*4,MITOT*4,MGP(7,12),PACK*4(50),STACK*4,
     &  CAT*2,UNITS*4,SW3
      REAL MLF,L,NEW
      CHARACTER CLASS*3,ANSWER*1
      CHARACTER GROUP*1,NSTKNR*18,STKNR*18
      COMMON /PT2/MITOT,MGP,MGPCNT
      SW3 = 0
      I = 0
      J = 0
      MGPCNT = 0
      MIKEY = 1
      WRITE (6,1)
      GO TO 21
13    WRITE (6,22)
21    READ ,NSTKNR
      IF (NSTKNR.EQ."*") GO TO 3
      REWIND 01
6     READ (01,4,END=5) STACK,STKNR,H,W,L,UNITS,GRWT,NEW,GROUP,CLASS,CAT
      IF (NSTKNR.NE.STKNR) GO TO 6
*                                                                      *
*   EXCLUDES MUNITION IF NEW = 0
*                                                                      *
      IF (NEW.GT.0.)GO TO 25
      WRITE (6,26) STKNR
*                                                                      *
*   TURNS ON GROUP/CLASS ELEMENT IN MGP MATRIX
*                                                                      *
      GO TO 13
25    IF (GROUP.EQ."A") J=1
      IF (GROUP.EQ."B") J=2
      IF (GROUP.EQ."C") J=3
      IF (GROUP.EQ."D") J=4
      IF (GROUP.EQ."E") J=5
      IF (GROUP.EQ."F") J=6
      IF (GROUP.EQ."G") J=7
      IF (GROUP.EQ."H") J=8
      IF (GROUP.EQ."J") J=9
      IF (GROUP.EQ."K") J=10
      IF (GROUP.EQ."L") J=11
      IF (GROUP.EQ."S") J=12
      IF (CLASS.EQ."1.1") II=1
      IF (CLASS.EQ."1.2".AND.CAT.EQ.18) II=2
      IF (CLASS.EQ."1.2".AND.CAT.EQ.12) II=3
      IF (CLASS.EQ."1.2".AND.CAT.EQ.08) II=4
      IF (CLASS.EQ."1.2".AND.CAT.EQ.04) II=5
      IF (CLASS.EQ."1.3") II=6
      IF (CLASS.EQ."1.4") II=7
      IF (II.EQ.0.OR.J.EQ.0) GO TO 18
      IF (MGP(II,J).NE.0) GO TO 30
      MGP(II,J) = 1
      MGPCNT = MGPCNT + 1
30    WRITE (6,7) NSTKNR
      READ ,LOTS
      DO 9 I = 1,LOTS
      WRITE (6,10) I
      READ ,PACK(I)
9     CONTINUE
      VOL = H*W*L
      MLF = NEW/VOL
```

```
*                                                                        *
*     WRITES MUNITION/LOT INFORMATION TO WORK FILE
*                                                                        *
          DO 12 I = 1,LOTS
          WRITE (02'MIKEY) I,NSTKNR,PACK(I),STACK,VOL,NEW,MLF,J,II
          MIKEY = MIKEY + 1
   12     CONTINUE
          GO TO 13
*                                                                        *
*     *****  FATAL ERROR MESSAGE
*                                                                        *
   18     WRITE (6,19) NSTKNR,GROUP,CLASS,CAT
          SW3 = 1
          GO TO 13
*     *  WARNING MESSAGE
*                                                                        *
    5     WRITE (6,14) NSTKNR
          GO TO 13
    3     WRITE (6,23)
          READ ,ANSWER
          IF (ANSWER.NE."N".AND.ANSWER.NE."Y") GO TO 3
          IF (ANSWER.EQ."N") GO TO 24
*                                                                        *
*     IF ANSWER = "Y" PROGRAM WILL STOP   ***
*                                                                        *
          SW3 = 1
          GO TO 20
   24     REWIND 11
          WRITE (11,15)
          MITOT = MIKEY - 1
          MIKEY = 1
*                                                                        *
*     WRITES MUNITION CROSS REFERENCE LISTS
*                                                                        *
   17     READ (02'MIKEY) LOT,NSTKNR,PACKAG,STACK,VOL,NEW,MLF,IGP,ICL
          M = MIKEY
          WRITE (11,16) M,NSTKNR,LOT,PACKAG
          MIKEY = MIKEY + 1
          IF (MIKEY.LE.MITOT) GO TO 17
*                                                                        *
    1     FORMAT (5X,"ENTER THE NATIONAL STOCK NUMBER OF MUNITION TO BE",
     &       /10X,"ENTERED IN INVENTORY AND HIT RETURN KEY"/,
     &       10X,"IF FINISHED ENTER '*' AND HIT RETURN")
*                                                                        *
    4     FORMAT (I4,A18,3F5.1,I4,2F10.4,A1,A3,I2)
*                                                                        *
    7     FORMAT (/5X,"ENTER THE NUMBER OF LOTS FOR MUNITION ",A18)
*                                                                        *
   10     FORMAT (5X,"ENTER THE NUMBER OF PACKAGES FOR LOT ",I2)
*                                                                        *
   14     FORMAT (2X,"** WARNING **  STOCK NUMBER ",A18,/5X,
     &       "DOES NOT RESIDE IN NSNDB -- MUNITION WILL NOT BE ACCEPTED"//)
*                                                                        *
   15     FORMAT (10X,"MUNITION INVENTORY CROSS REFERENCE LIST"/,
     &       5X,"ID NR",5X,"STOCK NUMBER",8X,"LOT",5X,"PACKAGES"//)
*                                                                        *
   16     FORMAT (6X,I4,2X,A18,5X,I2,8X,I4)
*                                                                        *
```

```
19   FORMAT (/5X,"MUNITION ",A18," HAS INVALID DATA IN GROUP,CLASS, ",
   &   "OR CAT",/10X,"VALUES ARE:  ",A1,",",A3,",",I2/,
   &   /20X,"CAUSING PROGRAM TO TERMINATE...."/)
*                                                                  *
22   FORMAT (/5X,"ENTER NEXT STOCK NR OR '*' IF FINISHED")
*                                                                  *
23   FORMAT (/5X,"DO YOU WISH TO STOP PROGRAM NOW? (Y OR N)")
*                                                                  *
26   FORMAT (/5X,"STOCK NUMBER ",A18," IS REJECTED BECAUSE NEW = 0"/)
*                                                                  *
20   CALL DETACH (01,ISTAT1, )
   RETURN
   END
*                                                                  *
****************                 END MUNINV           *****************
```

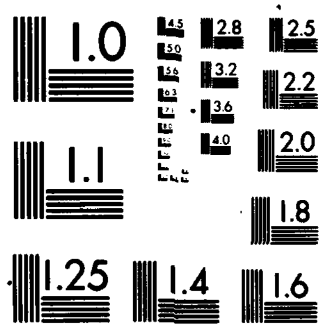
```
*************************************************************************
*                                                                       *
      SUBROUTINE STORE (SW3)
*                                                                       *
*************************************************************************
*                                                                       *
*************             PROGRAM IDENTIFICATION             *************
*                                                                       *
*     READS IN THE BUILDING INFORMATION FROM MSADB AND SBDB
*
*     READS THE USABLE BUILDING VOLUME PERCENTAGE FROM TERMINAL
*
*     CALCULATES THE BUILDING VOLUME
*                                                                       *
*     WRITES NECESSARY INFORMATION TO TEMPORARY WORK FILE
*                                                                       *
*************************************************************************
*                                                                       *
*************************************************************************
*                                                                       *
*************             VARIABLE IDENTIFICATION            *************
*                                                                       *
*     AISLE - PERCENTAGE OF USABLE BUILDING VOLUME
*     ANSWER - VALUE:  Y - YES,  N - NO
*     BNUMB - BUILDING NUMBER
*     BVOL - BUILDING VOLUME
*     BNAME - BUILDING NAME
*     BTYPE - STANDARD BUILDING TYPE
*     C11 - NEW FOR CLASS/DIV 1.1
*     C1204 - NEW FOR CLASS/DIV/CAT 1.2 04
*     C1208 - NEW FOR CLASS/DIV/CAT 1.2 08
*     C1212 - NEW FOR CLASS/DIV/CAT 1.2 12
*     C1218 - NEW FOR CLASS/DIV/CAT 1.2 18
*     C13 - NEW FOR CLASS/DIV 1.3
*     C14 - NEW FOR CLASS/DIV 1.4
*     DTHICK - DOOR THICKNESS
*     EHI - ENTRANCE HEIGHT
*     EWID - ENTRANCE WIDTH
*     LEN - BUILDING LENGTH
*     MAXWT - BUILDING GROSS STORAGE CAPACITY
*     NAME - BUILDING NAME
*     RAD - RADIUS OF IGLOO ROOF
*     ROOF - BUILDING ROOF STYLE:  RND - IGLOO,  FLT - OTHER
*     RT - ROOF THICKNESS
*     SW3 - ERROR SWITCH:  0 - NO ERROR, 1 - ERROR
*     SWHI - BUILDING SIDE WALL HEIGHT
*     TMKEY - INDEX KEY FOR BUILDING WORK FILE
*     TMTOT - NUMBER OF BUILDINGS IN MUNITION STORAGE AREA
*     TYPE - BUILDING TYPE
*     WID - BUILDING WIDTH
*     WT - WALL THICKNESS
*                                                                       *
****************             SUBROUTINE NAMES             ******************
*                                                                       *
*     CALLED BY:  MAIN
*                                                                       *
*     CALLS:
*            BLDVOL - TO CALCULATE USABLE BUILDING VOLUME
```

