

AD-A095 729

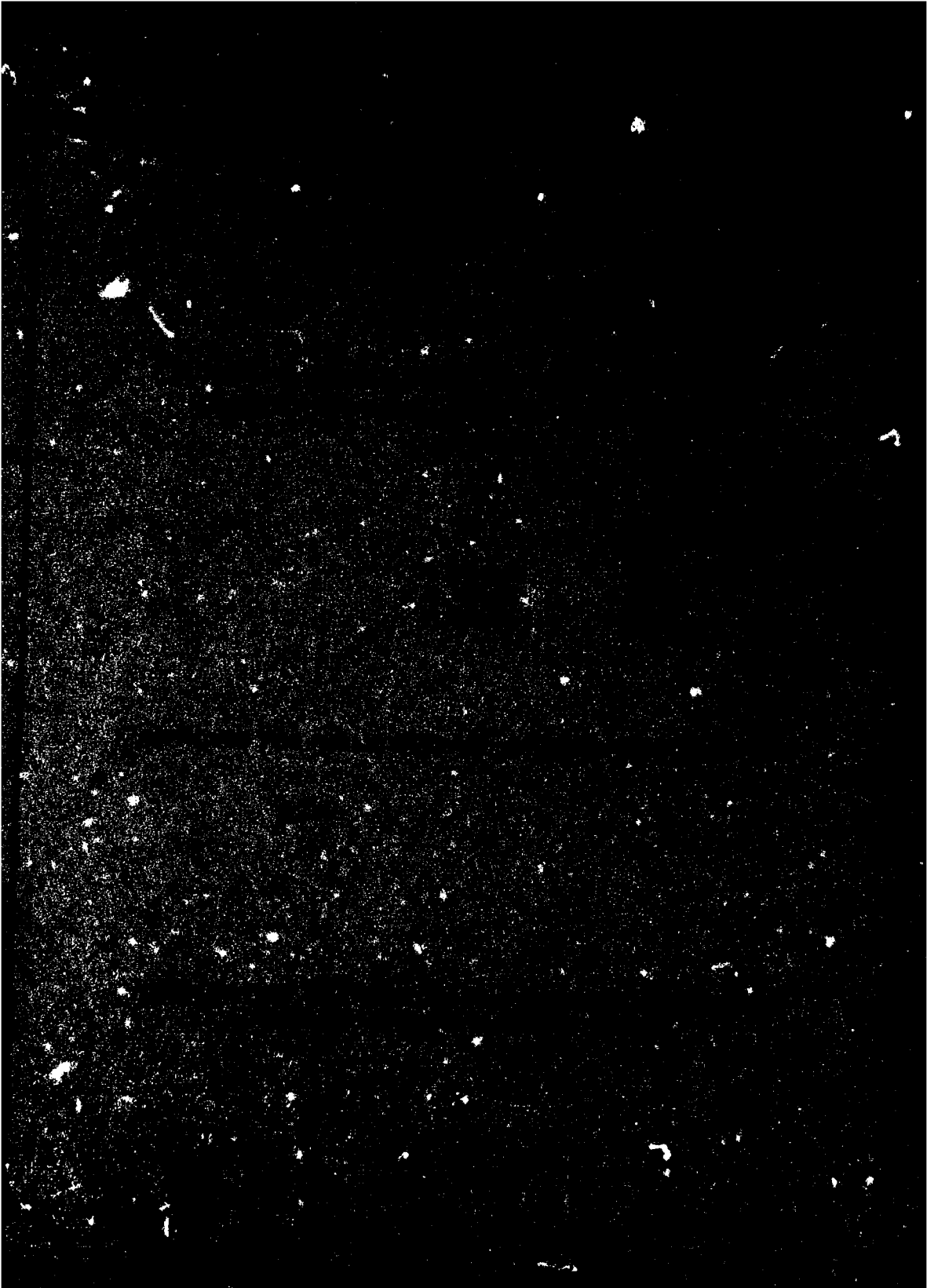
DAVID W TAYLOR NAVAL SHIP RESEARCH AND DEVELOPMENT CE--ETC F/G 12/2
AUTOMATIC VEHICLE SCHEDULING (AVS) PROGRAMMER'S INSTRUCTION MAN--ETC(U)
FEB 81 R WINCHELL, R MELTON, M NATRELLA
DTNSRDC-81/017

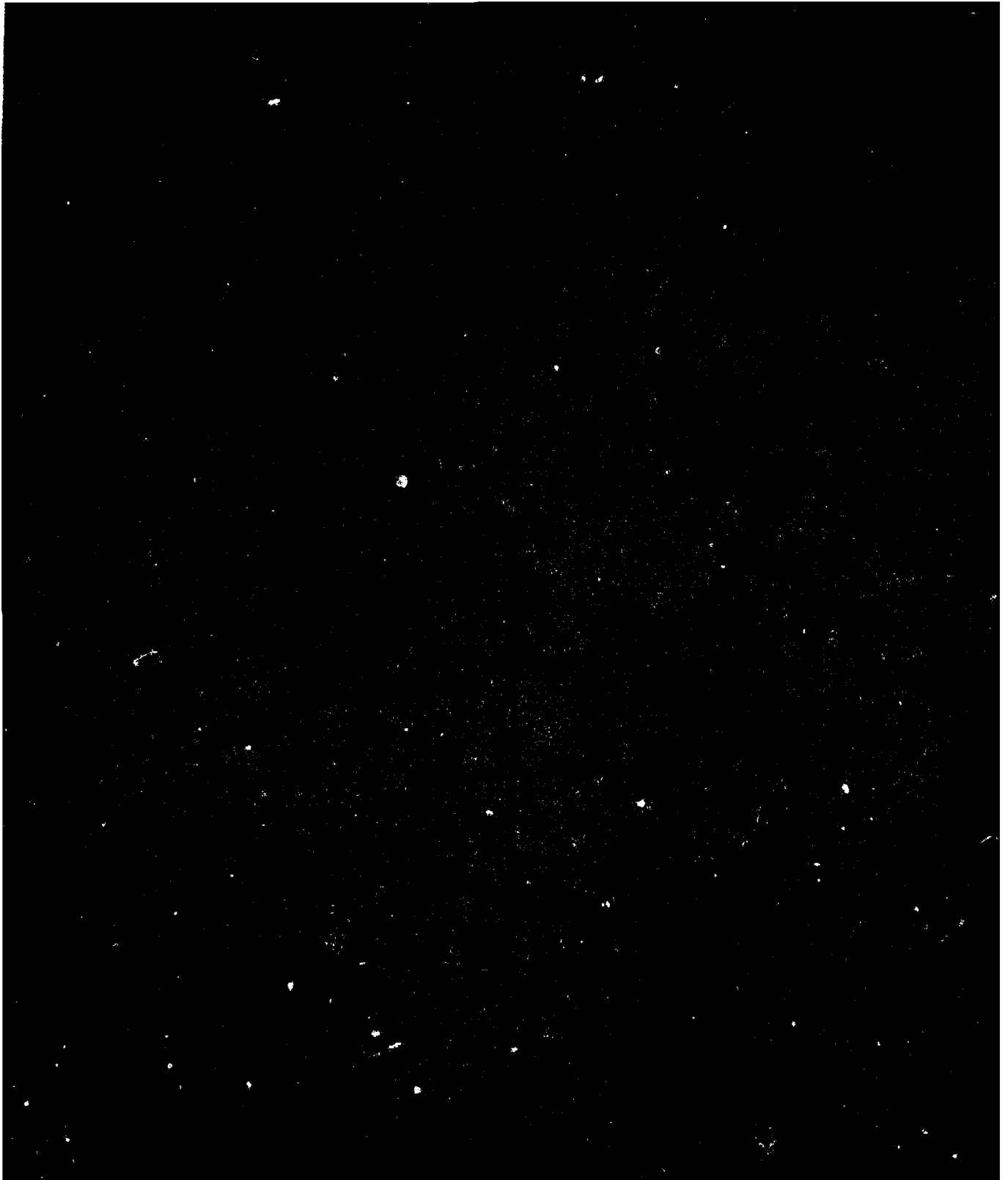
UNCLASSIFIED

NL

1 1/2
36(1)-129







UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER DTNSRDC-81/017	2. GOVT ACCESSION NO. AD-A095729	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) AUTOMATIC VEHICLE SCHEDULING (AVS) PROGRAMMER'S INSTRUCTION MANUAL FOR THE BURROUGHS B3500 COMPUTERS		5. TYPE OF REPORT & PERIOD COVERED Final Report
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) R. Winchell R. Melton M. Natrella		8. CONTRACT OR GRANT NUMBER(s) 121201
9. PERFORMING ORGANIZATION NAME AND ADDRESS David W. Taylor Naval Ship Research and Development Center Bethesda, Maryland 20084		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Work Unit 1800-007
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE February 1981
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		13. NUMBER OF PAGES 197
		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		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		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) AVS, warehousing, scheduling, straddle trucks, transporters, industrial tractors, NSC		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) Automated Vehicle Scheduling (AVS) is a software package designed to assist in scheduling palletized cargo delivery to warehouses in a Navy Supply Center. The package consists of two scheduling programs, which schedule regular and emergency orders, respectively, and a transaction history file update/report program.		

DD FORM 1473
1 JAN 73

EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

TABLE OF CONTENTS

	Page
LIST OF FIGURES	iv
TABLE	iv
ABSTRACT	1
ADMINISTRATIVE INFORMATION	1
INTRODUCTION	1
BACKGROUND	3
AVS1 - REGULAR ORDER SCHEDULING	8
INPUT	8
METHOD AND ALGORITHM	10
PROGRAM OUTPUT	11
AVS2 - EMERGENCY ORDER SCHEDULING	14
INPUT	14
METHOD AND ALGORITHM	15
PROGRAM OUTPUT	16
SPECIAL TECHNIQUES	17
TRAVEL TIME TECHNIQUES	17
LINKED LIST TECHNIQUES	18
DATA PACKING	23
AVS3 - HISTORY/UPDATE PROGRAM	24
INPUT	29
PROGRAM OUTPUT	29
ACKNOWLEDGMENT	31
APPENDIX A - USER'S MANUAL	33
APPENDIX B - PROGRAMMER'S GUIDE - SUBROUTINE FLOWCHARTS/LISTINGS	49
APPENDIX C - AVSIN1, A SINR COBOL DRIVER ROUTINE, LISITNGS	123
APPENDIX D - AVS3, HISTORY/UPDATE, FLOWCHARTS AND LISTINGS	139
APPENDIX E - SAMPLE RUNS	177

LIST OF FIGURES

	Page
1 - Map of Charleston Navy Base - AVS Serviced Warehouses	7
2 - AVS System Flow Chart	9
3 - All Vehicle Types - Order Allocations	12
4 - Straddle Order Allocations	12
5 - Transporter Order Allocations	12
6 - Tractor Trailers/Industrial Tractors Order Allocation	13
7 - Methods of Data Storage, Duplicate Arrays	20
8 - Methods of Data Storage, Sequential Sort	21
9 - Methods of Data Storage, Linked Lists	22
10 - Individual Entry Update.	27
11 - Record Consolidation	27
12 - Update From AVS Schedule	28
13 - AVS3 Frame Description	30
14 - Frame A01	38
15 - Frame A02	39
16 - Frame A03	40
17 - Frame A04	41
18 - Frame A05	44
19 - Frame A07	47

Table 1 - NSC Charleston Warehouses Serviced by AVS Listed by Group and Number Within Group.	5
---------------------------------------------------------------------------------------------------------	---

ABSTRACT

Automated Vehicle Scheduling (AVS) is a software package designed to assist in scheduling palletized cargo delivery to warehouses in a Navy Supply Center. The package consists of two scheduling programs, which schedule regular and emergency orders, respectively, and a transaction history file update/report program.

ADMINISTRATIVE INFORMATION

This study was authorized by the Naval Supply Systems Command with funding under Task Area 53/531/091.

INTRODUCTION

Automated Vehicle Scheduling (AVS) is a software package designed to assist in scheduling palletized cargo delivery among warehouses in a Navy Supply Center (NSC). The package consists of two scheduling programs, AVS1 and AVS2, which schedule, respectively, regular and emergency orders, and a transaction history file update/report program, AVS3. They are written in FORTRAN and are designed to run on Burroughs B3500 computers at NSC, Charleston, S.C. and at the Fleet Material Support Office, Mechanicsburg, PA. An earlier version was designed for the CDC 6600 computer at DTNSRDC.

AVS can schedule up to 99 orders totalling about 2000 pallets among as many as 99 warehouses. Deliveries and/or pickups are made by as many as 50 vehicles of four general types: straddle trucks, transporter vehicles, tractor trailers, and industrial tractors. Routes are built to "maximum" efficiency within the limitations of the algorithm used.

AVS2 uses the routes prepared by AVS1 to schedule servicing of emergency orders placed during the regular daily routine. An emergency order can include from 1 to 99 pallets; it can preempt regular orders if the dispatcher desires; it can be handled by a single vehicle type or by a mix of vehicles; finally, the vehicles selected to service it may be those used for regular orders, a subset of these, vehicles previously

unused, or any combination of these vehicles. As many as 99 emergency orders may be considered in the same AVS2 run.

For AVS to be successful, the programs must be easily usable by dispatch personnel who have had minimal computer training. In addition, the scheduling programs must execute rapidly to assure fast response to orders. For these reasons the AVS programs are interactive, tutorial, and corrective, using cathode ray tube (CRT) terminals connected to the B3500. Program procedures, execution instructions, and output file storage are simple. Data are requested from the user by the Information Retrieval System (SINR) and inputs are checked for validity by a COBOL driver program, AVSIN1. Instructions are available to the user in frame form displayed at the CRT terminal. Schedules are generated only after the user has checked the correctness of the data.

Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A	

BACKGROUND

Since the impetus for undertaking the AVS project came from NSC Charleston, the AVS programs described here address operations at that installation. A brief description of Charleston's local delivery procedures is given in this section.

NSC Charleston includes 78 pick-up/delivery sites and eight piers, plus six off-base sites (Table 1). Twenty or more of these are used in a typical half-day's schedule. The dispatch operation is run from building 1078. (A map of the Charleston complex is given in Figure 1; the warehouses are listed in Table 1.)