```
*            DETACH - RELEASES SPECIFIED FILE FROM PROGRAM CONTROL
*                                                                            *
******************************************************************************
*                                                                            *
          INTEGER SW3*1,TMKEY*4,TMTOT*4,TYPE*2,BTYPE*2
          INTEGER *7 C11,C1204,C1208,C1212,C1218,C13,C14
          CHARACTER NAME*6,BNAME*6,ROOF*3,ANSWER*1,BNUMB*6
          REAL MAXWT,LEN
          COMMON /PT3/ AISLE,BVOL,ROOF,LEN,WID,RAD,SWHI
          COMMON /PT4/ TMTOT,IRECTL
          SW3 = 0
          TMKEY = 1
          WRITE (6,1)
    15    WRITE (6,11)
          READ ,ANSWER
*                                                                            *
        IF ANSWER = "Y" READ IN AISLE FOR EVERY BUILDING
        IF ANSWER = "N" READ IN ONE AISLE THAT IS USED FOR ALL BUILDINGS
*                                                                            *
          IF (ANSWER.NE."N".AND.ANSWER.NE."Y") GO TO 15
          IF (ANSWER.EQ."Y") GO TO 12
          WRITE (6,10)
          READ ,AISLE
    12    REWIND 03
     8    READ (03,2,END=3) TYPE,NAME,BNUMB,C11,C1204,C1208,C1212,C1218,C13,C14
          IF (ANSWER.EQ."N") GO TO 14
          WRITE (6,13) BNUMB
          READ ,AISLE
    14    REWIND 04
     6    READ (04,4,END=5) BTYPE,BNAME,ROOF,LEN,WID,RAD,SWHI,WT,RT,EHI,
     &      EWID,DTHICK,MAXWT
          IF (TYPE.NE.BTYPE) GO TO 6
          CALL BLDVOL
*                                                                            *
        WRITES BUILDING INFORMATION TO WORK FILE
*                                                                            *
          WRITE (07'TMKEY) TYPE,BNUMB,BVOL,C11,C1218,C1212,C1208,C1204,C13,C14
          TMKEY = TMKEY + 1
          GO TO 8
*                                                                            *
        *****  FATAL ERROR MESSAGE
*                                                                            *
     5    WRITE (6,9) BNUMB,TYPE
          SW3 = 1
     3    TMTOT = TMKEY - 1
          CALL DETACH (03,ISTAT3, )
          CALL DETACH (04,ISTAT4, )
*                                                                            *
     1    FORMAT (//5X,"STORAGE FACILITY DATA IS NOW BEING GENERATED"//)
*                                                                            *
     2    FORMAT (I2,2A6,7I7)
*                                                                            *
     4    FORMAT (I2,A6,A3,3F6.2,F5.2,2F4.2,2F5.2,F4.2,F7.2)
*                                                                            *
     9    FORMAT (5X,"BUILDING - ",A6," IS IDENTIFIED AS TYPE ",I2/,
     &      10X,"BUT THIS TYPE OF BUILDING IS NOT DEFINED IN THE SBDB,"/,
     &      5X,"***** FATAL ERROR --- PROGRAM IS NOW TERMINATING ...."//)
*                                                                            *
```

AD-A083 708 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/G 15/5
OPTIMIZATION OF MUNITIONS STORAGE.(U)
DEC 79 B A BOGGS, L M GUSMUS
UNCLASSIFIED AFIT/GSM/SM/79D-15 NL

4 of 4

END
DATE
FILMED
6-80
DTIC



MICROCOPY RESOLUTION TEST CHART

```
10    FORMAT (5X,"ENTER THE PERCENTAGE OF BUILDING VOLUME THAT"/,
    &   10X,"IS CONSIDERED USABLE, E.G., 75.8")
*                                                                              *
11    FORMAT (/5X,"PLEASE READ THIS QUESTION CAREFULLY...",
    &   //8X,"DO YOU WISH TO ENTER A DIFFERENT PERCENTAGE OF USABLE",
    &   /5X,"VOLUME FOR EACH BUILDING? (Y OR N)")
*                                                                              *
13    FORMAT (/5X,"ENTER THE PERCENTAGE OF USABLE VOLUME FOR BUILDING ",
    &   A6," E.G.,75.8")
     RETURN
     END
*                                                                              *
****************                END STORE                     ******************
```

```
****************************************************************************
*                                                                          *
      SUBROUTINE BLDVOL
*                                                                          *
****************************************************************************
*                                                                          *
*************              PROGRAM IDENTIFICATION                *************
*                                                                          *
*       CALCULATES THE USABLE BUILDING VOLUME FOR EACH BUILDING
*       BASED ON BUILDING DESIGN
*                                                                          *
****************************************************************************
*                                                                          *
****************************************************************************
*                                                                          *
*************              VARIABLE IDENTIFICATION               *************
*                                                                          *
*       AISLE - PERCENTAGE OF USABLE BUILDING VOLUME
*       BAREA - BUILDING AREA
*       BVOL - BUILDING VOLUME
*       DELTA - PART OF SEGMENT AREA CALCULATION
*       LEN - BUILDING LENGTH
*       PI - CONSTANT
*       RAD - RADIUS OF IGLOO ROOF
*       ROOF - STYLE OF BUILDING:  RND - IGLOO  FLT - OTHER
*       SEG - ARC LENGTH
*       SWHI - BUILDING SIDE WALL HEIGHT
*       THETA - ANGLE OF SEGMENT
*       WID - BUILDING WIDTH
*
*       FUNCTIONS:
*
*                   ARCSIN - SYSTEM FUNCTION
*                   COS - SYSTEM FUNCTION
*                                                                          *
****************                SUBROUTINE NAMES           ******************
*                                                                          *
*       CALLED BY:  STORE
*                                                                          *
*       CALLS:
*               FUNCTIONS:
*                           ARCSIN AND COS
*                                                                          *
****************************************************************************
*                                                                          *
        REAL LEN
        CHARACTER ROOF*3
        COMMON /PT3/ AISLE,BVOL,ROOF,LEN,WID,RAD SWHI
        PI = 3.1415926536
        BAREA = LEN*WID
        IF (ROOF.EQ."RND") GO TO 1
        BVOL = BAREA*SWHI
        GO TO 2
    1   IF (RAD.GT..5*WID) GO TO 3
        BVOL = .5*PI*RAD*RAD*LEN+BAREA*SWHI
        GO TO 2
    3   THETA = 2*ARCSIN(WID/(2*RAD))
        SEG = RAD*THETA
```

```
        DELTA = RAD*COS(THETA/2)
        BAREA = .5*(RAD*SEG-WID*DELTA)
        BVOL = (BAREA + WID*SWHI)*LEN
    2   BVOL =BVOL*AISLE/100
      RETURN
      END
*                                                                        *
****************                 END BLDVOL                ******************
```

```
****************************************************************************
*                                                                          *
      SUBROUTINE FORM (SW3)
*                                                                          *
****************************************************************************
*                                                                          *
*************             PROGRAM IDENTIFICATION               *************
*                                                                          *
*     THE DRIVER ROUTINE FOR FORMATTING THE OBJECTIVE FUNCTION,
*     CONSTRAINTS, SPECIAL SET VARIABLES, AND RIGHT HAND SIDE VALUES
*     USED BY LP600
*                                                                          *
*     CLOSES OPEN FILES EXECPT FOR JCL FILE
*                                                                          *
****************************************************************************
*                                                                          *
****************************************************************************
*                                                                          *
*************             VARIABLE IDENTIFICATION              *************
*                                                                          *
*     ISTAT2 - FILE STATUS SWITCH FOR FILE 02
*     ISTAT7 - FILE STATUS SWITCH FOR FILE 07
*     ISTAT8 - FILE STATUS SWITCH FOR FILE 08
*     ISTA10 - FILE STATUS SWITCH FOR FILE 10
*     SW3 - ERROR CONDITION SWITCH:  0 - NO ERROR, 1 - ERROR
*                                                                          *
***************               SUBROUTINE NAMES               *****************
*                                                                          *
*     CALLED BY:  MAIN
*                                                                          *
*     CALLS:
*             OBJECTV - GENERATES OBJECTIVE FUNCTION COEFFICIENTS
*             MUNITN - GENERATES MUNITION CONSTRAINTS
*             VOLUME - GENERATES BUILDING VOLUME CONSTRAINTS
*             SSET - GENERATES SPECIAL SET VARIABLES USED FOR
*                    CONSTRAINT EXCLUSION
*             BLDNEW - GENERATES SUBGROUP CONSTRAINTS
*             RHANDS - GENERATES RIGHT HAND SIDE VALUES FOR CONSTRAINTS
*             DETACH - RELEASES SPECIFIED FILE FROM PROGRAM CONTROL
*                                                                          *
****************************************************************************
*                                                                          *
       COMMON /PT2/ MITOT,MGP,MGPCNT
       COMMON /PT4/ TMTOT,IRECTL
       COMMON /PT5/ IROW,ISS
       INTEGER TMTOT,SW3
       SW3 = 0
       WRITE (06,1)
       REWIND 08
*                                                                          *
*     THIS WRITES FILE NAME OF FILE LPINFO - USED BY LP600
*                                                                          *
       WRITE (08,2)
       CALL OBJCTV (SW3)
       IF (SW3.EQ.1) GO TO 4
       CALL MUNITN
       CALL VOLUME
       CALL SSET
```

```
      CALL BLDNEW
      CALL RHANDS
      CALL DETACH (02,ISTAT2, )
      CALL DETACH (07,ISTAT7, )
      CALL DETACH (08,ISTAT8, )
      CALL DETACH (10,ISTA10, )
*                                                                        *
    1   FORMAT (5X,"GENERATING OBJECTIVE FUNCTION AND CONSTRAINTS NOW,",
     &    /10X,"PLEASE WAIT...."/)
*                                                                        *
    2   FORMAT ("FILE   AMMO")
    4 RETURN
      END
*                                                                        *
****************                   END FORM                  ******************
```

```
************************************************************************
*                                                                      *
      SUBROUTINE OBJCTV (SW3)
*                                                                      *
************************************************************************
*                                                                      *
*************              PROGRAM IDENTIFICATION          *************
*                                                                      *
*      RESPONSIBLE FOR DEVELOPING THE OBJECTIVE COEFFIEIENTS,
*      FORMATTING AND WRITING THEM TO LPINFO
*                                                                      *
************************************************************************
*                                                                      *
************************************************************************
*                                                                      *
*************              VARIABLE IDENTIFICATION         *************
*                                                                      *
*      ARNEW(1) - BUILDING NEW FOR CLASS/DIVISION 1.1
*      ARNEW(2) - BUILDING NEW FOR CLASS/DIVISION/CAT 1.2 18
*      ARNEW(3) - BUILDING NEW FOR CLASS/DIVISION/CAT1.2 12
*      ARNEW(4) - BUILDING NEW FOR CLASS/DIVISION/CAT 1.2 08
*      ARNEW(5) - BUILDING NEW FOR CLASS/DIVISION/CAT 1.2 04
*      ARNEW(6) - BUILDING NEW FOR CLASS/DIVISION 1.3
*      ARNEW(7) - BUILDING NEW FOR CLASS/DIVISION 1.4
*      BNUMB - BUILDING NUMBER
*      BVOL - BUILDING VOLUME
*      CLASS - MUNITION CLASS ID
*      GROUP - MUNITION GROUP ID
*      I - INDEX
*      INEW - INDEX
*      IREC - INDEX KEY FOR DECISION VARIABLE WORK FILE
*      IRECTL - NUMBER OF DECISION VARIABLES
*      ITYPE - STANDARD BUILDING TYPE
*      LOT - MUNITION LOT ID
*      MGP - MATRIX CONTAINING IDENTIFICATION OF DIFFERENT GROUP/CLASS
*            COMBINATIONS IN INVENTORY
*      MIKEY - INDEX KEY FOR MUNITION INVENTORY WORK FILE
*      MITOT - NUMBER OF RECORDS IN INVENTORY WORK FILE
*      MLF - MUNITION DENSITY FACTOR
*      MUNR - INTERNAL MUNITION/LOT IDENTIFICATION NUMBER
*      NEW - NET EXPLOSIVES WEIGTH
*      NSTKNR - MUNITION NATIONAL STOCK NUMBER
*      PACKAG - LOT NUMBER OF CURRENT MUNITION
*      STACK - PACKAGE STACKING HEIGHT (IN PACKAGES)
*      SW3 - ERROR CONDITION SWITCH: 0 - NO ERROR, 1 - ERROR
*      SW5 - FIRST PASS SWITCH: 0 - FIRST PASS, 1 - SUBSEQUENT PASSES
*      TMKEY - INDEX KEY FOR BUILDING WORK FILE
*      TMTOT - NUMBER OF BUILDINGS IN MUNITION STORAGE AREA
*      VOL - MUNITION PACKAGE VOLUME
*                                                                      *
****************              SUBROUTINE NAMES          ****************
*                                                                      *
*      CALLED BY:  FORM
*                                                                      *
*      CALLS:  NSORT - SORTS THE CLASS ID AND BUILDING NEW
*                                                                      *
************************************************************************
*                                                                      *
```

```
      COMMON /PT2/ MITOT,MGP,MGPCNT
      COMMON /PT4/ TMTOT,IRECTL
      REAL MLF,NEW
      INTEGER TMTOT,SW3,SW5,TMKEY,GROUP*2,MUNR*4,CLASS*1
      INTEGER ARNEW(7),MGP(7,12),SW6,ARCLASS*1(7)
      CHARACTER NSTKNR*18,BNUMB*6
      SW3 = 0
      SW5 = 0
      IREC = 0
      MUNR = 0
      MIKEY = 1
      WRITE (6,17)
    8 MUNR = MUNR + 1
      READ (02'MIKEY) LOT,NSTKNR,PACKAG,STACK,VOL,NEW,MLF,GROUP,CLASS
      TMKEY = 1
    9 READ (07'TMKEY) ITYPE,BNUMB,BVOL,(ARNEW(I),I=1,7)
      IF (CLASS.LT.1.OR.CLASS.GT.7) GO TO 10
      DO 111 I=1,7
      ARCLASS(I) = I
  111 CONTINUE
      CALL NSORT (ARNEW,ARCLASS)
*                                                                    *
*     GENERATES SUBGROUP CONSTRAINTS STARTING WITH MOST RESTRICTIVE
*                                                                    *
      DO 5 INEW=1,7
      IF (ARNEW(8-INEW).EQ.0) GO TO 5
      IF (MGP(ARCLASS(8-INEW),GROUP).NE.1) GO TO 5
      IF (SW5.EQ.1) GO TO 12
      SW5 = 1
      WRITE (08,13) MUNR,ARCLASS(8-INEW),BNUMB,PACKAG,
     &  ARNEW(8-INEW)/BVOL/MLF
      GO TO 1
   12 WRITE (08,14) MUNR,ARCLASS(8-INEW),BNUMB,PACKAG,
     &  ARNEW(8-INEW)/BVOL/MLF
    1 IREC = IREC + 1
      WRITE (10'IREC) ARCLASS(8-INEW),MUNR,BNUMB,VOL
      IF (ARCLASS(8-INEW).EQ.CLASS) GO TO 2
    5 CONTINUE
    2 TMKEY = TMKEY + 1
      IF (TMKEY.LE.TMTOT) GO TO 9
      MIKEY = MIKEY + 1
      IF (MIKEY.LE.MITOT) GO TO 8
      IRECTL = IREC
      DO 3 I=1,MITOT
      WRITE (08,20) I
    3 CONTINUE
      GO TO 15
*                                                                    *
*     *****  FATAL ERROR MESSAGE
*                                                                    *
   10 WRITE (6,16) NSTKNR
      SW3 = 1
*                                                                    *
   13 FORMAT ("MATRIX OBJECT:IVE(F),MU",I4,":C",I1," BD :",A6,
     &  "(I=0,",I4,")=",F16.8)
*                                                                    *
   14 FORMAT (7X,",MU",I4,":C",I1," BD :",A6,"(I=0,",I4,")=",F16.8)
*                                                                    *
```