Orders for palletized cargo movement fall into three priority classes. Group 3 orders are telephoned to the dispatcher twice a day: at 1000 and 1500 hours. Group 2 and Group 1 are priority orders requiring service within 8 hours and 4 hours, respectively. They may be called in at any time, but in practice are usually phoned in at the same time as Group 3 orders. Orders are ready for shipment at the warehouses when the dispatcher is called to request transportation. At present the dispatch supervisor prepares the vehicle schedules from the order list using his knowledge of the base layout; there is no documented formal procedure. The vehicles are radio dispatched to service these requests.

However, there is additional cargo movement which is not handled in this way. At certain warehouses the high volume of cargo that is routinely shipped/received is moved by vehicles assigned exclusively to those locations. These movements will not be scheduled by AVS initially.

Orders are serviced by four types of vehicles: straddle trucks, transporter vehicles, conventional tractor trailers, and industrial tractors. These vehicles will be designated in the remainder of this report by the abbreviations ST, TR, TT, and IT, respectively. These vehicles are distinguished by their operational characteristics, such as highway speed, load time, manner of loading, and the skills and ratings of the drivers who operate them.

TT's carry from four to fourteen pallets, are capable of highway speeds, but are relatively slow at loading and unloading. They must be

backed up to a loading platform and loaded by forklifts. TT's are the only vehicles which service the off-base sites.

TR's carry either ten or twelve pallets, are somewhat slower than TT's, and are more efficient at loading. They drive up to a loading platform, the operator's cab swings out of the way, the height of the truck bed is adjusted, and pallets are loaded onto a gravity conveyor which delivers them to a roller bed in the truck. When gravity conveyors are not available at a site, forklifts must be used instead.

ST's carry five or seven pallets, are slower than TT's or TR's on the road, but are the most efficient at loading. Pallets are aligned at the pick-up site; the ST lowers a set of lifting rails which fit into channels on the sides of the pallets; the ST then lifts the pallets and drives off. The procedure is reversed for unloading. ST's can service up to three warehouse origins per route segment, i.e., loading/unloading cycle.

IT's carry up to fourteen pallets, are slower than TT's, TR's, or ST's on the road. Loading and unloading are the same as for TT's. IT's are for use within the complex only.

The algorithm places minimum load requirements on each vehicle type for route assignment. These requirements are: ST, 3 pallets; TR, 8 pallets; TT, 14 pallets; and IT, 8 pallets.

TABLE 1 - NSC CHARLESTON WAREHOUSES SERVICED BY AVS
LISTED BY GROUP AND NUMBER WITHIN GROUP

Group		
Num	Name(s)	Activity
NORTH		
1	191	NSC
2	1601a	"
3	1601b	"
4	1602	"
5	1603	"
6	1604	"
7	1605	"
8	1606	B279
9	1621	NSC
10	1622	"
11	1628	"
12	A	"
13	1620	"
14	1157	"
CENTR		
15	SM,45	Serve Mart
16	46	6th Nav Dist
17	53C	"
18	64E	NSC
19	64W	"
20	66E	"
21	66W	"
22	67E	"
23	67W	"
24	198	"
25	1078	"
26	1127	"
27	1138	"
28	56	"
29	49	"
30	SF	"
31	SFR	"
NSYN		
32	2	USNSY
33	3	"
34	5	"
35	8	"
36	35	"
37	43C	"
38	44	"
39	59	"
40	223	"

Group		
Num	Names(s)	Activity
WEST		
41	1502	NSC
42	1503	"
43	1507	"
NSWT		
44	80	USNSY
45	177	"
46	1143	Spec Serv
47	1199	USNSY
NSYC		
48	98	NSC
49	187	USNSY
50	216	"
51	1175	"
52	1169	"
53	1171	"
54	1172	S.O.A.P.
55	1173	NSC
56	1174	USNSY
57	218	"
NSYS		
58	X10	AS17
59	193	NSC
60	224	USNSY
61	L	PIER
62	M	"
63	N	"
64	P	"
65	Q	"
66	R	"
67	S	"
68	T	"
69	X20	"
SOUTH		
70	30	RTC1
71	43S	not used
72	61	FBMTC
73	84	Comm.Ctr.
74	202	RTC1
75	646	USNS
76	647	"
77	655	Comm.Store
78	656	Navy Ex.
79	52	"

TABLE 1 (continued)

Group		
Num	Name(s)	Activity
MCRFT		
80	1	Mine Craft
81	7	"
82	16	"
83	23	"
84	26	"
85	53S	"
X54		
86	X54	Comm.Ctr.
OFF BASE		
87	ABASE	Air base
88	NWS	Nav Weap Sta
89	DEYTN	Deytens SY
90	BRASW	Braswell SY
91	CSNWS	ComStoreNWS
92	NMEDC	Nav Hosp

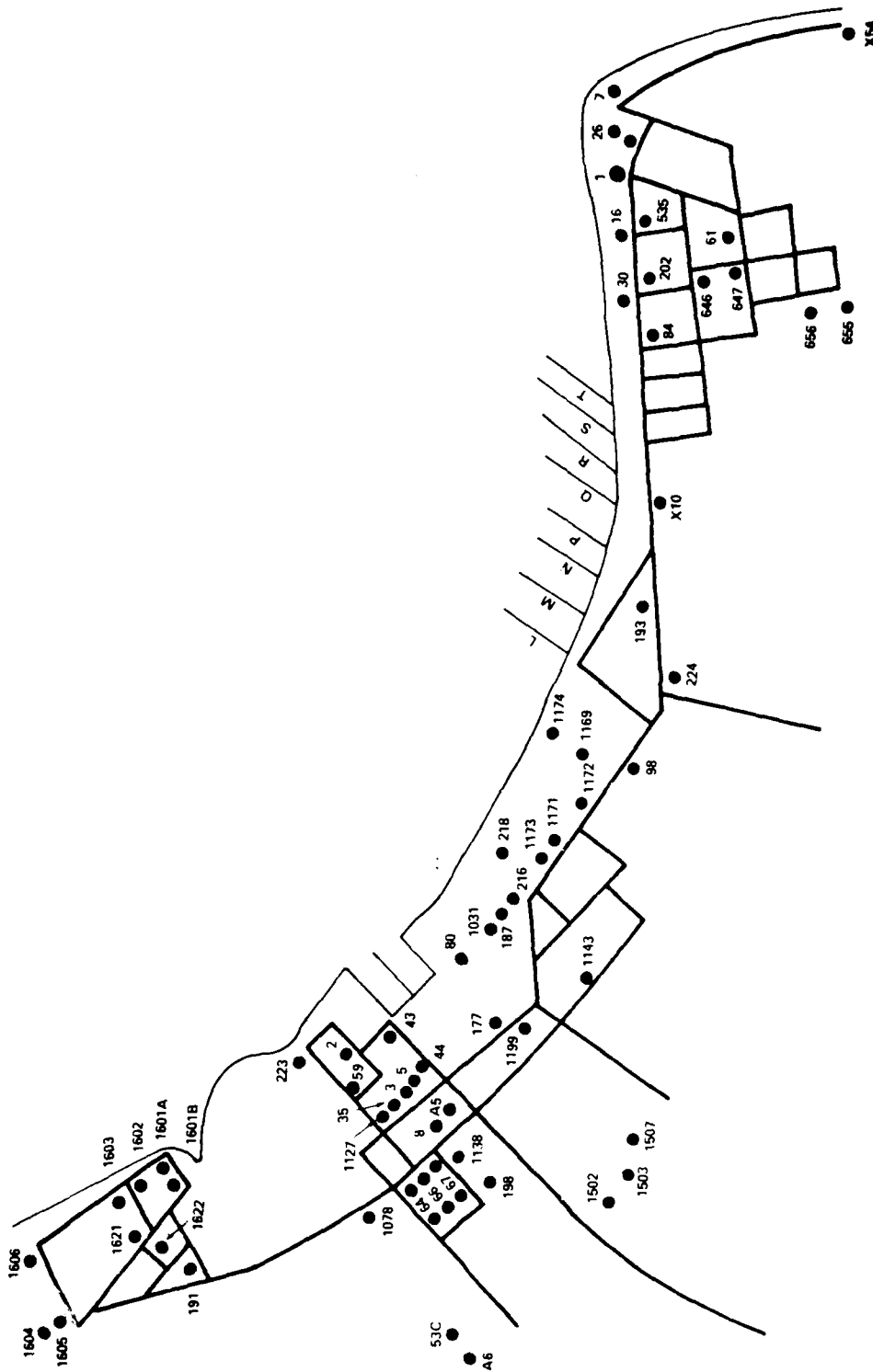


Figure 1 - Map of Charleston Navy Base - AVS Serviced Warhouses

AVS1 - REGULAR ORDER SCHEDULING

Regular order scheduling takes place in four phases. The first phase is interactive data entry from a remote CRT terminal (handled by the SINR Information Retrieval System available on the B3500 computer). In the second phase (subroutine AVSN2) the program examines the input orders individually and sorts them to reduce vehicle order selection time. In the third phase (subroutine ROUTE) the four vehicle-type order lists are assembled into vehicle routes. The last phase converts the vehicle route arrays into usable printout (subroutine TCARP). Figure 2 gives the AVS System Flowchart. The combined program, AVS, which consists of AVS1 and AVS2, uses 84 kilo digits (KD) of core locations.

INPUT

When procedures specified in the User's Manual (Appendix A) are followed, the data input to AVS1 is accomplished interactively from a remote CRT terminal. The user may enter the following data:

Orders. Orders are entered by listing the order sizes and originating and destination warehouses. Entries are made by "filling in the blanks". Data correctness messages are displayed on the CRT screen.

Vehicles. Vehicles are entered by listing the vehicle type, capacity, and maximum route duration. Omission of either capacity or maximum route duration for any vehicle will cause the algorithm to substitute default (built-in) values.

Begin Time. The beginning time for the schedules must be entered. The program uses 24-hour clock time.

Route Length. A maximum value for route duration in minutes is entered. This entry will replace a default value of 480 minutes. This route length does not supersede the value which is specified in the vehicle entry.

SINR's COBOL Driver Routine, AVSIN1, checks the input data for validity. An order's origin and destination warehouses must match a built-in list of warehouse names. Input numeric data (e.g., order size, times, vehicle capacity, and route duration) must be within specified ranges. The corrected data are made available to AVS through the VS2IN file. After all input has been entered satisfactorily, AVSIN1 will execute AVS.

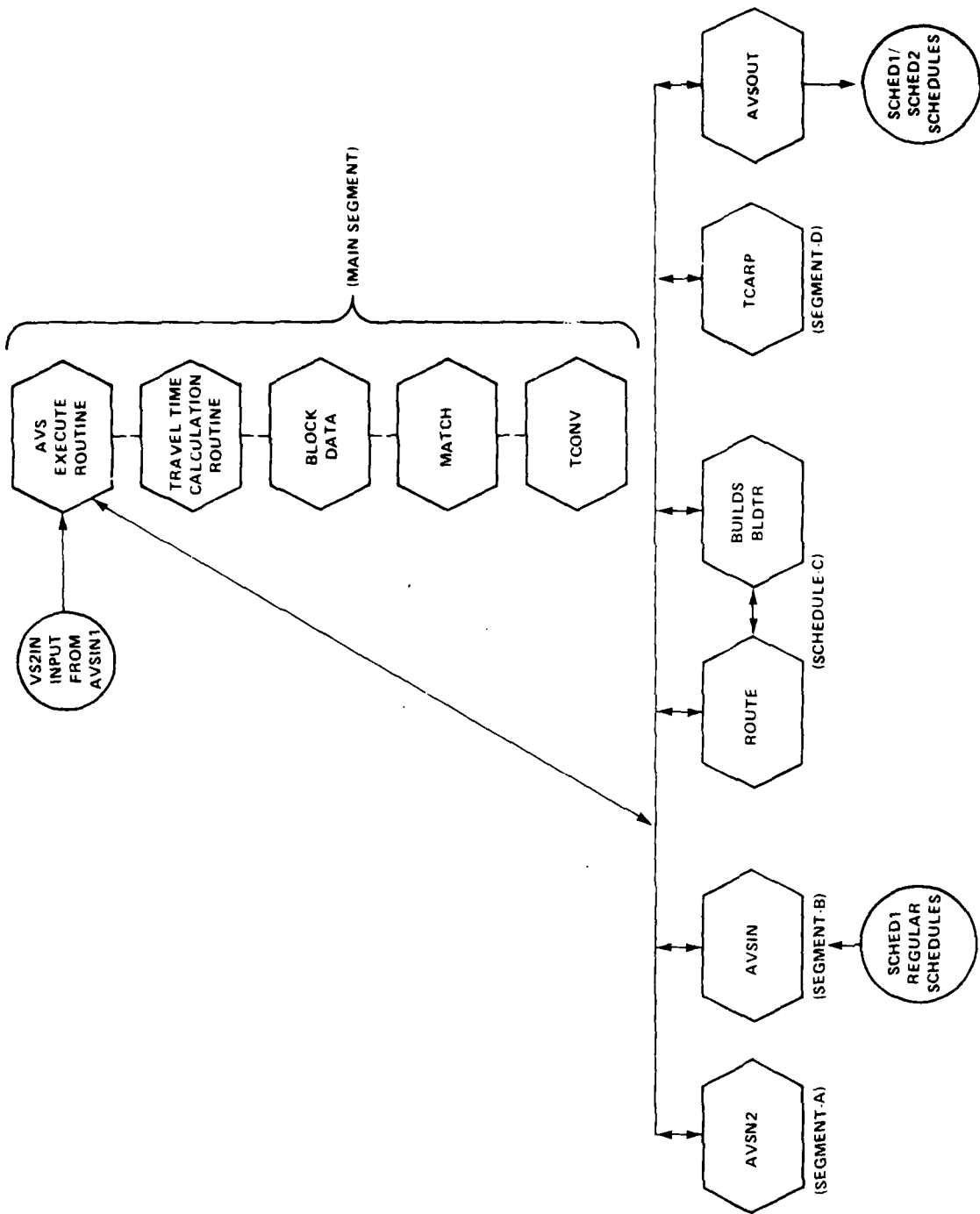


Figure 2 - AVS System Flow Chart

METHOD and ALGORITHM

Route building is accomplished by subroutines ROUTE, BLDTR, and BUILDS. The schedules for ST's are built first, then for TR's, TT's, and finally for IT's. The order of schedule building for the four vehicle types may be changed to fit the needs of the user. The algorithm operates on the sorted lists of orders. In the following discussion the ST routes are built first, then the variations used in TR, TT, and IT route building.