```
16   FORMAT (5X,"THE CLASS DATA ITEM FOR STOCK NR ",A18/,
   &   10X,"HAS THE WRONG VALUE -- PROGRAM NOW TERMINATING"/)
*                                                                            *
17   FORMAT (/5X,"THE OBJECTIVE FUNCTION IS NOW BEING ORGANIZED"/)
*                                                                            *
20   FORMAT (7X,",LEFT:OVER:MU",I4,"(P)=-9E6")
15 RETURN
   END
*                                                                            *
****************             END OBJCTV              ******************
```

```
**************************************************************************
*                                                                        *
      SUBROUTINE MUNITN
*                                                                        *
**************************************************************************
*                                                                        *
*************             PROGRAM IDENTIFICATION             *************
*                                                                        *
*      GENERATES THE MUNITION CONSTRAINTS AND WRITES THEM TO LPINFO
*                                                                        *
*      BUILDS ONE CONSTRAINT FOR EACH MUNITION/LOT CONTAINING
*      COEFFIEIENTS = 1 FOR EACH BUILDING/CLASS COMBINATION ASSOCIATED
*      WITH THE CURRENT MUNITION
*                                                                        *
**************************************************************************
*                                                                        *
**************************************************************************
*                                                                        *
*************            VARIABLE IDENTIFICATION             *************
*                                                                        *
*      BNUMB - BUILDING NUMBER
*      CLASS - MUNITION CLASS ID
*      CURMUN - CURRENT INTERNAL MUNITION/LOT ID
*      IREC - INDEX KEY FOR DECISION VARIABLE WORK FILE
*      IRECTL - NUMBER OF DECISION VARIABLES
*      MUNR - INTERNAL MUNITION/LOT ID
*      VOL - BUILDING VOLUME
*      SW5 - CONTROL SWITCH: 0 - FIRST PASS, 1 - SUBSEQUENT PASSES
*                                                                        *
****************             SUBROUTINE NAMES            *****************
*                                                                        *
*      CALLED BY:  FORM
*                                                                        *
*      CALLS:  NONE
*                                                                        *
**************************************************************************
*                                                                        *
       COMMON /PT4/ TMTOT,IRECTL
       INTEGER CLASS*1,CURMUN*4,MUNR*4,SW5
       CHARACTER BNUMB*6
       WRITE (6,1)
       IREC = 1
       SW5 = 0
       CURMUN = 0
  2    READ (10'IREC) CLASS,MUNR,BNUMB,VOL
       IF (CURMUN.EQ.MUNR) GO TO 3
       CURMUN = MUNR
       SW5 = 0
  3    IF (SW5.EQ.1) GO TO 4
       WRITE (08,5) CURMUN,CURMUN,CLASS,BNUMB
       WRITE (08,7) MUNR
       SW5 = 1
       GO TO 8
  4    WRITE (08,6) CURMUN,CLASS,BNUMB
  8    IREC = IREC + 1
       IF (IREC.LE.IRECTL) GO TO 2
*                                                                        *
  1    FORMAT (/5X,"NOW GENERATING MUNITION CONSTRAINTS...."/)
```

```
*                                                                    *
      5   FORMAT ("MATRIX MU",I4,"(Z),MU",I4,":C",I1," BD :",A6,"=1")
*                                                                    *
      6   FORMAT (7X,",MU",I4,":C",I1," BD :",A6,"=1")
*                                                                    *
      7   FORMAT (7X,",LEFT:OVER:MU",I4,"=1")
        RETURN
        END
*                                                                    *
****************                END MUNITN            ******************
```

```
*****************************************************************************
*                                                                           *
      SUBROUTINE VOLUME
*                                                                           *
*****************************************************************************
*                                                                           *
*************             PROGRAM IDENTIFICATION              *************
*                                                                           *
*       GENERATES BUILDING VOLUME CONSTRAINTS AND WRITES THEM TO LPINFO
*                                                                           *
*       BUILDS ONE CONSTRAINT FOR EACH BUILDING CONTAINING COEFFIEIENTS
*       = MUNITION DENSITY FACTOR (MLF) FOR MUNITION/CLASS COMBINATIONS
*       ASSOCIATED WITH THE CURRENT BUILDING
*                                                                           *
*****************************************************************************
*                                                                           *
*****************************************************************************
*                                                                           *
*************             VARIABLE IDENTIFICATION             *************
*                                                                           *
*       ANUMB - BUILDING NUMBER
*       BNUMB - BUILDING NUMBER
*       BVOL - BUILDING VOLUME
*       CLASS - MUNITION CLASS ID
*       C1 - BUILDING NEW FOR CLASS/DIVISION 1.1
*       C1218 - BUILDING NEW FOR CLASS/DIVISION/CAT 1.2 18
*       C1212 - BUILDING NEW FOR CLASS/DIVISION/CAT 1.2 12
*       C1208 - BUILDING NEW FOR CLASS/DIVISION/CAT 1.2 08
*       C1204 - BUILDING NEW FOR CLASS/DIVISION/CAT 1.2 04
*       C13 - BUILDING NEW FOR CLASS/DIVISION 1.3
*       C14 - BUILDING NEW FOR CLASS/DIVISION 1.4
*       IREC - INDEX KEY FOR DECISION VARIABLE WORK FILE
*       IRECTL - NUMBER OF DECISION VARIABLES
*       ITYPE - STANDARD BUILDING TYPE
*       MUNR - INTERNAL MUNITION/LOT ID
*       SW5 - CONTROL SWITCH:  0 - FIRST PASS, 1 - SUBSEQUENT PASSES
*       TMKEY - INDEX KEY FOR THE BUILDING WORK FILE
*       TMTOT - NUMBER OF BUILDINGS IN MUNITION STORAGE AREA
*       VOL - MUNITION PACKAGE VOLUME
*                                                                           *
****************               SUBROUTINE NAMES             ******************
*                                                                           *
*                                                                           *
*       CALLED BY:  FORM
*                                                                           *
*       CALLS:  NONE
*                                                                           *
*****************************************************************************
*                                                                           *
        COMMON /PT4/ TMTOT,IRECTL
        INTEGER TMTOT,TMKEY,SW5,IRECTL
        INTEGER *7 C1,C1218,C1212,C1208,C1204,C13,C14
        CHARACTER ANUMB*6,BNUMB*6
        WRITE (6,6)
        TMKEY = 1
    1   SW5 = 0
        READ (07'TMKEY) ITYPE,BNUMB,BVOL,C1,C1218,C1212,C1208,C1204,C13,C14
        IREC = 1
```