The list of orders is scanned to determine the combination of orders which will, if serviced by a single ST, provide the greatest time savings (or least time cost) over the situation in which each order is serviced by a separate truck. All time estimates pertain to warehouse areas, rather than to individual warehouses. There is almost always a time savings involved in joining two or more orders in this manner. However, to prevent excessive order joining and over-utilization of individual vehicles, a least time savings restriction was added to the algorithm. Since joined order routes are assigned to vehicles first, the least time saving restriction reduces the number of joined orders and allows the assignment of single order routes to available vehicles. If the minimum load requirement for the vehicle is not met, order segments are not joined.

Having selected the best set of orders to start an ST's route, the algorithm examines the remaining orders in the list for that single order which, if joined to the route, results in the least time cost over servicing the order separately. As in the starting case the limit on time cost applies. The new order is placed at the end the existing route, since examining intermediate positions along the route would be too time consuming and the coding would be too complex.

The algorithm continues in this manner, adding orders to the end of the previous route, until the route time limit for the vehicle precludes further additions, or until the pool of unassigned orders is exhausted. In the latter case the algorithm proceeds directly to consideration of the TR vehicles. In the former case the next ST route is

begun, using the same method. ST routes are built using a "first on, first off" strategy.

TR, TT, and IT building is exactly like ST route building and requires no additional elaboration. Leftover orders from TR route building are passed to the TT's and leftover TT orders are passed to IT's in the same manner that leftover ST orders are passed to TR's. Figure 3 shows beginning pairs that may be serviced by all vehicles. Figures 4 through 6 illustrate order assignment for each vehicle type. Routes for TR's, TT's, and IT's are built using a "first on, last off" scheme.

Since TT's are the only vehicles equipped for highway travel, they alone service the six off-base activities.

Route building ceases when all IT routes have been built. If any orders are still unserved, they are printed out so that the dispatcher can schedule them at a later time. They may be scheduled later as "emergency" orders using program AVS2, they may be postponed to the next shift, or they may require special scheduling without the use of AVS.

PROGRAM OUTPUT

Schedule output from AVS1 consists of a summary of the input data and the schedules for the individual vehicles. Each schedule gives the vehicle name, capacity, route starting time, and dates in a header; a list of scheduled stops specifying site, time, pallets picked up or delivered, reference order number, and approximate stay time at the site; and a trailer of finishing time and location, time still available, and number of pallets moved. AVS1 also creates a system schedule file, SCHED1, to be used by AVS2.

The coding for schedule printout is quite complex (see Special Techniques section). Changes to this coding should be made only after a thorough study of the programming details covered in Appendixes A and B.

- + PICK-UP
- DELIVERY
- # ORDER
- WAREHOUSE
- DIRECTION

ALL VEHICLE TYPES

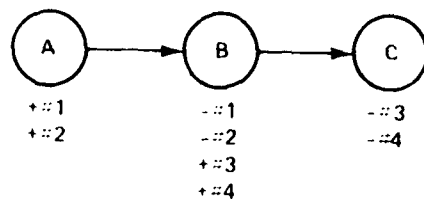


Figure 3 - All Vehicle Types, Order Allocations

STRADDLES

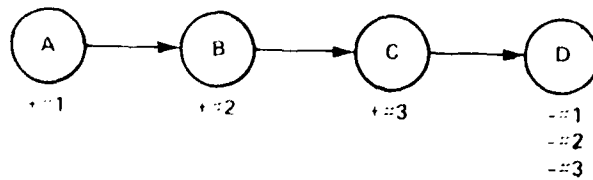


Figure 4 - Straddle Order Allocations

DIFFERENT ORIGINS SAME DESTINATION

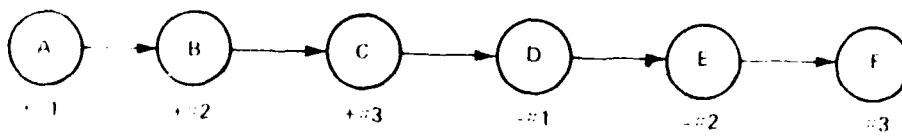


Figure 5 - Transporter Order Allocations

TRACTOR TRAILERS/INDUSTRIAL TRACTORS

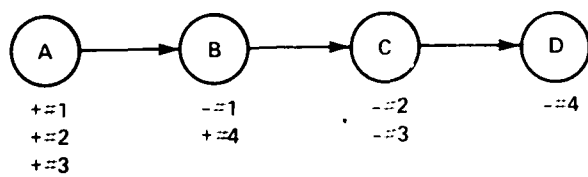


Figure 6 - Tractor Trailers/Industrial Tractors Order Allocation

AVS2 - EMERGENCY ORDERS SCHEDULING

The emergency order program AVS2 comprises three phases; data entry (AVSIN1-Appendix C), order scheduling (Subroutine Route), and schedule printout (Subroutine TCARP). These phases are described in the following paragraphs. Figure 2 gives the AVS2 system flowchart as a subset of AVS.

INPUT

AVS2 also uses SINR, and data items are entered by "filling in the blanks" as specified by the selected frames. AVSIN1, a SINR COBOL driver routine, consolidates the data and makes the results available to AVS2. The corrected data are passed to AVS2 on file, VS2IN. When data entry is complete, AVSIN1 executes AVS. Route data from the previous set of schedules, whether generated by AVS1 or by an earlier run of AVS2, are used by the program along with the current emergency order data as entered at the terminal. The following data are entered interactively:

Orders. The order origin, size, and destination are entered. Origin and destination are checked against a list of warehouse designations. Order size must be a numerical entry in the range 1-99 pallets. Up to 99 orders may be entered for emergency scheduling; if more than 99 are input, only the first 99 are retained.

Vehicles. A considerable choice of vehicles to service the emergency order is given to the user. All vehicles made available when the schedules were created by AVS1 (or augmented by earlier AVS2 runs) may be used; any subset of these may be chosen; or new vehicles may be selected. For new vehicles, capacity and maximum route duration may be specified; otherwise, default (built-in) values for capacity and maximum route duration are used. New vehicles chosen by the algorithm to service the order are added to the available vehicles list for the next use of AVS2, but if a vehicle just added by the user during data entry is not selected by the algorithm, it does not join the list.

Time. The time of the emergency order is input. This time is the basis on which the existing schedules are examined to determine vehicle availability. Consequently, sufficient lead time should be allowed to permit the program to execute and the dispatcher to notify the selected driver of his change in route.

Date. It is possible during AVS2 to change the date stored in the AVS1 schedules.

Bump Option. A "bump" option can be exercised to allow the servicing of emergency orders before regular orders.

METHOD AND ALGORITHM

AVS2 examines the existing vehicle schedules and determines which vehicle or vehicles should service an emergency order. There is a fundamental assumption that the vehicles are, in fact, following the computer generated schedules fairly closely. This assumption allows AVS2 to work with schedule data rather than with real time data.

The criterion for determining which vehicle(s) service an emergency order is quite simple, but complexities in the coding arise from a number of options designed to make the algorithm more flexible, and from the rather complicated method of storing the schedule data (linked list technique). According to the criterion, the emergency vehicle selected is the one which can pick up the emergency order the soonest, subject to restrictions imposed by the algorithm options.

The "bump" option affects this criterion. Under the bump option the user may allow vehicles to exceed their allotted maximum route duration. The default case for this option is not to permit this. Therefore, a vehicle which could service the emergency soonest would not be chosen if such an action meant that its regular route would not be finished on time. This action is altered by specifying the bump option during interactive data entry (see above and Appendix A, User's Manual). The term "bump" signifies that delivery of regular orders would be interrupted to handle the emergency. When time is clearly critical, specifying the bump option will enable the emergency order to receive the fastest possible service. If the bump option is not specified, and if the vehicles under consideration all have relatively full schedules, the program may inform the user that no vehicles are available to service the order.

An important feature of AVS2 is that the user may specify the vehicles to be considered, regardless of which vehicles were made available to the AVS1 algorithm. That is, the same vehicles may be used

as were used for the previous schedules; or a subset of those vehicles may be used; or additional vehicles may be specified. This gives the dispatcher considerable control over the manner in which the program schedules the emergency order. As an extreme example, the user could specify a single truck which was already in use, together with the bump option, to force the algorithm to fit the emergency into that vehicle's schedule.

Several points about the AVS2 algorithm need to be mentioned: First, the emergency order algorithm may be used any number of times during the processing of the AVS schedules. At the conclusion of each emergency run, the schedules are updated for use by the next run.

Second, but related to the first point, when the schedules are being searched for the placing of a new emergency order, no vehicle servicing a previous emergency order is available for the new order until the previous order is delivered. Emergency orders have a single priority, and are filled on a first come, first served basis.

This leads to a third point: emergency orders should be run in the order in which they are placed. Failure to do this may give erroneous results. Also the program will take a few minutes to run and print out, and this should be considered in specifying the start time for an emergency order.

Fourth, the actual updating of the schedules is not done automatically within the AVS2 program; consequently, if the schedules printed at the remote terminal do not satisfy the user, he may change options, vehicles, or even order data and rerun the program. Previous schedules may be saved or discarded by simple file handling commands given in Appendix A. The schedule may be modified using AVS3, Update Program. Detailed flowcharts and coding descriptions are found in the Appendixes.

PROGRAM OUTPUT

Schedule output for AVS2 is straightforward. A summary of the input data is provided, then a vehicle availability table. Any vehicles which cannot be used are listed and the actual vehicle(s) chosen is (are) given. Finally the new schedules for the chosen vehicle(s) are printed, following the same format as in AVS1.

SPECIAL TECHNIQUES

Several techniques used in the AVS programs will be described here to help in understanding the program coding. All the techniques were used to reduce execution time and core requirements so that the programs could run on the rather limited Burroughs B3500 computers in use by Navy Supply Centers. The penalty for the gain in efficiency of the programs is increased program complexity. Three techniques have to do with the calculation of travel times between warehouses; two are general data storage techniques used to reduce sort times in the AVS algorithms.

TRAVEL TIME TECHNIQUES

The AVS programs were set up to service up to 99 warehouses, and the test facility (NSC Charleston) has, in fact, 92 sites. The algorithms make frequent use of the travel times between sites. The times differ for the four vehicle types, giving more than 15,000 intra-activity time measurements. The prohibitive cost of storing such a collection of data demands that this figure be reduced to a more manageable level; it is this problem that the three techniques mentioned address.

The major reduction in the time array sizes is achieved by grouping the warehouse sites; each group of warehouses in close proximity is considered a single site (area). Figure 1 shows the groupings of the Charleston sites. (These groupings reflect some functional as well as geographic differentiation). The travel times between warehouses within an area are taken to be constant (two minutes).

A further reduction in the time array sizes is gained by considering the six off-base sites separately. These sites are serviced only by TT's and all movements take place between the main base and the sites; i.e., there are no movements between off-base locations. The number of measurements necessary to represent travel to the off-base sites is thus reduced to six.

The final reduction in array size is based on an assumption of symmetry in the travel time matrices; i.e., the time to travel from site A to site B is the same as the time to travel from site B to site A. This assumption is justified by actual travel time data collected at Charleston.

Applying these three techniques reduced the arrays from more than 15,000 to 150 storage locations, with only a slight increase in the procedural code generated and with little or no decrease in the accuracy of the schedules.

LINKED LIST TECHNIQUES

The linked list method of data storage is one of several which were tried in various versions of the AVS programs; it is demonstrably faster than the sorting method and uses considerably less core than the duplicate arrays method, both of which are discussed here.

Two sets of data arrays are used in the programs: one set contains information about the orders and the routes to which they belong; the other set contains information about the vehicles used. In both cases the information contained in the arrays is initially stored in a particular sequence. Later the same information is used in a different sequence. For example, the orders generated by AVS1 are stored in the sequence in which they are input at the remote terminal. They are then scanned repeatedly and assembled into the final vehicle schedules.

The problem is how to re-organize the data in these arrays from their initial sequence to their final sequence. The first and most natural method is actual physical re-organization of the data. The advantage of this method is that the final arrays are easy to process, either by computer or the human mind. For example, the orders processed by vehicle #1 would appear first in the final arrays, and would appear in the order in which the vehicle would service them. There are two ways in which this physical re-organization can take place: through the use of duplicate arrays or by sorting the original arrays.

In using duplicate arrays the first set of arrays is examined and the appropriate element is selected, stored in the second set of arrays, and deleted from (or marked as processed in) the original arrays. The obvious disadvantage to this method, particularly when large amounts of data are being processed, is that the memory requirements of the program are doubled.

The second method of physically re-organizing data is by sorting. The initial arrays are examined; the chosen element is selected and physically moved to the first position in the arrays; the remaining items in the arrays are shifted to make room for it. This process eliminates the need for duplicate arrays and their large memory requirements; however, sorting is a time consuming process when the arrays involved are large.

Both these methods for physical re-organization of data were used in early versions of AVS software, but the constraints of time and space made them unacceptable.

A common method of processing large amounts of data is that of embedded links; this technique is used in several of the large data base management systems now commercially available. In this technique a sequence of data items in a large set of data arrays is linked together by providing an array of pointers or links. The pointer associated with a data element gives the address of the next datum in the sequence. A pointer external to the arrays gives the address of the first element in the sequence, the link variable associated with that element gives the address of the second element in the sequence, etc. The time constraints of sorting, where n^2 movements are required to sort n elements, are not encountered. The introduction of an additional array of pointers does not usually involve a significant increase in storage, since the data elements being sorted are usually made up of corresponding components of many parallel arrays (or the addresses of indices in the pointer array are much smaller than the items which they label).

When the initial duplicate array versions of AVS were reprogrammed using sorting techniques, the size of the program decreased by 50 percent and when further reprogramming introduced the linked list techniques, the time of execution for a relatively large test case (99 orders) was decreased to about 25 per cent of its previous value.

The three methods of data restructuring are illustrated in Figures 7, 8, and 9.