```
      4   READ (10'IREC) CLASS,MUNR,ANUMB,VOL
          IF (BNUMB.NE.ANUMB) GO TO 9
          IF (SW5.EQ.1) GO TO 2
          SW5 = 1
          WRITE (08,3) BNUMB,MUNR,CLASS,BNUMB,VOL
          GO TO 9
      2   WRITE (08,5) MUNR,CLASS,BNUMB,VOL
      9   IREC = IREC + 1
          IF (IREC.LE.IRECTL) GO TO 4
          TMKEY = TMKEY + 1
          IF (TMKEY.LE.TMTOT) GO TO 1
*                                                                            *
      3   FORMAT ("MATRIX BLDG  :",A6,":   VOL(P),MU",I4,":C",I1," BD :",A6,"=",
     &      F16.8)
*                                                                            *
      5   FORMAT (7X,",MU",I4,":C",I1," BD :",A6,"=",F16.8)
*                                                                            *
      6   FORMAT (/5X,"STARTING THE BUILDING VOLUME CONSTRAINTS...."/)
      7 RETURN
        END
*                                                                            *
****************               END VOLUME                *****************
```

```
**************************************************************************
*                                                                        *
      SUBROUTINE SSET
*                                                                        *
**************************************************************************
*                                                                        *
*************             PROGRAM IDENTIFICATION             *************
*                                                                        *
*        GENERATES SPECIAL SET INFORMATION, AS DEFINED IN
*        HONEYWELL MANUAL DA88
*                                                                        *
*        EACH SPECIAL SET CONSISTS OF A SET OF BIVALENT (0 OR 1)
*        DECISION VARIABLES FOR EACH BUILDING, ONE DECISION VARIABLE
*        IS ASSIGNED TO EACH SUBGROUP CONSTRAINT FOR THAT BUILDING (MGPCNT)
*                                                                        *
*        ONE CONSTRAINT IS GENERATED FOR EACH BUILDING THAT CONTAINS
*        ALL ASSOCIATED SPECIAL SET DECISION VARIABLES, THE SUM OF WHICH
*        IS EQUAL TO 1 - - IMPLYING THAT ONLY ONE OF THESE SPECIAL SET
*        VARIABLES WILL BE USED PER SET IN THE FINAL SOLUTION -- THIS
*        ALLOWS FOR SELECTING ONLY ONE OF A GROUP OF CONSTRAINTS
*                                                                        *
**************************************************************************
*                                                                        *
**************************************************************************
*                                                                        *
*************             VARIABLE IDENTIFICATION            *************
*                                                                        *
*        BNUMB - BUILDING NUMBER
*        BVOL - BUILDING VOLUME
*        I - INDEX
*        INEW(1) - NEW FOR CLASS/DIVISION 1.1
*        INEW(2) - NEW FOR CLASS/DIVISION/CAT 1.2 18
*        INEW(3) - NEW FOR CLASS/DIVISION/CAT 1.2 12
*        INEW(4) - NEW FOR CLASS/DIVISION/CAT 1.2 08
*        INEW(5) - NEW FOR CLASS/DIVISION/CAT 1.2 04
*        INEW(6) - NEW FOR CLASS/DIVISION 1.3
*        INEW(7) - NEW FOR CLASS/DIVISION 1.4
*        ISS - NUMBER OF SPECIAL SET VARIABLES
*        ITYPE - STANDARD BUILDING TYPE
*        J - INDEX
*        MGP - MARTIX CONTAINING IDENTIFICATION OF ALL GROUP/CLASS
*              COMBINATIONS IN INVENTORY
*        MGPCNT - NUMBER OF GROUP/CLASS COMBINATIONS IN INVENTORY
*        TMKEY - INDEX KEY FOR BUILDING WORK FILE
*        TMTOT - NUMBER OF BUILDINGS IN MUNITION STORAGE AREA
*                                                                        *
****************              SUBROUTINE NAMES          *****************
*                                                                        *
*        CALLED BY:  FORM
*                                                                        *
*        CALLS:  NONE
*                                                                        *
**************************************************************************
         COMMON /PT2/ MITOT,MGP(7,12),MGPCNT
         COMMON /PT4/ TMTOT,IRECTL
         COMMON /PT5/ IROW,ISS
         CHARACTER BNUMB*6
         INTEGER TMTOT,TMKEY,INEW(7)
```

```
      ISS = 0
      TMKEY = 1
    1 READ (07'TMKEY) ITYPE,BNUMB,BVOL,(INEW(I),I=1,7)
      WRITE (08,2) BNUMB,MGPCNT
      DO 3 J=1,12
      DO 4 I=1,7
      IF (MGP(I,J).NE.1) GO TO 4
      WRITE (08,6) BNUMB,J,I
      WRITE (08,8) BNUMB,J,I,BNUMB,J I,INEW(I)
      WRITE (08,9) BNUMB,BNUMB,J,I
      ISS = ISS + 1
    4 CONTINUE
    3 CONTINUE
      TMKEY = TMKEY + 1
      IF (TMKEY.LE.TMTOT) GO TO 1
*                                                                        *
    2 FORMAT ("STR    SS:",A6,"(S=",I4,")")
*                                                                        *
    6 FORMAT ("STR    SSBLDG:",A6,":G",I2," C",I1,"(P)")
*                                                                        *
    8 FORMAT ("MATRIX BLDG  :",A6,":G",I2," C",I1,"(P),SSBLDG:",A6,
     &   ":G",I2," C",I1,"=-",I7)
    9 FORMAT ("MATRIX SSET:",A6,"(Z),SSBLDG:",A6,":G",I2," C",I1,"=1")
*                                                                        *
    RETURN
    END
*                                                                        *
****************                  END SSET                ****************
```

```
*************************************************************************
*                                                                       *
      SUBROUTINE BLDNEW
*                                                                       *
*************************************************************************
*                                                                       *
*************             PROGRAM IDENTIFICATION            *************
*                                                                       *
*     GENERATES THE BUILDING SUBGROUP CONSTRAINTS AND WRITES
*     THEM TO LPINFO
*                                                                       *
*     CAN GENERATE UP TO 84 CONSTRAINTS FOR EACH BUILDING - ONE FOR
*     EVERY GROUP/CLASS COMBINATION IN INVENTORY
*                                                                       *
*************************************************************************
*                                                                       *
*************************************************************************
*                                                                       *
*************             VARIABLE IDENTIFICATION           *************
*                                                                       *
*     ANUMB - BUILDING NUMBER
*     ARCLASS - ARRAY CONTAINING THE INTERNAL MUNITION CLASS ID
*     ARNEW(1) - NEW FOR CLASS/DIVISION 1.1
*     ARNEW(2) - NEW FOR CLASS/DIVISION/CAT 1.2 18
*     ARNEW(3) - NEW FOR CLASS/DIVISION/CAT 1.2 12
*     ARNEW(4) - NEW FOR CLASS/DIVISION/CAT 1.2 08
*     ARNEW(5) - NEW FOR CLASS/DIVISION/CAT 1.2 04
*     ARNEW(6) - NEW FOR CLASS/DIVISION 1.3
*     ARNEW(7) - NEW FOR CLASS/DIVISION 1.4
*     BNUMB - BUILDING NUMBER
*     BVOL - BUILDING VOLUME
*     CLASS - MUNITION CLASS ID
*     CLASS1 - MUNITION CLASS ID
*     GROUP - MUNITION GROUP ID
*     I - INDEX
*     ICK - INDEX
*     ICURGP - NUMBER OF CLASSES IN CURRENT GROUP
*     INEW - INDEX
*     IREC - INDEX KEY FOR THE DECISION VARIABLE WORK FILE
*     IRECTL - NUMBER OF DECISION VARIABLES
*     ITMGP - INDEX
*     ITYPE - STANDARD BUILDING TYPE
*     J - INDEX
*     LOT - LOT IDENTIFICATION NUMBER
*     MGP - MARTIX CONTAINING IDENTIFICATION OF DIFFERENT GROUP/CLASS
*           COMBINATIONS IN INVENTORY
*     MIKEY - INDEX KEY FOR THE MUNITION INVENTORY WORK FILE
*     MLF - MUNITION DENSITY FACTOR
*     MUNR - INTERNAL MUNITION/LOT ID
*     NEW - MUNITION PACKAGE NET EXPLOSIVE WEIGHT
*     NSTKNR - MUNITION NATIONAL STOCK NUMBER
*     PACKAG - NUMBER OF PACKAGES FOR CURRENT MUNITION/LOT
*     STACK - MUNITION PACKAGE STACKING HEIGHT (IN PACKAGES)
*     SW5 - CONTROL SWITCH:  0 - FIRST PASS, 1 - SUBSEQUENT PASSES
*     SW6 - CONTROL SWITCH: 0 - CURRENT GROUP NOT IN INVENTORY
*                           1 - CURRENT GROUP IN INVENTORY
*     TMGP - ARRAY CONTAINING CLASSES IN CURRENT GROUP
*     TMKEY - INDEX KEY FOR BUILDING WORK FILE
```