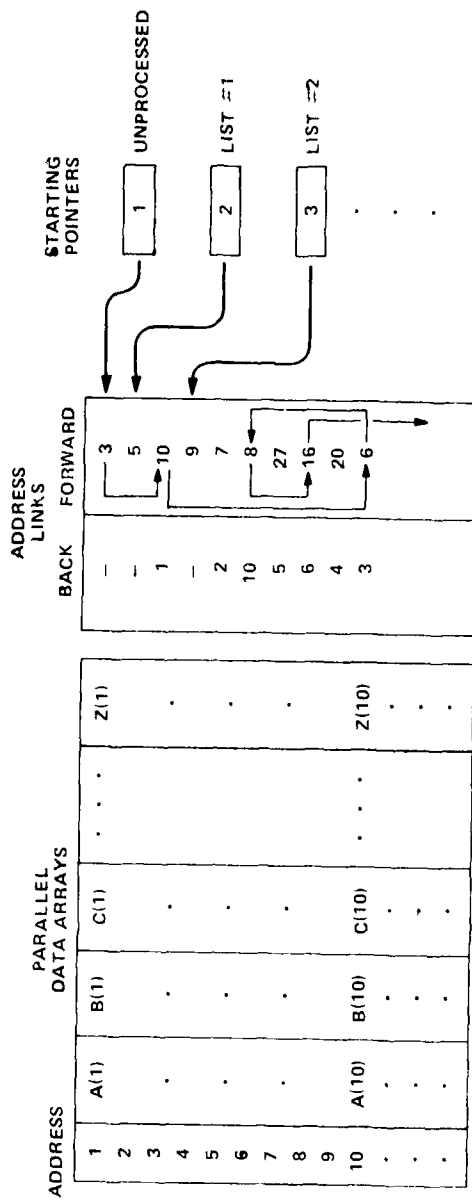


Figure 7 - Methods of Data Storage, Duplicate Arrays

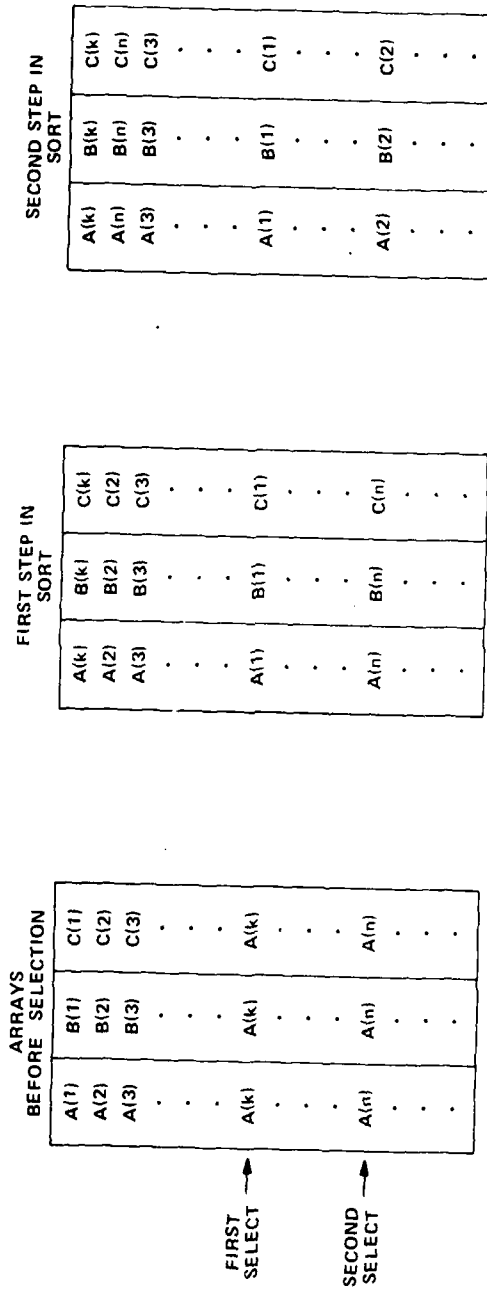


Figure 8 - Methods of Data Storage, Sequential Sort

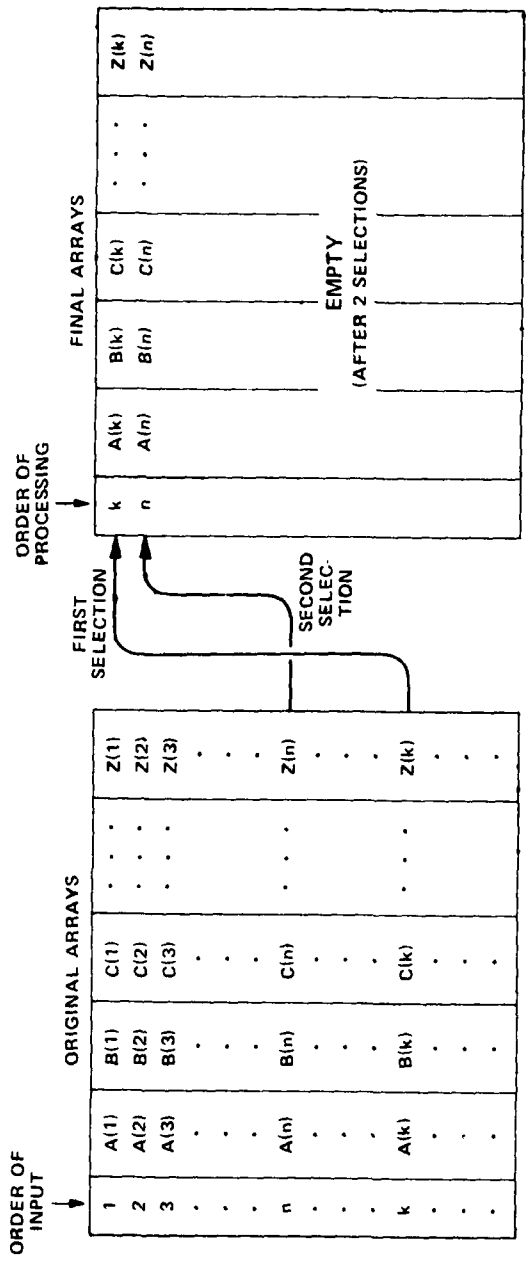


Figure 9 - Methods of Data Storage, Linked Lists

A major advantage to the linked list method of data organization is that it speeds access to the data. For example, rather than searching an entire array for an item which is in a specific vehicle route, only the items in the route need be examined. The data examination process is made more efficient in the AVS case because there is a separate linked list for unprocessed orders, i.e., those orders not yet assigned to a vehicle route. As routes are built, orders pass from the unprocessed linked list to a specific vehicle's linked list. Thus each successive search of the unprocessed orders takes less time.

The savings in space and time of the linked list system must be paid for by increased complexity of the program code.

A separate linked list must be maintained if the arrays are to be searched in reverse order, or if items are to be inserted in a list. Thus two link arrays must usually be specified to determine a linear chain of items. Examination of the coding for AVS, particularly subroutine TCARP, shows how complicated the coding can become using the linked list technique.

DATA PACKING

Because of the limited magnitude of order size and the number of warehouses considered, it was felt that all order information could be placed, "packed", in one data location rather than in three. The array INFO represents all order information, and each entry has the following format:

INFO Word Configuration

Order allocation indicator	Origin warehouse number	Destination warehouse number	Order size
+, unassigned			
-, assigned			
1 digit	2 digits	2 digits	2 digits

Data packing also reduces the number of internal sorts by listing one data element rather than three.

AVS3 - HISTORY/UPDATE PROGRAM

The AVS history file is created from the schedules produced by AVS1 and AVS2. The quantities of orders scheduled for delivery are accumulated by originating warehouse, destination warehouse, and truck name. This file is maintained by AVS3.

AVS3 is the set of subroutines used for creating, maintaining, and reporting records in the AVS history file.

File Description

The history file is sequentially organized, with records sorted by START DATE with the earliest date first.

Record Structure

The record for the history file is 1424 bytes long. Each record is uniquely identified by two of the following three fields:

START DATE

END DATE

SHIFT

The Start Date is the earliest date covered by the data in the record. Similarly, the End Date indicates the end of the period of time covered by the dates. When the two dates are unequal, the shift is set to zero and not used. For equal dates, the shift may or may not be zero. A non-zero shift indicates that the data in the record are for a single shift; a zero shift indicates a record with data accumulated over many shifts. Records for a single shift are created from the SCHED1 file produced by AVS1 or AVS2. Before the new record is inserted in the file, a copy of the mark in the SCHED1 file is made for the record, since AVS2 increments the mark by one when it updates the schedule with an emergency order. This enables AVS3 to determine whether the SCHED1 file used is a newer generation than the file which originally created the record.

Record Creation

AVS3 creates new records in the history file using the schedules generated by either AVS1 or AVS2. These schedules are contained in the SCHED1 file. AVS3 first reads the SCHED1 file. It then reads the history file for the same date and shift as the SCHED1 file. If a record

is found with the specified date and shift, the trailer mark on the history file record is compared with the mark in the SCHED1 file. If the mark on the history file record is the same as the mark on the SCHED1 file, the SCHED1 file created the history file schedule record. If the history file trailer mark is less than that in the SCHED1 file, the incoming SCHED1 file is of a more recent generation than the one that created the record. In this case, as well as when a record is found with the specified date and shift, a new record with the date, shift, and trailer mark of the SCHED1 file is created; otherwise, processing is terminated.

Assuming the record can be added to, or replaced by an existing record on the history file, AVS3 then reads through the SCHED1 schedule, accumulating the number of pallets in each scheduled order by originating warehouse, destination warehouse, truck name, and type of order (regular or special, i.e., emergency). When all scheduled orders are exhausted, AVS3 accumulates the number of pallets of each unscheduled order by originating and destination warehouse. The sum of unscheduled pallets is designated as backlog. The record is placed in its proper position in the history file, and a report of the new record is generated.

Update

When an update is made with input from the terminal, AVS3 searches the history file for the record with the specified date and shift. If a record is found which either has the specified date and shift, or spans a period of time which includes the specified date, the record is updated. If the specified record is not in the file, a message is printed and the next update record is read. If several update records are read which update the same record, the record is left in memory. The file is then searched for the new record. When the program is terminated, the file is updated and the last record updated is printed.

Record Merge

The merge function of AVS3 is used to reduce the number of records in the file by consolidating a number of records into one. Merges are performed on the file by defining the period of time from which the individual records are to be taken. AVS3 reads through the date span. It then continues to read the file, accumulating entries from the subsequent

records in the first record read. When the end of the specified date span is reached, AVS3 updates the Date fields in the original record to indicate the period of time to which the data apply. The file is then copied, with the new record inserted in its proper place. At the conclusion of the run, a report of the new records is generated. Figures 10, 11, and 12 show record entry, consolidation, and update.

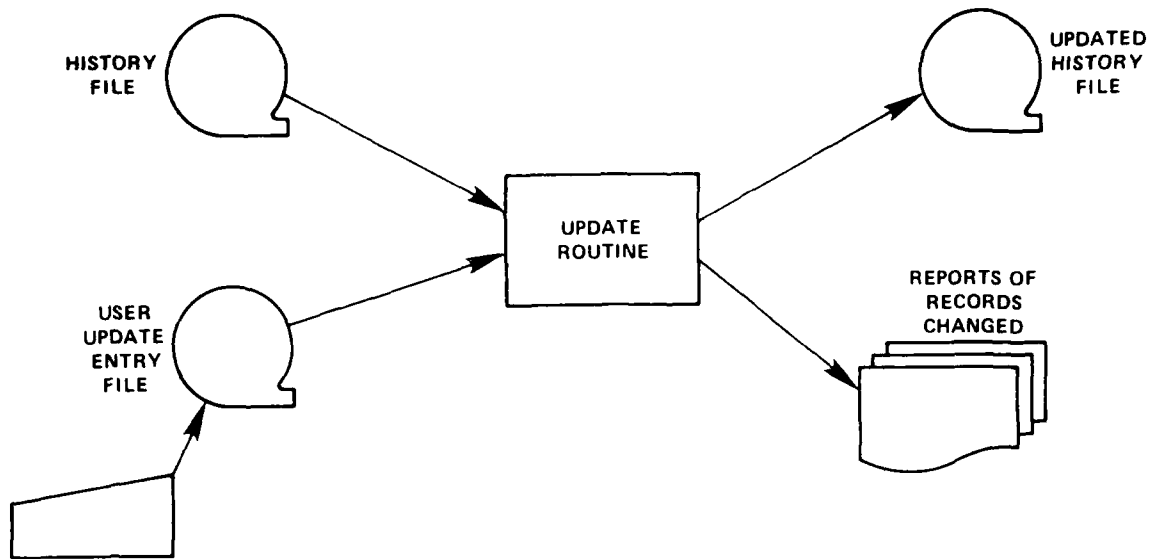


Figure 10 - Individual Entry Update

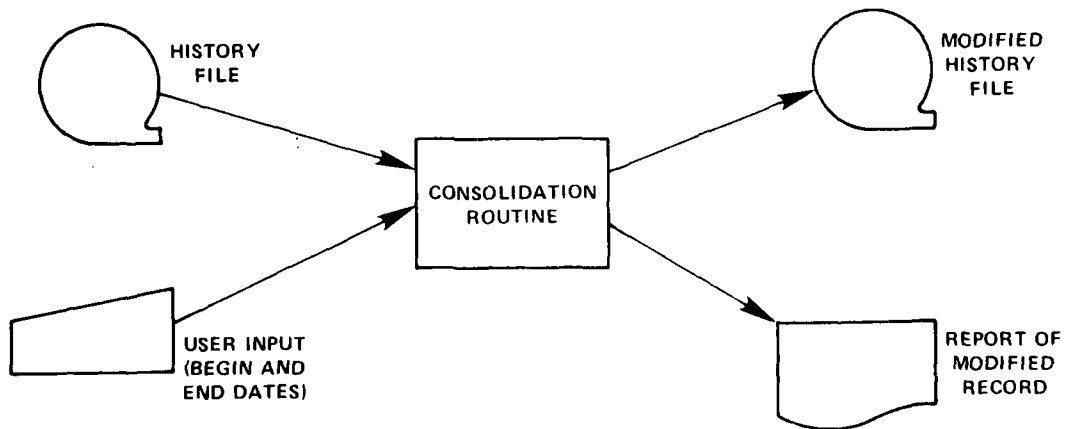


Figure 11 - Record Consolidation

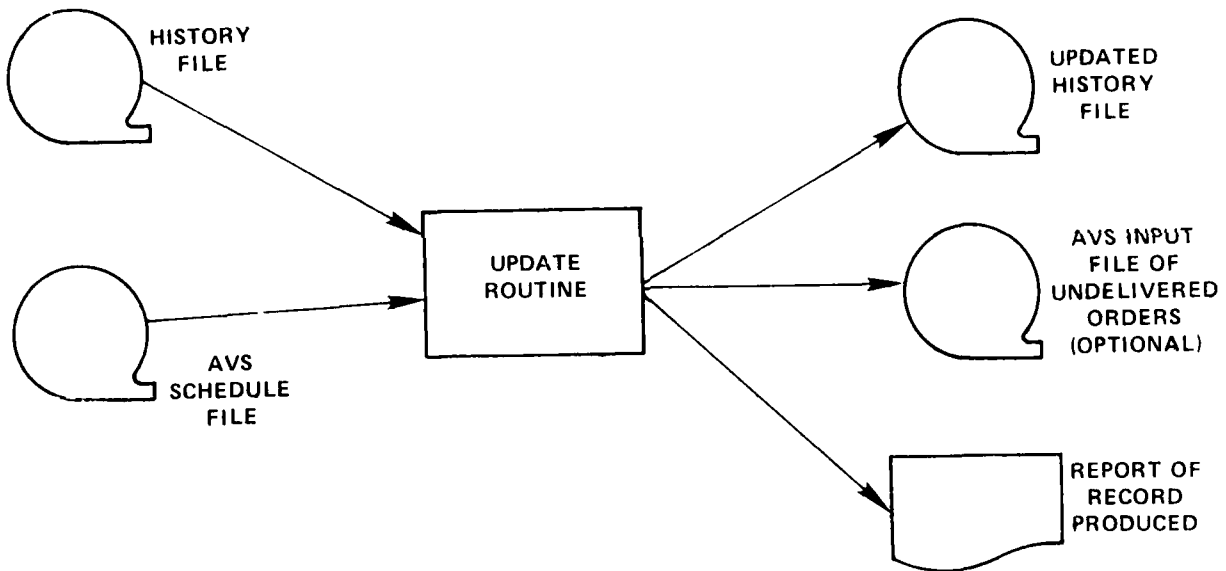


Figure 12 - Update From AVS Schedule

INPUT

AVS3 uses SINR, and data entry is accomplished by "filling in the blanks" as specified by the selected frame. AVSIN1, a SINR COBOL driver routine, consolidates the data and makes the resulting file, VS3IN, available to AVS3. When data entry is complete, AVSIN1 executes AVS3.