```
*       TMTOT - NUMBER OF BUILDINGS IN MUNITION STORAGE AREA
*       VOL - MUNITION PACKAGE VOLUME
*       VVOL - MUNITION PACKAGE VOLUME
*                                                                        *
****************                SUBROUTINE NAMES          ******************
*                                                                        *
*       CALLED BY:  FORM
*                                                                        *
*       CALLS:  NSORT - SORTS CLASS AND ASSOCIATED BUILDING NEW IN
*                       DESCENDING (NEW) ORDER
*                                                                        *
**************************************************************************
*                                                                        *
        COMMON /PT2/ MITOT,MGP(7,11),MGPCNT
        COMMON /PT4/ TMTOT,IRECTL
        INTEGER SW5,ARNEW(7),ARCLASS(7),TMGP(7),ISUBGP*2,MUNR*4,
     &    SW6,TMKEY,TMTOT,GROUP*2,CLASS*1,CLASS1*1
        CHARACTER NSTKNR*18,ANUMB*6,BNUMB*6
        REAL NEW
        ICURGP = 0
        SW5 = 0
        SW6 = 0
        TMKEY = 1
        WRITE (6,1)
    2   READ (07'TMKEY) ITYPE,BNUMB,BVOL,(ARNEW(I),I=1,7)
        DO 13 I=1,7
        ARCLASS(I) = I
   13   CONTINUE
        CALL NSORT (ARNEW,ARCLASS)
        DO 3 J = 1,12
        ITMGP = 0
        DO 4 I = 1,7
        IF (MGP(I,J).EQ.0)  GO TO 4
        ITMGP = ITMGP + 1
        TMGP(ITMGP) = I
    4   CONTINUE
        IF (ITMGP.EQ.0) GO TO 3
        ICURGP = ITMGP
        DO 5 INEW = 1,7
        IF (ARNEW(8-INEW).EQ.0) GO TO 5
        SW5 = 0
        DO 6 ITMGP = 1,ICURGP
        IF (TMGP(ITMGP).NE.ARCLASS(8-INEW)) GO TO 6
        IREC = 1
   14   READ (10'IREC) CLASS1,MUNR,ANUMB,VOL
        IF (ANUMB.EQ.BNUMB.AND.ARCLASS(8-INEW).EQ.CLASS1) GO TO 10
    7   IREC = IREC + 1
        IF (IREC.LE.IRECTL) GO TO 14
        GO TO 6
   10   MIKEY = MUNR
        READ (02'MIKEY) LOT,NSTKNR,PACKAG,STACK,VVOL,NEW,MLF,GROUP,CLASS
        IF (GROUP.NE.J) GO TO 7
        SW6 = 0
        DO 12 ICK=INEW,7
        IF(CLASS.EQ.ARCLASS(8-ICK)) SW6=1
   12   CONTINUE
        IF (SW6.EQ.0) GO TO 7
        IF (SW5.NE.0) GO TO 8
```

```
        SW5 = 1
        WRITE (08,20) BNUMB,J,TMGP(ITMGP)
        WRITE (08,9)BNUMB,J,TMGP(ITMGP),MUNR.CLASS1,BNUMB,NEW
        GO TO 7
    8   WRITE (08,11) MUNR,CLASS1,BNUMB,NEW
        GO TO 7
    6   CONTINUE
    5   CONTINUE
    3   CONTINUE
        TMKEY = TMKEY + 1
        IF (TMKEY.LE.TMTOT) GO TO 2
*                                                                       *
    1   FORMAT (/5X,"GENERATING GROUP AND SUBGROUP CONSTRAINTS NOW...."/)
*                                                                       *
*     SORTS THE CLASS ID'S AND ASSOCIATED BUILDING NEW'S FOR CURRENT
*     BUILDING BY THE NEW, IN DESCENDING ORDER
    9   FORMAT ("MATRIX BLDG  :",A6,":G",I2," C",I1,"(P),MU",I4,
     &    ":C",I1," BD :",A6,"=",F10.4)
*                                                                       *
   11   FORMAT (7X,",MU",I4,":C",I1," BD :",A6,"=",F10.4)
*                                                                       *
   20   FORMAT ("MATRIX OBJECT:IVE,SSBLDG:",A6,":G",I2," C",I1,"=+0.")
      RETURN
      END
*                                                                       *
****************                END BLDNEW            ******************
```

```
*************************************************************************
*                                                                       *
      SUBROUTINE NSORT (ARNEW,ARCLASS)
*                                                                       *
*************************************************************************
*                                                                       *
*************              PROGRAM IDENTIFICATION            ************
*                                                                       *
*      SORTS THE CLASS ID'S AND ASSOCIATED BUILDING NEW'S
*      ACCORDING TO THE NEW VALUE IN DESCENDING ORDER
*                                                                       *
*************************************************************************
*                                                                       *
*************************************************************************
*                                                                       *
*************              VARIABLE IDENTIFICATION           ************
*                                                                       *
*      ARCLASS - ARRAY OF CLASS ID'S FOR THE CURRENT GROUP
*      ARNEW - ARRAY OF NEW'S MATCHING THE CLASS ID'S FOR THE CURRENT GROUP
*      ATCLASS - TEMPORARY HOLDING AREA
*      ATNEW - TEMPORARY HOLDING AREA
*      IC - INDEX
*      IH - INDEX
*                                                                       *
****************              SUBROUTINE NAMES           ****************
*                                                                       *
*      CALLED BY:  BLDNEW
*                                                                       *
*      CALLS:  NONE
*                                                                       *
*************************************************************************
*                                                                       *
       INTEGER ARNEW(7),ARCLASS(7),ATCLASS,ATNEW
       DO 1 IC=1,6
       DO 2 IH=IC,6
       IF (ARNEW(IC).GT.ARNEW(IH+1)) GO TO 2
       ATNEW = ARNEW(IC)
       ATCLASS = ARCLASS(IC)
       ARNEW(IC) = ARNEW(IH+1)
       ARCLASS(IC) = ARCLASS(IH+1)
       ARNEW(IH+1) = ATNEW
       ARCLASS(IH+1) = ATCLASS
    2  CONTINUE
    1  CONTINUE
      RETURN
      END
*                                                                       *
****************                 END NSORT               ****************
```

```
*************************************************************************
*                                                                       *
      SUBROUTINE RHANDS
*                                                                       *
*************************************************************************
*                                                                       *
*************             PROGRAM IDENTIFICATION             *************
*                                                                       *
*      GENERATES THE RIGHT HAND SIDE (RHS) VALUES FOR THE MUNITION,
*      VOLUME, AND SPECIAL SET CONSTRAINTS AND WRITES THEM TO LPINFO
*                                                                       *
*************************************************************************
*                                                                       *
*************************************************************************
*                                                                       *
*************             VARIABLE IDENTIFICATION            *************
*                                                                       *
*      BNUMB - BUILDING NUMBER
*      BVOL - BUILDING VOLUME
*      C1 - BUILDING NEW FOR CLASS/DIVISION 1.1
*      C1218 - BUILDING NEW FOR CLASS/DIVISION/CAT 1.2 18
*      C1212 - BUILDING NEW FOR CLASS/DIVISION/CAT 1.2 12
*      C1208 - BUILDING NEW FOR CLASS/DIVISION/CAT 1.2 08
*      C1204 - BUILDING NEW FOR CLASS/DIVISION/CAT 1.2 04
*      C13 - BUILDING NEW FOR CLASS/DIVISION 1.3
*      C14 - BUILDING NEW FOR CLASS/DIVISION 1.4
*      ICL - MUNITION CLASS ID
*      IGP - MUNITION GROUP ID
*      IROW - NUMBER OF CONSTRAINTS
*      ITYPE - STANDARD BUILDING TYPE
*      LOT - INTERNAL MUNITION LOT ID
*      MIKEY - INDEX KEY FOR MUNITION INVENTORY WORK FILE
*      MITOT - NUMBER OF MUNITION/LOT COMBINATIONS IN INVENTORY
*      MLF - MUNITION DENSITY FACTOR
*      MUNR - INTERNAL MUNITION/LOT ID
*      NEW - MUNITION PACKAGE NET EXPLOSIVE WEIGHT
*      NSTKNR - MUNITION NATIONAL STOCK NUMBER
*      PACKAG - NUMBER OF PACKAGES IN CURRENT MUNITION/LOT
*      STACK - STACKING HEIGHT (MEASURED IN PACKAGES)
*      TMKEY - INDEX KEY FOR BUILDING WORK FILE
*      TMTOT - NUMBER OF BUILDINGS IN MUNITION STORAGE AREA
*      VOL - MUNITION PACKAGE VOLUME
*                                                                       *
****************             SUBROUTINE NAMES            *****************
*                                                                       *
*      CALLED BY:  FORM
*                                                                       *
*      CALLS: NONE
*                                                                       *
*************************************************************************
*                                                                       *
       COMMON /PT2/ MITOT,MGP,MGPCNT
       COMMON /PT4/ TMTOT,IRECTL
       COMMON /PT5/ IROW,ISS
       INTEGER TMTOT,TMKEY,MUNR*4,PACKAG*4
       INTEGER *7 C1,C1218,C1212,C1208,C1204,C13,C14
       CHARACTER NSTKNR*18,BNUMB*6
       MUNR = 1
```