Route data from the previous set of schedules may be modified or updated and placed on a master file. The function number is specified as data and entered interactively: Figure 13 shows one AVS3 frame.

<u>FUNCTION</u>	<u>ACTION</u>
1	Update the history file using the schedule produced by AVS1 or AVS2
2	Update or add specific entries to the history file
3	Merge two or more records to create an aggregate of the records in the history file, replacing the old records with the new one
4	Produce a report of either an individual record or an aggregate of several records

PROGRAM OUTPUT

AVS2 produces a Summary Report of the history file for the date and shift specified. This report gives cargo transfer statistics by warehouse, indicating the number of pallets delivered, shipped, and backlogged.

For most functions of AVS3, the program will produce two outputs. One will be the history file with the new or updated record. The other output is a report of the record as created or updated. The report includes:

- the date of the updated or created record
- the shift, if applicable
- the warehouse from which the pallets were sent
- the receiving warehouse

FRAME A07

```

(FRM #A07
DOCID | 1 |
FRAME | 2 | (ENTER 7) (FUNCTION | 3 | (1 - UPDATE FROM AVS SCHEDULE) (FOR FUNCTIONS 3&4
(FOR FUNCTION 1) (2 - UPDATE FROM KEYBOARD) MMDDYY
(DO YOU WANT BACKLOG INCLUDED) (3 - MERGE RECORDS) (START DATE | 13 |
(IN NEXT AVS RUN - Y OR N) | 4 | (4 - REPORT) (END DATE | 14 |
MMDDYY (SHIFT | 15 |
(FOR FUNCTION 2) (ENTER DATE FOR ALL UPDATES) | 5 | (SHIFT) | 6 |
(OPTION) (DESTINATION) (SIZE) (VEHICLE) (EMERGENCY?) (RPLCE ENTRY?)
(1) | 7 | | 8 | | 9 | | 10 | | 11 | | 12 |
(2) | | | | | | | | | | | |
(3) | | | | | | | | | | | |
(4) | | | | | | | | | | | |
(5) | | | | | | | | | | | |
(6) | | | | | | | | | | | |
(ERROR) | | (FIELD) (DESCRIPTION) (ERROR) | (DESCRIPTION)
(ERROR) | | | (ERROR) | |
(ERROR) | | | (ERROR) | |
(ERROR) | | | (ERROR) | |

```

Figure 13 - AVS3 Frame Description

- the name of the truck which delivered the pallets
- the number of pallets delivered as regular orders
- the number of pallets delivered as emergency orders

Report of records that have data from only one shift will also include the number of pallets that were not scheduled as deliveries by AVS1 or AVS2. These statistics are arranged in the same way as the others and are organized by sending and receiving warehouses.

Appendix D gives the AVS3 program listing and flowcharts.

Appendix E provides illustrative examples of AVS1, AVS2, and history file entries and output. The printouts are designed to be used directly as dispatch schedules.

ACKNOWLEDGMENT

The authors wish to acknowledge the cooperation and B3500 System expertise of Robert E. Lee and Robert Owens, Code 61 - NSC, Charleston, S.C. Without their assistance AVS could not have been modified for the B3500 computer system.

APPENDIX A - USER'S MANUAL

INFORMATION RETRIEVAL SYSTEM (SINR)

For ease of user data entry, AVS employs the SINR Routines provided by the Fleet Material Support Office as a uniform automatic data processing system for Naval Supply Centers. User instructions and data formatting are displayed on a CRT screen. This display is referred to by SINR as a "frame". AVS frames are given as follows:

FRAME #	DESCRIPTION
A00	General description of all frames available to AVS
A01	Order description input
A02	Vehicle description input
A03	Input options for regular and emergency orders
A04	Run execution
A05	Input options for emergency orders
A06	Clear and restart, given as an option of A04
A07	History file update options

Enter 6 digit password and transmit by pressing XMIT key. When "PASSWORD" has been cleared from the screen, enter the following commands:

ENTER

*DCH MODE FRAME (transmit)

System Reply

FRAME MODE ENABLED

ENTER

FRM #Name, #A01, #A02, #A03, #A04, #A05, or #A07, of frame desired
(transmit)

* System commands are specified by upper case letters

SYSTEM REPLY

The frame specified will appear on the screen

ENTER

Key in data between displayed []. The skip tab key may be used to position data. When the frame is complete, (transmit).

SYSTEM REPLY

The system will check the correctness of the data entered and will display error or acceptance messages. If errors appear in the data, position the cursor at the beginning of the frame and re-enter the frame. To clear the screen before the next frame request, key "HOME" and "SHIFT" at the same time. When the screen is clear, the user may request the next frame. When all necessary frames have been completed and AVS has been executed, the user may exit the system by:

ENTER

DCH BYE (transmit)

SYSTEM REPLY

PASSWORD

REGULAR ORDER SCHEDULING INSTRUCTIONS

To perform a regular order scheduling run, complete frames A01, A02, A03, and A04.

FRAME A01

Figure 14 shows a typical Frame A01 as it appears on the CRT screen. Enter the following data on A01:

<u>FIELD</u>	<u>DESCRIPTION</u>
1	"AVS" System Doc ID
2	"1" for this data frame

<u>FIELD</u>	<u>DESCRIPTION</u>
3	Order size or number of pallets to be moved from the origin site to the destination site. If the number of pallets is less than 10, enter leading zero
4	Origin site name, up to 6 characters, for a site as given in Table 1
5	Destination site, up to 6 characters, for a site as given in Table 1

Twenty orders may be entered on each A01 frame. If more than 20 orders are needed, fill and transmit A01 as many times as required. The maximum number of orders per run is 99. If the number of orders needed is less than 20, skip forward and position cursor at "LAST ORDER PROCESSED" space and transmit. Wait for the system's reply. If errors appear, re-enter frame and transmit.

FRAME A02

Figure 15 shows Frame A02 as it appears on the CRT screen. Enter the following data on A02:

<u>FIELD</u>	<u>DESCRIPTION</u>
1	"AVS" System Doc ID
2	"2" for this frame
3	"X" if this vehicle is to be used in AVS "*" if specified industrial tractor is to be an IT. Otherwise, the vehicle will be considered a tractor trailer, TT.
4	Capacity, maximum number of pallets vehicle can carry. Skip field if default value is desired. Defaults: ST = 7, TR = 12, TT = 14, IT = 10
5	Maximum route time in minutes for this vehicle. Skip if default is desired Default = 480 min

Follow instruction given on frame to transmit data.

FRAME A03

Figure 16 shows Frame A03 as it appears on the CRT screen. Enter the following data on A03:

<u>FIELD</u>	<u>DESCRIPTION</u>
1	"AVS" System Doc ID
2	"3" for the frame
3	Date of schedule
4	Start time of shift using 24 hr clock
5	Maximum length of shift Default = 480 min

Position cursor at "CURSOR" and transmit. Wait until system replies before proceeding.

FRAME A04

Figure 17 shows Frame A04 as it appears on the CRT screen. Enter the following data on A04:

1	"AVS" for system Doc ID
2	"4" to produce schedules for data "6" to clear registers and re-enter data for frames A01, A02, A03, A05

Move cursor to "CURSOR" and transmit. No reply will be made by system.

EMERGENCY ORDERS SCHEDULING INSTRUCTIONS

To perform an emergency order scheduling run, complete frames A01, A02, A03, A04, and A05. The emergency order program (AVS2) uses the same information as the regular orders. Delivery preference options are given in Frame A05, Figure 18. The items for frames A01, A02, and A03 which apply only to the emergency orders are described as follows:

FRAME A01

AUTOMATED VEHICLE SCHEDULING -AVS- ORDER INPUT

```

(FRM :A01
DOCID ( 1 )
FRAME ( 2 )
) (ENTER ORDERS -
( ORDERS 1) ( 3 ) ( 4 ) ( 5 ) (
( ORDERS 3) ( ) ( ) ( ) ( )
( ORDERS 5) ( ) ( ) ( ) ( )
( ORDERS 7) ( ) ( ) ( ) ( )
( ORDERS 9) ( ) ( ) ( ) ( )
( ORDERS 11) ( ) ( ) ( ) ( )
( ORDERS 13) ( ) ( ) ( ) ( )
( ORDERS 15) ( ) ( ) ( ) ( )
( ORDERS 17) ( ) ( ) ( ) ( )
( ORDERS 19) ( ) ( ) ( ) ( )
(LAST ORDER PROCESSED) ( )
) ( ERRORS ORDER EXPLANATION ORDER EXPLANATION
( ERRORS ( ) ( ) ( ) ( )
( ERRORS ( ) ( ) ( ) ( )
( ERRORS ( ) ( ) ( ) ( )
( ERRORS ( ) ( ) ( ) ( )
( ERRORS ( ) ( ) ( ) ( )
( ERRORS ( ) ( ) ( ) ( )
(CURSORS)

```

Figure 14 - Frame A01

FRAME A02

AUTOMATED VEHICLE SCHEDULING -AVS- VEHICLE INPUT

```

(FRM #A02
DOCID | 1 |
FRAME | 2 | (ENTER 2)
(SELECT VEHICLES, SPECIFY CAPACITIES AND TIMES BELOW
( TYPE USE CAPAC TIME USE CAPAC TIME USE CAPAC TIME
(STRADDLES 1|3| |4| |5| |2| | | |3| | | |
(4| | | | |5| | | | |6| | | |
(7| | | | |8| | | | |9| | | |
(10| | | | |11| | | | |12| | | |
(TRANSPRTR 1| | | | |2| | | | |3| | | |
(4| | | | |5| | | | |6| | | |
(IND TRCTRS 1| | | | |2| | | | |3| | | |
(4| | | | |5| | | | |6| | | |
(7| | | | |8| | | | |9| | | |
(10| | | | |11| | | | |12| | | |
(POSITION CURSOR HERE| | (BEFORE XMITTING DATA
(WAIT FOR OUTPUT BELOW BEFORE PROCEEDING
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
(LAST VEHICLE PROCESSED| | | | |

```

Figure 15 - Frame A02

FRAME A03

(FRM #A03 AUTOMATED VEHICLE SCHEDULING -AVS-
 (SCHEDULE INFORMATION)

DOCID [1]
 FRAME [2] (ENTER 3)

(ENTER DATE) [3] (SCHEDULING START TIME) [4]
 (24 HR CLOCK)
 (MMDDYY)

(MAXIMUM ROUTE DURATION) [5] (MINUTES) (CURSOR)[]

(ERRORS) [] [NO ERRORS]
 (ERRORS) [] []
 (ERRORS) [] []
 (ERRORS) [] []
 (ERRORS) [] []
 (ERRORS) [] []

Figure 16 - Frame A03

FRAME A04

(FRM #A04 AUTOMATED VEHICLE SCHEDULING -AVS-
 (START / CLEAR AND RESTART)

DOCID [1]

(TO GENERATE SCHEDULES ENTER 4]

(TO CLEAR ALL DATA AND RESTART ENTER 6))

[2]

(CURSOR)[]

Figure 17 - Frame A04

FRAME A01

<u>FIELD</u>	<u>DESCRIPTION</u>
1	"AVS" System Doc ID
2	"1" for this frame
3	Order size, number of pallets
4	Origin site name
5	Destination site name

A maximum of 99 emergency orders may be entered per emergency order run.

FRAME A02

<u>FIELD</u>	<u>DESCRIPTION</u>
1	"AVS" System Doc ID
2	"2" for this frame
3	"S" to use the same vehicles as the previous regular order run. Otherwise, enter "X" or "*" for each vehicle to be used and complete Field 5 for each vehicle to be modified.
5	Time in minutes of maximum route Default = 480 min

FRAME A03

<u>FIELD</u>	<u>DESCRIPTION</u>
1	"AVS" System Doc ID
2	"3" for this frame
3	Date
4	Time of emergency orders using 24-hr clock
5	Maximum length of shift (same as regular orders) Default = 480 min

FRAME A04

Frame A04 is used as before to execute the emergency orders program.

FRAME A05

Figure 18 shows Frame A05 as it appears on CRT screen. Data to be entered on A05 are as follows:

<u>FIELD</u>	<u>DESCRIPTION</u>
1	"AVS" System Doc ID
2	"5" for this frame
3/4	Answer questions on frame

Position cursor at "CURSOR" and transmit. Do not clear screen until MIX and PROPT values are returned.