```
          MIKEY = 1
          IROW= 0
          WRITE (6,1)
          WRITE (08,2)
*                                                                        *
*       WRITES RIGHT HAND SIDE VALUES OF MUNITION CONSTRAINTS
*       TO LPINFO
*                                                                        *
    4     READ (02'MIKEY) LOT,NSTKNR,PACKAG,STACK,VOL,NEW,MLF,IGP,ICL
          WRITE (08,3) MUNR,PACKAG
          MUNR = MUNR + 1
          IROW = IROW + 1
          MIKEY = MIKEY + 1
          IF (MIKEY.LE.MITOT) GO TO 4
          TMKEY = 1
*                                                                        *
*       WRITES RIGHT HAND SIDE VALUES FOR THE BUILDING VOLUME
*       AND SPECIAL SET VARIABLE CONSTRAINTS TO LPINFO
*                                                                        *
    5     READ (07'TMKEY) ITYPE,BNUMB,BVOL,C1,C1218,C1212,C1208,C1204,C13,C14
          WRITE (08,6) BNUMB,BVOL
          WRITE (08,7) BNUMB
          TMKEY = TMKEY + 1
          IROW = IROW + 2
          IF (TMKEY.LE.TMTOT) GO TO 5
          WRITE  (08,20)
*                                                                        *
   20     FORMAT ("END***")
*                                                                        *
    1     FORMAT (/5X,"CONCLUDING BY GENERATING RHS..."/)
*                                                                        *
    2     FORMAT ("RHS    OBJECT:IVE,RHS=0")
*                                                                        *
    3     FORMAT (7X,"MU",I4,"=",I4)
*                                                                        *
    6     FORMAT (7X,"BLDG  :",A6,":    VOL=",F12.3)
*                                                                        *
    7     FORMAT (7X,"SSET:",A6,"=1")
        RETURN
        END
*                                                                        *
****************                END RHANDS               ******************
```

```
*************************************************************************
*                                                                       *
      SUBROUTINE CLOSE (SW3)
*                                                                       *
*************************************************************************
*                                                                       *
*************              PROGRAM IDENTIFICATION            *************
*                                                                       *
*       IF TOO MANY (>4095) CONSTRAINTS OR (>262K) DECISION
*       VARIABLES ARE GENERATED PRINTS APPROPRIATE ERROR MESSAGE
*                                                                       *
*       PRINTS MUNITION, GROUP, AND CLASS CROSS REFERENCE LISTS
*                                                                       *
*       PRINTS NUMBER OF DECISION VARIABLES AND CONSTRAINTS USED
*                                                                       *
*************************************************************************
*                                                                       *
*************************************************************************
*                                                                       *
*************              VARIABLE IDENTIFICATION           *************
*                                                                       *
*       CLASS - MUNITION CLASS ID
*       GROUP - MUNITION GROUP ID
*       I - INDEX
*       IRECTL - NUMBER OF DECISION VARIABLES
*       IROW - NUMBER OF CONSTRAINTS
*       ISS - NUMBER OF SPECIAL SET VARIABLES
*       SW3 - ERROR SWITCH: 0 - NO ERROR, 1 - ERROR
*       WORD - CROSS REFERENCE OUTPUT RECORD
*                                                                       *
****************              SUBROUTINE NAMES          *****************
*                                                                       *
*       CALLED BY:  MAIN
*                                                                       *
*       CALLS:  NONE
*                                                                       *
*************************************************************************
*                                                                       *
        COMMON /PT2/ MITOT,MGP,MGPCNT
        COMMON /PT4/ TMTOT,IRECTL
        COMMON /PT5/ IROW ISS
        INTEGER TMTOT,SW3
        CHARACTER WORD*70
        CHARACTER GROUP*1(12)/"A","B","C","D","E","F","G","H","J","K",
     &    "L","S"/
        CHARACTER CLASS*6(7)/"1.1","1.2/18","1.2/12","1.2/08","1.2/04",
     &    "1.3","1.4"/
        SW3 = 0
        IF ((IRECTL+ISS+MITOT).LE.262000) GO TO 15
        SW3 = 1
        WRITE (6,16) IRECTL+ISS+MITOT
   15   IF ((IROW+ISS).LE.4095) GO TO 12
        SW3 = 1
        WRITE (6,13) IROW+ISS
   12   REWIND 11
        WRITE (6,11)
    3   READ (11,1,END=2) WORD
        WRITE (6,1) WORD
```

```
          GO TO 3
     2    WRITE (6,4)
          BACKSPACE 11
          WRITE (11,4)
          DO 5 I=1,12
          WRITE (6,6) I,GROUP(I)
          WRITE (11,6) I,GROUP(I)
     5    CONTINUE
          WRITE (6,8)
          WRITE (11,8)
          DO 7 I=1,7
          WRITE (6,9) I,CLASS(I)
          WRITE (11,9) I,CLASS(I)
     7    CONTINUE
          WRITE (6,10) IRECTL+ISS+MITOT,IROW+ISS
          WRITE (11,10) IRECTL+ISS+MITOT,IROW+ISS
    14    CALL DETACH (11,ISTA11, )
*                                                                      *
     1    FORMAT (A70)
*                                                                      *
     4    FORMAT (//10X,"GROUP CROSS REFERENCE LIST",/16X,"ID NR    GROUP"//)
*                                                                      *
     6    FORMAT (18X,I2,7X,A1)
*                                                                      *
     8    FORMAT  (//10X,"CLASS CROSS REFERENCE LIST",/16X,
     &      "ID NR    CLASS"//)
*                                                                      *
     9    FORMAT (18X,I2,5X,A6)
*                                                                      *
    10    FORMAT (//5X "THIS PROBLEM CONTAINS ",I8," DECISION VARIABLES",
     &      " IN THE ",/10X,"OBJECTIVE FUNCTION AND ",I4," CONSTRAINTS"//)
*                                                                      *
    11    FORMAT (//)
*                                                                      *
    13    FORMAT (/5X,"THE MAXIMUM NUMBER OF CONSTRAINTS (4095) HAS BEEN",
     &      /10X,"EXCEEDED BY THIS PROBLEM, CONTAINING ",I4," CONSTRAINTS"/)
*                                                                      *
    16    FORMAT (/5X,"THE MAXIMUM NUMBER OF DECISION VARIABLES (262K)",
     &      /"HAS BEEN EXCEEDED BY THIS PROBLEM, CONTAINING",
     &      /I7," DECISION VARIABLES"/)
        RETURN
        END
*                                                                      *
****************             END CLOSE              *****************
```