FRAME A05

EMERGENCY VEHICLE SCHEDULING -EVS- OPTION INPUT

DOCID([1]) ENTER AVS

FRAME [2] ENTER 5 FOR THIS FRAME

MAY THE USE OF MORE THAN ONE VEHICLE TYPE

BE ALLOWED TO FILL THIS ORDER [3] ENTER Y FOR YES - N FOR NO

MAY NON-EMERGENCY ORDERS BE BUMPED TO FILL

THIS ORDER [4] ENTER Y FOR YES - N FOR NO

CURSOR []

RESULTS

MIX []

PROPT []

Figure 18 - Frame A05

History File Instructions

To perform updates and report generation of the AVS history file, frame A07 must be completed (See Figure 19). The following four functions may be performed using this frame:

<u>FUNCTION</u>	<u>ACTION</u>
1	Update the history file using the schedule produced by AVS1 or AVS2
2	Update or add specific entries to the history file
3	Merge two or more records to create an aggregate of the records in the history file, replacing the old records with the new one.
4	Produce a report of either an individual record or an aggregate of several records

Frame A07 as it is filled out for each of the functions is described as follows:

<u>FIELD</u>	<u>DESCRIPTION</u>
1	"AVS" System Doc ID
2	"7" for this frame
3	Desired function "1", "2", "3", or "4"

Function 1 Instructions

<u>FIELD</u>	<u>DESCRIPTION</u>
4	"y" or "N". If "Y", a file containing any orders that were not scheduled by AVS1 or AVS2 will be included in the next AVS run (UNIMPLEMENTED)

Function 2 Instructions

<u>FIELD</u>	<u>DESCRIPTION</u>
5	Date (MMDDYY) of the AVS History File record to be updated
6	Time (24-hour clock) of AVS history file record to be updated
7	Warehouse from which shipment was sent
8	Warehouse which received shipment
9	Number of pallets
10	Name of vehicle on which the shipment was made
11	"Y" emergency order, "N" otherwise
12	"Y" replace any data in the History File, "N" corresponding entry in the history file is to be incremented by the number in Field 9

Function 3 and 4 Instructions

<u>FIELD</u>	<u>DESCRIPTION</u>
13	Start Date (MMDDYY) for the range of history file records to be included in the report or in the merge record
14	End Date (MMDDYY) to indicate the date of the last record to be included in the report or in the merge record. A value equal to the start date (Field 13) indicates all records for that date are to be included.
15	Shift of record to be in the report. This field is used only for function 4 (report generator) and only when fields 13 and 14 are the same. It is ignored in all other instances.

FRAME A07

```

(FRM #A07
DOCID [ 1 ]
FRAME [ 2 ] (ENTER 7) (FUNCTION [ 3 ] (1- UPDATE FROM AVS SCHEDULE) (FOR FUNCTIONS 3&4
(FOR FUNCTION 1) (2- UPDATE FROM KEYBOARD) MDDYY
(DO YOU WANT BACKLOG INCLUDED) (3- MERGE RECORDS) (START DATE [ 13 ]
(IN NEXT AVS RUN - Y OR N) [ 4 ] (4- REPORT) (END DATE [ 14 ]
MMDDYY (SHIFT [ 15 ]
(FOR FUNCTION 2) (ENTER DATE FOR ALL UPDATES) [ 5 ] (SHIFT) [ 6 ]
(OPTION) (DESTINATION) (SIZE) (VEHICLE) (EMERGENCY?) (RPLCE ENTRY?)
(1) [ 7 ] [ 8 ] [ 9 ] [ 10 ] [ 11 ] [ 12 ]
(2) [ ] [ ] [ ] [ ] [ ] [ ]
(3) [ ] [ ] [ ] [ ] [ ] [ ]
(4) [ ] [ ] [ ] [ ] [ ] [ ] (BEFORE X-MIT)
(5) [ ] [ ] [ ] [ ] [ ] [ ] (POSITION)
(6) [ ] [ ] [ ] [ ] [ ] [ ] (CURSOR HERE [ ]
(ERROR) [ ] (DESCRIPTION) (ERROR) (DESCRIPTION)
(ERROR) [ ] (ERROR) [ ]
(ERROR) [ ] (ERROR) [ ]
(ERROR) [ ] (ERROR) [ ]

```

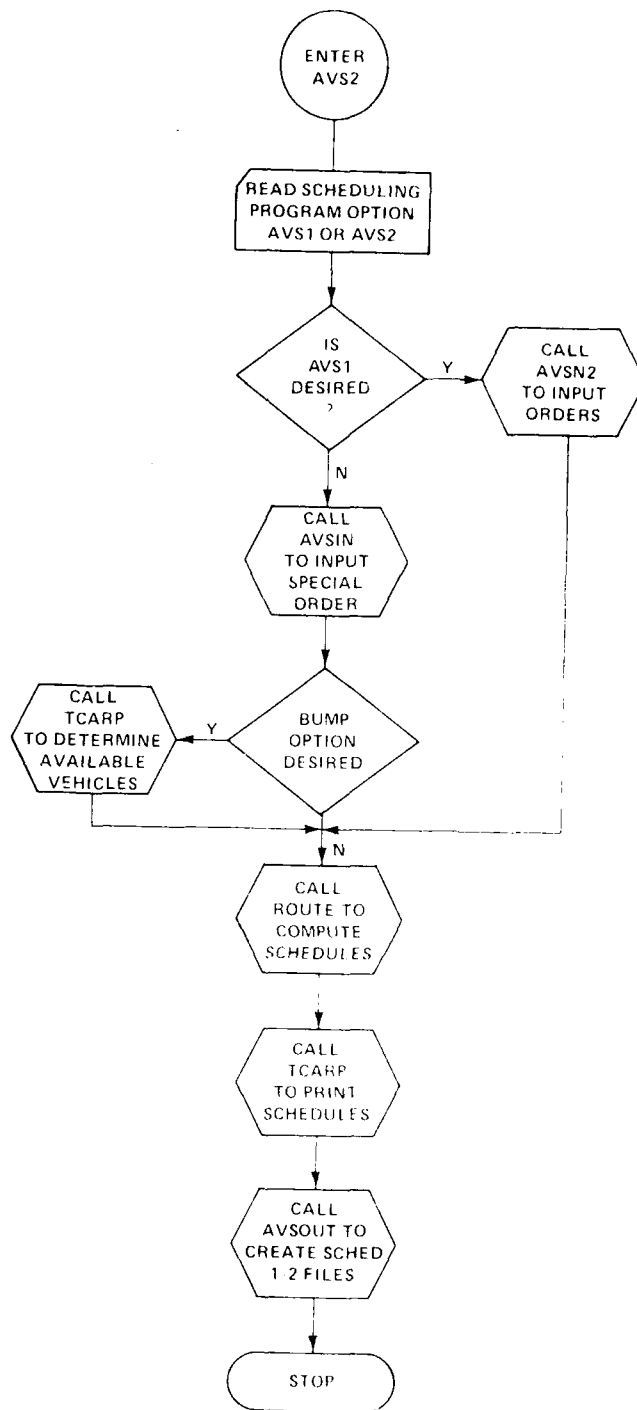
Figure 19 - Frame A07

APPENDIX B - PROGRAMMER'S GUIDE - SUBROUTINE FLOWCHARTS/LISTINGS

ROUTINE: AVS2

Description:

AVS2 is an executive routine which executes each segment of the scheduling algorithm.




```

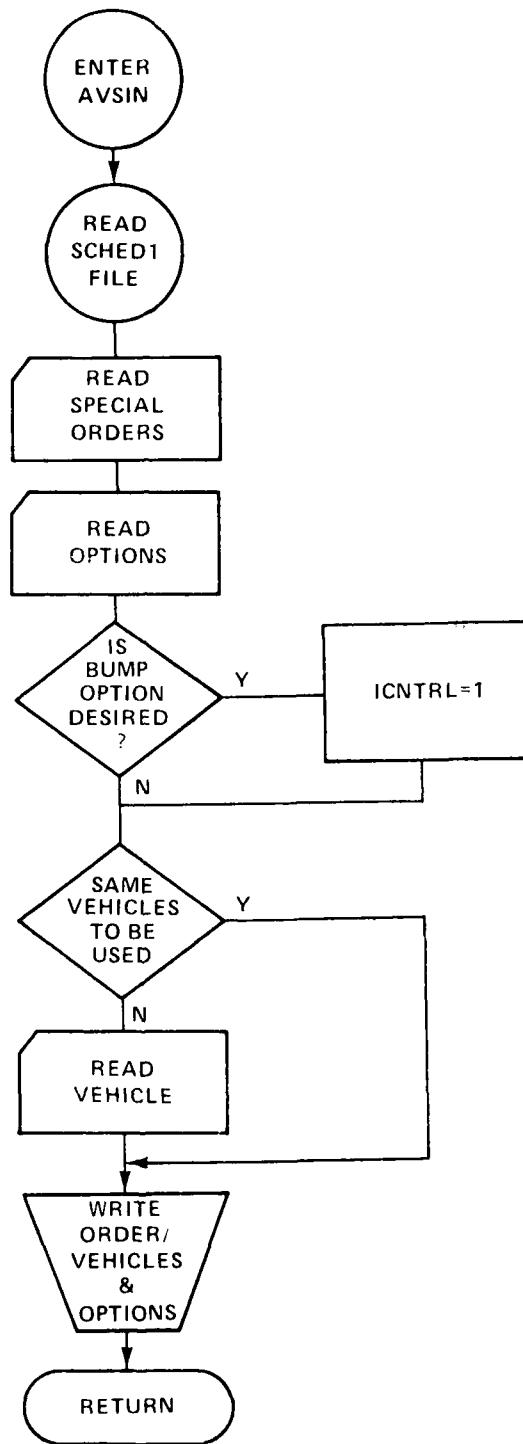
1      IDENT AVS2                                AVS2  2
      SIZE INTEGER=7                            AVS2  3
      SEGMENT AVSIN,AVSN2,AVSOUT,ROUTE,ALDTR,BUILDOS,SRDST,TCARP,TCONV,NOM AVS2  4
5      FILE 1=SCHEM1,UNIT=DISK,BLOCKING=1,RECORD=80,LOCK AVS2  5
      FILE 2=SCHEM2,UNIT=DISK,BLOCKING=1,RECORD=80,LOCK AVS2  6
      FILE 3=VS2IN,UNIT=DISK,BLOCKING=1,RECORD=80,LOCK AVS2  7
      C.....AVS2  8
      C AVS2  9
      C          (XFORN COMPILER) AVS2 10
10     C AVS2 11
      C AVS2 12
      C AVS2 13
      C AVS2 14
      C AVS2 15
15     C AVS2 IS A PROGRAM FOR ROUTING UP TO 50 VEHICLES OF AVS2 16
      C THREE DIFFERENT CLASSES TO AS MANY AS 100 WAREHOUSES AVS2 17
      C IN A NAVY WAREHOUSE COMPLEX. THIS PARTICULAR DECK IS AVS2 18
      C FOR THE NAVAL SUPPLY CENTER IN CHARLESTON, S.C. AVS2 19
      C EXECUTIVE ROUTINE CALLS EACH MODULE INTO CORE AVS2 20
20     C AVSIN - READS ALL INPUT AND CREATES INITIAL ORDER LISTS AVS2 21
      C ROUTE - CREATES SCHEDULE FROM ORDERS AND AVAILABLE VEHICLES AVS2 22
      C TCARP - PRINTS SCHEDULES AVS2 23
      C AVSOUT - CREATES UPDATE TAPE 2 AVS2 24
      C.....AVS2 25
25     C AVS2 26
      C INTEGER PTSOR,CAPAC,TRUCK,DATE,PTTRK,ONUMB AVS2 27
      C * ,TRKSAV AVS2 28
      C ALPHA WHNAM AVS2 29
      C REAL LTIME AVS2 30
      C COMMON AVS2 31
30     C */GEN/ RTTIM,MINLD(4) AVS2 32
      C */WHINTG/ NWARE,NAREA(1E) AVS2 33
      C */SPORD/ ICNTRL,SAVTIM,ISORD AVS2 34
      C */SAVP/ NSAV(4),TRKSAV(4) AVS2 35
35     C */SCHEM/ ONUM(200),INFO(200),IOESTP(200),IAREA(200),LFRWD(200),
      C *LBKWD(200),LCFWD(200),LCHWD(200),ATIME(200),STOPT(200) AVS2 36
      C */TRUCKS/ PTSOR(50),TRUCK(50),CAPAC(50),STIME(50),LTIME(50), AVS2 38
      C * RTLIM(50),TLEFF(50),NT AVS2 39
      C */WHOUSE/ WHNAM(52) AVS2 40
40     C */TIMTAB/ TTINE(3,45),TTIM2(6) AVS2 41
      C */NSCLNS/ NORDR,NTRKS(4),PTTRK(4) AVS2 42
      C */INOPT/ SHIFT,CATE AVS2 43
      C */BLDS/ LASTW,LASTA,NEXTA,ITRUCK,ITYPE,IPASS,NPALS,TIME AVS2 44
      C */LOADSV/ LOAD AVS2 45
45     C ISGRD=-100 AVS2 46
      C DO 1000 I=1,4 AVS2 47
      C TRKSAV(I)=-1 AVS2 48
1000  C NSAV(I)=0 AVS2 49
      C ICNTRL=1 AVS2 50
50     C READ(5,500) IAVS AVS2 51
      C IF(IAVS.EQ.1) CALL AVSP AVS2 52
      C IF(IAVS.EQ.2) CALL AVSIN AVS2 53
      C IF(ICNTRL.EQ.0) CALL TCARP AVS2 54
      C CALL ROUTE(IAVS) AVS2 55
55     C ICNTRL=1 AVS2 56
      C CALL TCARP AVS2 57
      C CALL AVSOUT(IAVS) AVS2 58
500  C FORMAT(4X,I1) AVS2 59
      C STOP AVS2 60
60     C END AVS2 61

```

ROUTINE: AVSIN

Description:

AVSIN inputs the special orders to be considered, the vehicles to handle them, and the schedule run options.



```

1          SUBROUTINE AVSIN                                AVSIN  2
C          .....                                        AVSIN  3
C          .....                                        AVSIN  4
C          AVSIN INPUTS ALL PARAMETERS NECESSARY FOR THE SCHEDULE AVSIN  5
C          COMPUTATIONS. IT ALSO SORTS ORDERS AND VEHICLES TO AVSIN  6
C          TO REDUCE EXECUTION TIME.                      AVSIN  7
C          .....                                        AVSIN  8
C          .....                                        AVSIN  9
C          .....                                        AVSIN 10
10         INTEGER OSTART(16),OEND(16)                   AVSIN 11
           INTEGER PTSOR,CAFAC,TRUCK,ONUMB               AVSIN 12
           INTEGER PTRK,DATE                             AVSIN 13
           ALPHA BUMP(2),WHNAM,TYPE(4),AST,IORIG,ITERP,ILPHA AVSIN 14
           INTEGER PROPT,SORCR,TRKSAV                   AVSIN 15
15         REAL LTIME                                    AVSIN 16
           COMMON                                        AVSIN 17
           * /GEN/ RTTIM,MINLD(4)                       AVSIN 18
           * /SAVEP/ NSAV(4),TRKSAV(6)                  AVSIN 19
           * /WHINTG/ WHARE,NAREA(16)                   AVSIN 20
           * /SCHEDL/ ONUMB(200),INFO(200),IDESTP(200),IAREA(200),LFRWD(200),
           * /LWKWD(200),LCFWD(200),LCBWD(200),ATIME(200),STOPT(200) AVSIN 21
           * /SPORD/ ICNTRL,SAVTIM,SORDR                AVSIN 22
           * /TRUCKS/ PTSOR(50),TRUCK(50),CAPAC(50),STIME(50),LTIME(50),
           * /RTLIM(50),TLEFT(50),NT                    AVSIN 23
           * /WHOUSE/ WHNAM(92)                         AVSIN 24
           * /TIMTAB/ TTIME(3,45),TTIM2(16)             AVSIN 25
           * /MSCLNS/ NORDR,NTRKS(4),PTRK(4)            AVSIN 26
           * /INOPT/ SHIFT,DATE                          AVSIN 27
           DATA TYPE/2HST,2HTR,2HTT,?HIT/              AVSIN 28
           DATA BUMP/2HMO,3HYES/                       AVSIN 29
           REMIND 1                                      AVSIN 30
           MSTART=0                                     AVSIN 31
           READ(1,102) WHARE,(NAREA(I),I=1,16),(WHNAM(I),I=1,WHARE) AVSIN 32
           READ(1,100) (CNUMB(I),INFO(I),IDESTP(I),IAREA(I), AVSIN 33
           * /LFRWD(I),LWKWD(I),LCFWD(I),LCBWD(I),ATIME(I),STOPT(I), AVSIN 34
           * /I=1,200)                                  AVSIN 35
           READ(1,101) (PTSOR(I),TRUCK(I),CAPAC(I),STIME(I),LTIME(I), AVSIN 36
           * /RTLIM(I),TLEFT(I),I=1,50)                 AVSIN 37
           READ(1,104) ((TTIME(I,J),I=1,3),J=1,45),TTIM2 AVSIN 38
           READ(1,105) PTRK,NTRKS,NORDR,SORDR           AVSIN 39
           READ(1,106) SHIFT,DATE                       AVSIN 40
           READ(1,107) MARK                             AVSIN 41
           REMIND 1                                      AVSIN 42
           DO 9000 I=1,NCRDR                             AVSIN 43
           INFO(I)=IABS(INFO(I))                        AVSIN 44
           IF (IDESTP(I).EQ.0) GO TO 9000                AVSIN 45
           INFO(I)=-INFO(I)                             AVSIN 46
           9000 CONTINUE                                 AVSIN 47
           WRITE(6,600)                                  AVSIN 48
           IMO = OATE / 10000                            AVSIN 49
           IDAY = (OATE - 10000*IMO) / 100              AVSIN 50
           IYR = OATE - 10000*IMO - 100*IDAY            AVSIN 51
           WRITE(6,601) IPO, IDAY, IYR, SHIFT           AVSIN 52
           WRITE(6,602)                                  AVSIN 53
           2000 READ(5,500) STIM,ISIZE,IORIG,ITERM       AVSIN 54
           IF (STIM.LE.0.0) GO TO 2500                  AVSIN 55
           SAVTIM = STIM                                AVSIN 56
           .....                                        AVSIN 57
           .....                                        AVSIN 58

```

```

        NORDR=NORDR+1
        IF(MSTART.EQ.0) MSTART=NORDR
60      SORDR=SORDR-1
        I=MOD(IABS(SORDR),100)
        WRITE(6,603) I,ISIZE,IORIG,ITERM
        IF((I-(I/5)*5).EQ.0) WRITE(6,606)
        INFO(NORDR)=ISIZE+MATCH(IORIG,IA1)*10000+MATCH(ITERM,IA2)*100
65      IAREA(NORDR)=IA1*100+IA2
        ONUMR(NORDR)=SCRDR
        GO TO 2000
C      SORT ORDERS BY AREA
2500   LIMIT=NORDR-1
70     DO 2600 I=MSTART,LIMIT
        ISTART=I+1
        IF(ISTART.GT.NCRDR) GO TO 2600
        DO 2700 J=ISTART,NORDR
        IF(IAREA(I).LE.IAREA(J)) GO TO 2700
75     ISAVEA=IAREA(J)
        ISAVEI=INFO(J)
        ISAVEO=ONUMR(J)
        ONUMR(J)=ONUMR(I)
        IAREA(J)=IAREA(I)
68     INFO(J)=INFO(I)
        ONUMR(I)=ISAVEC
        IAREA(I)=ISAVEA
        INFO(I)=ISAVEI
2700   CONTINUE
85     2600 CONTINUE
C
C      MARK START OF COMMCNS. ORIGIN. DESTINATIONS
C
90     DO 2400 I=1,16
        OEND(I)=0
2400   OSTART(I)=0
        DO 2300 I=MSTART,NORDR
        JAREA=IAREA(I)/100
        IF(OSTART(JAREA).LE.0) OSTART(JAREA)=I
        IF(I.EQ.NORDR) OEND(JAREA)=I
        IF(I+1.GT.NORDR) GO TO 2300
        IF(JAREA.NE.IAREA(I+1)/100) OEND(JAREA)=I
2300   CONTINUE
100    C      SORT BY ORIGIN AND DESTINATION IN SAME AREA
        DO 2200 I=1,16
        IF(OSTART(I).LE.0) GO TO 2200
        ISTART=OSTART(I)
        IEND=OEND(I)
105    IF(ISTART.EQ.IEND) GO TO 2200
        JEND=IEND-1
        DO 2250 J=ISTART,JEND
        JSTART=J+1
        DO 2275 K=JSTART,IEND
        IF(INFO(J).LE.INFO(K)) GO TO 2275
110    ISAVEA=IAREA(J)
        ISAVEI=INFO(J)
        ISAVEO=ONUMR(J)
        IAREA(J)=IAREA(K)
        AVSIN 59
        AVSIN 60
        AVSIN 61
        AVSIN 62
        AVSIN 63
        AVSIN 64
        AVSIN 65
        AVSIN 66
        AVSIN 67
        AVSIN 68
        AVSIN 69
        AVSIN 70
        AVSIN 71
        AVSIN 72
        AVSIN 73
        AVSIN 74
        AVSIN 75
        AVSIN 76
        AVSIN 77
        AVSIN 78
        AVSIN 79
        AVSIN 80
        AVSIN 81
        AVSIN 82
        AVSIN 83
        AVSIN 84
        AVSIN 85
        AVSIN 86
        AVSIN 87
        AVSIN 88
        AVSIN 89
        AVSIN 90
        AVSIN 91
        AVSIN 92
        AVSIN 93
        AVSIN 94
        AVSIN 95
        AVSIN 96
        AVSIN 97
        AVSIN 98
        AVSIN 99
        AVSIN100
        AVSIN101
        AVSIN102
        AVSIN103
        AVSIN104
        AVSIN105
        AVSIN106
        AVSIN107
        AVSIN108
        AVSIN109
        AVSIN110
        AVSIN111
        AVSIN112
        AVSIN113
        AVSIN114
        AVSIN115

```

115	INFO(J)=INFO(K)	AVSIN116
	ONUMB(J)=ONUMB(K)	AVSIN117
	IAREA(K)=ISAVEA	AVSIN118
	INFO(K)=ISAVEI	AVSIN119
	ONUMB(K)=ISAVEC	AVSIN120
120	2275 CONTINUE	AVSIN121
	2250 CONTINUE	AVSIN122
	2200 CONTINUE	AVSIN123
	WRITE(6,608) SAVTIM	AVSIN124
	READ(5,501) MIX,PROPT	AVSIN125
125	WRITE(6,607) BUMP(PROPT+1)	AVSIN126
	ICNTRL=0	AVSIN127
	IF(PhOPT.LE.0) ICNTRL=2	AVSIN128
	NTSAV=1	AVSIN129
	NT=NTRKS(1)+NTRKS(2)+NTRKS(3)	AVSIN130
130	C	AVSIN131
	C TRUCK INPUT	AVSIN132
	C	AVSIN133
	READ(5,502) ILFHA	AVSIN134
	IF(ILFHA.EQ.1HA) GO TO 4000	AVSIN135
135	DO 2225 J=1,4	AVSIN136
	NSAV(J)=NTRKS(J)	AVSIN137
	TRKSAV(J)=PTTRK(J)	AVSIN138
	PTTRK(J)=PTTRK(J)+NTRKS(J)	AVSIN139
	2225 NTRKS(J)=0	AVSIN140
140	ISS=SHIFT	AVSIN141
	XSHFT=(ISS/100)*60+MOD(ISS,100)	AVSIN142
	ISS=SAVTIM	AVSIN143
	SORTIM=(ISS/100)*60+MOD(ISS,100)	AVSIN144
	SORTIM=SORTIM-XSHFT	AVSIN145
145	SORTIM=RTTIM-SORTIM	AVSIN146
	RTMAX=RTTIM	AVSIN147
	IF(SORTIM.LE.RTMAX) RTMAX=SORTIM	AVSIN148
	IF(IRTMX.NE.0) RTMAX=FLOAT(IRTMX)	AVSIN149
	I=NT	AVSIN150
150	NTSAV=NT+1	AVSIN151
	ITCNT=0	AVSIN152
	3000 READ(5,505) AST,INTRK,INTIM,INCAP	AVSIN153
	IF(INTRK.LE.0.OR.INTRK.GT.50) GO TO 3100	AVSIN154
	IF(AST.NE.1H*) GO TO 3005	AVSIN155
155	ITCNT=ITCNT+1	AVSIN156
	INTRK=PTTRK(4)+ITCNT	AVSIN157
	IATYPE=4	AVSIN158
	3005 IF(INTRK.LE.40) IATYPE=3	AVSIN159
	IF(INTRK.LE.30) IATYPE=2	AVSIN160
160	IF(INTRK.LE.20) IATYPE=1	AVSIN161
	ICNTRL=1	AVSIN162
	NTRKS(IATYPE)=NTRKS(IATYPE)+1	AVSIN163
	INTRK=PTTRK(IATYPE)+NTRKS(IATYPE)	AVSIN164
	I=I+1	AVSIN165
165	TRUCK(I) = INTRK	AVSIN166
	IF(INTIM.NE.0) RTTIM(INTRK) = FLOAT(INTIM)	AVSIN167
	IF(INTIM.EQ.0) RTTIM(INTRK) = RTMAX	AVSIN168
	TLEFT(INTRK)=RTTIM(INTRK)	AVSIN169
	IF(INCAP.NE.0) CAPAC(INTRK) = INCAP	AVSIN170
170	GO TO 3000	AVSIN171
	3100 NT=I	AVSIN172

```

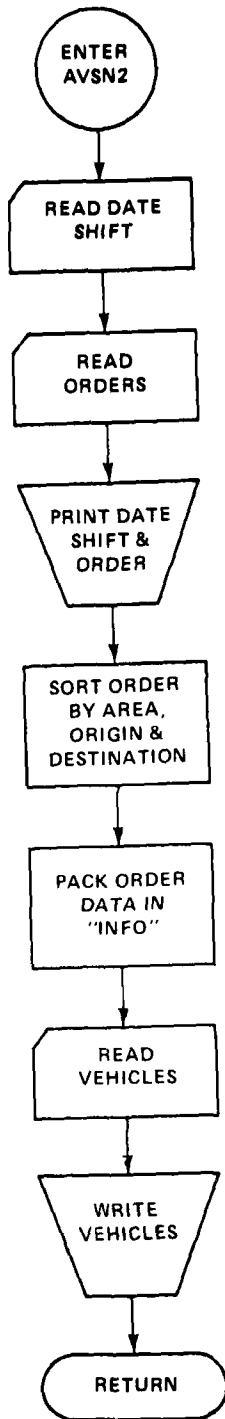
C
4000 WRITE(6,604)
      IX=0
175   DO 4400 ITYPE=1,4
      IF(NTRKS(ITYPE).LE.0) GO TO 4400
      NO=NTRKS(ITYPE)
      DO 4450 II=1,NO
      I=PTRK(ITYPE)+II
180   IX=IX+1
      IF(TRKSAV(ITYPE).LT.0) TRKSAV(ITYPE)=PTRK(ITYPE)
      IVEH=I-TRKSAV(ITYPE)
      WRITE(6,605) IX,TYPE(ITYPE),IVEH,CAPAC(II),RTLIN(II)
      IF((IX-(IX/5)*5).EQ.0) WRITE(6,606)
185   4450 CONTINUE
      4400 CONTINUE
      RETURN
100  FORMAT(4I8,2F6.1)
101  FORMAT(3I8,4F6.1)
150  102  FORMAT(10I8/7I8/(1X,10A5))
      104  FORMAT(3F6.1)
      105  FORMAT(4I8/4I8/2I8)
      106  FORMAT(6.1,I8)
      107  FORMAT(I6)
195  500  FORMAT(F5.1,1X,I2,1X,A6,1X,A6)
      501  FORMAT(I,1X,I)
      502  FORMAT(7X,A1,5X,I3)
      503  FORMAT(I2,1X,A6,1X,A6)
      504  FORMAT(A6,1X,I2,1X,I2,1X,I2,1X,I3)
200  505  FORMAT(A1,I2,1X,I3,1X,I2)
      600  FORMAT(1H1///13X,26(1H-)/13X,26H AVS SPECIAL ORDER PROGRAM/
      *
      13X,26(1H-))
      601  FORMAT(///20X,I2,1H/,I2,1H/,I2//21X,F6.1)
      602  FORMAT(/20X,6(1H-)/20X,6HOPDERS/20X,6(1H-)/)
205  603  FORMAT( 8X,I2,3X,I2,14M PALLETS FROM ,A6,4H TO ,A6)
      604  FORMAT(/20X,17(1H-)/20X,17HVEHICLES SELECTED
      *
      /20X,17(1H-)/)
      605  FORMAT(4X,I2,1X,8HVEHICLE ,A2,I2,12M CAPACITY = ,I3,
      *
      8H PALLETS, 19H, ROUTE DURATION = ,F6.1,6H MINS. )
210  606  FORMAT(1H0)
      607  FORMAT(//5X,14HBUFP OPTION = ,A6//)
      608  FORMAT(//5X,21HSPECIAL ORDER TIME = ,F6.0)
      END
AVSIN173
AVSIN174
AVSIN175
AVSIN176
AVSIN177
AVSIN178
AVSIN179
AVSIN180
AVSIN181
AVSIN182
AVSIN183
AVSIN184
AVSIN185
AVSIN186
AVSIN187
AVSIN188
AVSIN189
AVSIN190
AVSIN191
AVSIN192
AVSIN193
AVSIN194
AVSIN195
AVSIN196
AVSIN197
AVSIN198
AVSIN199
AVSIN200
AVSIN201
AVSIN202
AVSIN203
AVSIN204
AVSIN205
AVSIN206
AVSIN207
AVSIN208
AVSIN209
AVSIN210
AVSIN211
AVSIN212
AVSIN213
AVSIN214