```
*************************************************************************
*                                                                       *
      SUBROUTINE SPAWN
*                                                                       *
*************************************************************************
*                                                                       *
*************             PROGRAM IDENTIFICATION            *************
*                                                                       *
*       COMPUTES THE TIME, CORE, AND FILE SPACE REQUIREMENTS FOR
*       PROGRAM EXECUTION
*                                                                       *
*       CREATES NECESSARY JCL
*                                                                       *
*       SUBMITS JCL JOB TO THE BATCH WORLD
*                                                                       *
*       COMPUTER DISPLAYS THE SNUMB NUMBER
*                                                                       *
*************************************************************************
*                                                                       *
*************************************************************************
*                                                                       *
*************            VARIABLE IDENTIFICATION            *************
*                                                                       *
*      ICORE - AMOUNT OF CORE NEEDED FOR THE LP600 JOB
*      IDISK - AMOUNT OF TEMPORARY WORK SPACE NEEDED BY THE LP600 JOB
*      INDISK - AMOUNT OF WORK SPACE ASSIGNED TO INDIVIDUAL DISKS
*      IPAGE - MAXIMUM NUMBER OF PRINT LINES OF OUTPUT ALLOWED
*      ITIME - AMOUNT OF TIME ALLOCATED FOR THE LP600 JOB
*                                                                       *
****************              SUBROUTINE NAMES            ****************
*                                                                       *
*       CALLED BY: MAIN
*                                                                       *
*       CALLS:
*               CALLSS - HONEYWELL SYSTEM ROUTINE THAT ALLOWS FOR SPECIAL
*                        OPERATIONS -- USED TO SUBMIT LP600 JCL JOB
*               DETACH - RELEASES SPECIFIED FILE FROM PROGRAM CONTROL
*                                                                       *
*************************************************************************
*                                                                       *
        COMMON /PT5/ IROW,ISS
        WRITE (6,1)
        IPAGE = 5
        ICORE = (14*IROW+29000)/1000
        IDISK = ICORE * 50 / 28
        INDISK = IDISK / 5
        ITIME = ICORE*5/28
        WRITE (09,2) ITIME+1000,ICORE+1000,IPAGE+1000,
     &   INDISK+1000,INDISK+1000,INDISK+1000,INDISK+1000,INDISK+1000
        WRITE (09,3)
        WRITE (09,4)
```

```
  1481         REWIND 09
  1482         CALL CALLSS ("RUN 09#","CDIN")
  1483         CALL DETACH (09,ISTAT9, )
  1484*                                                                       *
  1485    1    FORMAT (/5X,"SPAWNING THE LP JOB NOW...",//20X,"BYE"//)
  1486*                                                                       *
  1487    2    FORMAT ("##A,J ;,8,16"/,
$ 1488    &      "$        IDENT   WP0354,AFIT,GUSMUS,MUNITION OPTIMIZATION"/,
  1489    &      "$        ENTRY   .LHSF"/,"$       USE     .LHSF"/,
  1490    &      "$        EXECUTE"/,"$      LIMITS  ",I3,",",I3,"K,,"I3,"K"/,
  1491    &      "$        PRMFL   H*,R,R,AF.LIB/LP.PAC"/,
$ 1492    &      "$        PRMFL   SO,W,L,79C06/DATA/LPOUT"/,
  1493    &      "$        DISC    AA,A1,",I2,"R"/,
  1494    &      "$        DISC    AB,A2,",I2,"R"/,
  1495    &      "$        DISC    AC,A3,",I2,"R"/,
  1496    &      "$        DISC    AD,A4,",I2,"R"/,
  1497    &      "$        DISC    AE,A5,",I2,"R")
  1498*                                                                       *
$ 1499    3    FORMAT ("$       PRMFL   IN,R,L,79C06/DATA/LPINFO"/,
  1500    &      "$        DATA    I*"/,"         PREPROCESS"/,
  1501    &      "          TITLE   MUNITION STORAGE OPTIMIZING"/,
  1502    &      "          SET     NOSOXO=ON"/,
  1503    &      "          CONVERT SOURCE=AMMO/IN,IDENT=MU"/,
  1504    &      "          SETUP   SOURCE=MU"/,
  1505    &      "          SET     OBJ=OBJECT:IVE,RHS=RHS"/,
  1506    &      "          SET     SCALE=-1"/)
  1507*                                                                       *
  1508    4    FORMAT ("        CRASH"/,"        INTEGER"/,"        RESET   NOSOXO"/,
  1509    &      "          OUTPUT"/,"        ENDLP"/,
  1510    &      "          EXECUTE"/,
  1511    &      "$        CONVER  NSPIN"/,
  1512    &      "$        LIMITS  1,6K,,2K"/,
$ 1513    &      "$        PRMFL   IN,R,L,79C06/DATA/LPOUT"/,
  1514    &      "$        REMOTE  OT"/,
  1515    &      "$        CONVER  NSPIN"/,
  1516    &      "$        LIMITS  1,6K,,2K"/,
$ 1517    &      "$        PRMFL   IN,R,L,79C06/DATA/CRSREF"/,
  1518    &      "$        REMOTE  OT"/,"$      ENDJOB"/,"***EOF")
  1519       RETURN
  1520*                                                                       *
  1521****************           END SPAWN                   ******************
  1522       END
```

NOTE: Cards identified with a $ in left margin will have be changed for new users. The source card numbers are 0184, 0187, 0188, 0191, 0196, 1488, 1492, 1499, 1513, and 1517.

VITA

Barton Allen Boggs was born in Toccoa, Georgia on September 10, 1950. He graduated from high school at Faith Academy, Manila, Philippines in 1968 and then attended Bryan College in Dayton, Tennessee from which he received his BA degree in Mathematics in May 1972. He then enlisted in the Air Force and served for two years as a Medical Administrative Specialist at Andrews AFB, Maryland. He was selected through the AECP to attend OTS and was commissioned in October 1974. Following completion of the Electronic Systems Officer course at Keesler AFB, Mississippi, he was assigned as a maintenance officer to the 729th Tactical Control Squadron at MacDill AFB, Florida. He remained at this assignment until entering the School of Engineering, Air Force Institute of Technology, in September 1978.

Permanent address: 105 Morningside Drive
Butler, Pennsylvania 16001

VITA

Louis Michael Gusmus was born on 16 January 1948 in Tuscumbia, Alabama. He graduated from Deshler High School in Tuscumbia in 1966. He attended the University of Alabama for two years, then enlisted and served a four year tour in the Air Force as a Refrigeration and Air Conditioning Specialist at Hurlbert AFB, Florida. After separating from the Air Force, he continued his education at the University of Southern Mississippi, earning a BS degree in Computer Science in May 1974. He received a commission in the Air Force through the AFROTC program. Upon graduation, he entered active duty as a computer Systems Analyst at Air Force Systems Command Headquarters, Andrews AFB, Maryland. He remained at this assignment until entering the School of Engineering, Air Force Institute of Technology, in June of 1978.

Permanent address: 700 North Commons
Tuscumbia, Alabama 35674

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/GSM/SM/79D-15	2. GOVT ACCESSION NO. AD-A083 708	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) OPTIMIZATION OF MUNITIONS STORAGE		5. TYPE OF REPORT & PERIOD COVERED Masters Thesis
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Barton A. Boggs, Capt USAF; Louis M. Gusmus, Capt USAF		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology (AFIT/EN) Wright-Patterson AFB, Ohio 45433		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Institute of Technology (AFIT/EN) Wright Patterson AFB, Ohio 45433		12. REPORT DATE December 1979
		13. NUMBER OF PAGES 314
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report)
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Approved for public release; IAW AFR 190-17

JOSEPH P. HIPPS, Major, USAF
Director of Information

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Munitions Storage
Explosives
Optimization
Mixed Integer Linear Programming
Quantity-Distance

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The problem of how to store as much as possible of a required munitions inventory in a given storage area is addressed by the formulation of a mixed integer linear programming model that will calculate optimal storage subject to a complex set of constraints. The Munitions Storage Optimization System (MSOS) provides a capability for setting up munitions storage optimization linear programming problems. MSOS allows the user to create data bases containing the required information for the munitions inventory items and the storage buildings. Munitions inventory is entered by stock number, number → next page

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered) cont.

of lots, and number of packages for each lot. A program extracts the necessary information from the data bases, formulates the objective function and constraint equations, then submits the problem to a mixed integer linear programming package for calculation of the solution.