```

ROUTINE: AVSN2

Description:

AVSN2 inputs the regular orders to be considered, the vehicles to handle them, and the date and shift of the schedules to be produced. AVSN2 also sorts orders by areas and origin and destination warehouses to reduce execution time. Order information is packed to reduced storage.



```

1      SUBROUTINE AVSN2                                AVSN2  2
C.....                                             AVSN2  3
C                                             AVSN2  4
C      AVSN INPUTS ALL PARAMETERS NECESSARY FOR THE SCHEDULE AVSN2  5
5      COMPUTATIONS. IT ALSO SORTS ORDERS AND VEHICLES TO AVSN2  6
C      TO REDUCE EXECUTION TIME. AVSN2  7
C.....                                             AVSN2  8
C.....                                             AVSN2  9
10     C                                             AVSN2 10
        INTEGER OSTART(16),OEND(16) AVSN2 11
        INTEGER CAPAC,TRUCK,ONUMB,PTTRK,PTSOR AVSN2 12
        * ,DATE AVSN2 13
        ALPHA WHNAM,AST,ILPHA,IORIG,ITERM AVSN2 14
        REAL LTIME AVSN2 15
        COMMON AVSN2 16
        */GEN/ RTTH,MINLD(4) AVSN2 17
        */WHINTG/ MWARE,NAREA(16) AVSN2 18
        */SCHEDL/ ONUMB(200),INFO(200),IDESTP(200),IAREA(200),LFRWD(200), AVSN2 19
        *LBKWD(200),LCFWD(200),LCRWD(200),ATIME(200),STOPT(200) AVSN2 20
        */TRUCKS/ PTSOR(50),TRUCK(50),CAPAC(50),STIME(50),LTIME(50). AVSN2 21
        *RTLIM(50),TLEFT(50),NT AVSN2 22
        */WHOUSE/ WHNAM(92) AVSN2 23
        */TIMTAB/ TTIME(3,45),TTIM2(6) AVSN2 24
        */MSCLNS/ NORDR,NTRKS(4),PTTRK(4) AVSN2 25
        */INOPT/ SHIFT,DATE AVSN2 26
        */SPORD/ ICNTRL,SAVTIM,ISORD AVSN2 27
        DATA TYPE/2HST,2MTR,2HTT,2HIT/ AVSN2 28
        WRITE(6,600) AVSN2 29
        READ(5,500) DATE AVSN2 30
        IMO = DATE / 10000 AVSN2 31
        IOAY = (DATE - 10000*IMO) / 100 AVSN2 32
        IYR = DATE - 10000*IMO - 100*IOAY AVSN2 33
        READ(5,501) SHIFT AVSN2 34
        SAVTIM=SHIFT AVSN2 35
        READ(5,502) IOPT AVSN2 36
        WRITE(6,601) IMO,IOAY,IYR,SHIFT,IOPT AVSN2 37
        WRITE(6,602) AVSN2 38
        NORDR=0 AVSN2 39
        2000 READ(5,503) ISIZE,IORIG,ITERM AVSN2 40
        IF(ISIZE.LT.0) GO TO 2500 AVSN2 41
        NORDR=NORDR+1 AVSN2 42
        I=NORDR AVSN2 43
        WRITE(6,603) I,ISIZE,IORIG,ITERM AVSN2 44
        IF((I-(I/5)*5).EQ.0) WRITE(6,606) AVSN2 45
        INFO(NORDR)=ISIZE+MATCH(IORIG,IAL)*10000+MATCH(ITERM,IA2)*100 AVSN2 46
        IAREA(NORDR)=IA1*100+IA2 AVSN2 47
        ONUMB(NORDR)=ACROW AVSN2 48
        GO TO 2000 AVSN2 49
        C      SORT ORDERS BY AREA AVSN2 50
50     2500 LIMIT=NORDR-1 AVSN2 51
        DO 2600 I=1,LIMIT AVSN2 52
        ISTART=I+1 AVSN2 53
        IF(ISTART.GT.NORDR) GO TO 2500 AVSN2 54
        DO 2700 J=ISTART,NORDR AVSN2 55
        IF(IAREA(I).LE.IAREA(J)) GO TO 2700 AVSN2 56
        ISAVEA=IAREA(J) AVSN2 57
        ISAVEI=INFO(J) AVSN2 58

```

	ISAVE0=ONUMB(J)	AVSN2 59
	ONUMB(J)=ONUMB(I)	AVSN2 60
60	IAREA(J)=IAREA(I)	AVSN2 61
	INFO(J)=INFO(I)	AVSN2 62
	ONUMB(I)=ISAVEC	AVSN2 63
	IAREA(I)=ISAVEA	AVSN2 64
	INFO(I)=ISAVEI	AVSN2 65
65	2700 CONTINUE	AVSN2 66
	2600 CONTINUE	AVSN2 67
	C	AVSN2 68
	C	AVSN2 69
	C MARK START OF COMMONS. ORIGIN. DESTINATIONS	AVSN2 70
70	C	AVSN2 71
	DO 2400 I=1,16	AVSN2 72
	OEND(I)=0	AVSN2 73
	2400 OSTART(I)=0	AVSN2 74
	DO 2300 I=1,NCFDR	AVSN2 75
75	JAREA=IAREA(I)/100	AVSN2 76
	IF(OSTART(JAREA).LE.0) OSTART(JAREA)=I	AVSN2 77
	IF(I.EQ.NORDR) OEND(JAREA)=I	AVSN2 78
	IF(I+1.GT.NORDR) GO TO 2300	AVSN2 79
	IF(JAREA.NE.IAREA(I+1)/100) OEND(JAREA)=I	AVSN2 80
80	2300 CONTINUE	AVSN2 81
	C SORT BY ORIGIN AND DESTINATION IN SAME AREA	AVSN2 82
	DO 2200 I=1,16	AVSN2 83
	IF(OSTART(I).LE.0) GO TO 2200	AVSN2 84
	ISTART=OSTART(I)	AVSN2 85
85	IEND=OEND(I)	AVSN2 86
	IF(ISTART.EQ.IEND) GO TO 2200	AVSN2 87
	JEND=IEND-1	AVSN2 88
	DO 2250 J=ISTART,JEND	AVSN2 89
	JSTART=J+1	AVSN2 90
90	DO 2275 K=JSTART,IEND	AVSN2 91
	IF(INFO(J).LE.INFO(K)) GO TO 2275	AVSN2 92
	ISAVEA=IAREA(J)	AVSN2 93
	ISAVEI=INFO(J)	AVSN2 94
	ISAVE0=ONUMB(J)	AVSN2 95
95	IAREA(J)=IAREA(K)	AVSN2 96
	INFO(J)=INFO(K)	AVSN2 97
	ONUMB(J)=ONUMB(K)	AVSN2 98
	IAREA(K)=ISAVEA	AVSN2 99
	INFO(K)=ISAVEI	AVSN2100
100	ONUMB(K)=ISAVEC	AVSN2101
	2275 CONTINUE	AVSN2102
	2250 CONTINUE	AVSN2103
	2200 CONTINUE	AVSN2104
	C TRUCK INPUT	AVSN2105
105	C	AVSN2106
	READ(5,504) ILPHA,(INTRK(I),I=1,3),IRTMX	AVSN2107
	RTMAX=RTTIM	AVSN2108
	IF(IRTPX.NE.0) RTMAX = FLOAT(IRTMX)	AVSN2109
	I=0	AVSN2110
	ITCNT=0	AVSN2111
110	3000 READ(5,505) AST,INTRK,INTIM,INCAP	AVSN2112
	IF(AST.NE.1M*) GO TO 3005	AVSN2113
	ITCNT=ITCNT+1	AVSN2114
	INTRK=PTTRK(4)+ITCNT	AVSN2115

115	3005 IF (INTRK.LE.0.CR.INTRK.GT.50) GO TO 3100	AVSN2116
	I=I+1	AVSN2117
	TRUCK(I) = INTRK	AVSN2118
	IF (INTIM.NE.0) RTLIM(INTRK) = FLOAT(INTIM)	AVSN2119
120	IF (INTIM.EQ.0) RTLIM(INTRK) = RTMAX	AVSN2120
	TLEFT(INTRK)=RTLIM(INTRK)	AVSN2121
	IF (INCAP.NE.0) CAPAC(INTRK) = INCAP	AVSN2122
	GO TO 3000	AVSN2123
	3100 NT=I	AVSN2124
	C	AVSN2125
125	C SORT TRUCK AND CALCULATE NTRKS	AVSN2126
	C	AVSN2127
	DO 3200 I = 1,NT	AVSN2128
	DO 3200 J = I,NT	AVSN2129
	IF (TRUCK(J).GE.TRUCK(I)) GO TO 3200	AVSN2130
130	ITEMP= TRUCK(J)	AVSN2131
	TRUCK(J) = TRUCK(I)	AVSN2132
	TRUCK(I) = ITEMP	AVSN2133
	3200 CONTINUE	AVSN2134
	DO 3600 I = 1,NT	AVSN2135
135	IF (TRUCK(I).LE.20) GO TO 3300	AVSN2136
	IF (TRUCK(I).LE.30) GO TO 3400	AVSN2137
	IF (TRUCK(I).LE.40) GO TO 3500	AVSN2138
	NTRKS(4)=NTRKS(4)+1	AVSN2139
	GO TO 3600	AVSN2140
140	3300 NTRKS(1) = NTRKS(1) + 1	AVSN2141
	GO TO 3600	AVSN2142
	3400 NTRKS(2) = NTRKS(2) + 1	AVSN2143
	GO TO 3600	AVSN2144
	3500 NTRKS(3) = NTRKS(3) + 1	AVSN2145
145	3600 CONTINUE	AVSN2146
	C	AVSN2147
	WRITE(6,604)	AVSN2148
	DO 4400 JX=1,NT	AVSN2149
	I=TRUCK(JX)	AVSN2150
150	ITYPE=1	AVSN2151
	IF (I.GT.PTRK(2)) ITYPE=2	AVSN2152
	IF (I.GT.PTRK(3)) ITYPE=3	AVSN2153
	IF (I.GT.PTRK(4)) ITYPE=4	AVSN2154
	IVEH=I-PTRK(ITYPE)	AVSN2155
155	WRITE(6,605) JX,TYPE(ITYPE),IVEH,CAPAC(I),RTLIM(I)	AVSN2156
	IF ((JX-(JX/5)*5).EQ.0) WRITE(6,606)	AVSN2157
	4400 CONTINUE	AVSN2158
	RETURN	AVSN2159
	500 FORMAT (I6)	AVSN2160
160	501 FORMAT (F5.1)	AVSN2161
	502 FORMAT (I1)	AVSN2162
	503 FORMAT (I2,1X,A6,1X,A6)	AVSN2163
	504 FORMAT (A6,1X,I2,1X,I2,1X,I2,1X,I3)	AVSN2164
	505 FORMAT (A1,I2,1X,I2,1X,I2)	AVSN2165
165	600 FORMAT (1H///13X,2((1H-)/13X,26H AVS REGULAR ORDER PROGRAM/ * 13X,26(1H-))	AVSN2166
	601 FORMAT (///20X,I2,1H/,I2,1H/,I2//21X,F6.1//21X,5H(OPT=,I1,14))	AVSN2167
	602 FORMAT (/20X,6(1H-)/20X,5HORDERS/20X,6(1H-)/)	AVSN2168
	603 FORMAT (8X,I2,3X,I2,14H PALLETIS FROM ,A6,4H TO ,A6)	AVSN2169
170	604 FORMAT (/20X,17(1H-)/20X,17HVEHICLES SELECTED * /20X,17(1H-)/)	AVSN2170
	605 FORMAT (4X,I2,1X,8HVEHICLE ,A2,I2,12H CAPACITY = ,I3, * 8H PALLETIS, 15H, ROUTE DURATION = ,F6.1,6H MINS.)	AVSN2171
	606 FORMAT (1H0)	AVSN2172
175	END	AVSN2173
		AVSN2174
		AVSN2175
		AVSN2176

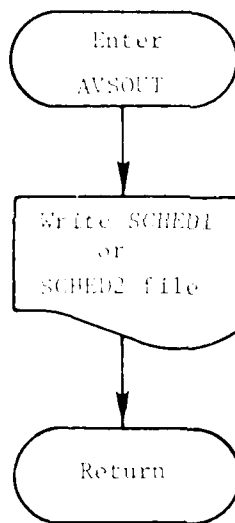
ROUTINE: AVSOUT

ARGUMENT:

- L - Disk unit number of SCHED file
- 1, SCHED1
- 2, SCHED2

Description:

AVSOUT creates SCHED1 or SCHED2 from the order lists and other variable values at the completion of each run.



```

1          SUBROUTINE AVSOUT(L)                                AVSOUT 2
C                                                    AVSOUT 3
C .....                                                    AVSOUT 4
C                                                    AVSOUT 5
5          AVSOUT CREATED UPDATE FILES TO BE USED BY NEXT AVS2 RUN AVSOUT 6
C .....                                                    AVSOUT 7
C .....                                                    AVSOUT 8
C .....                                                    AVSOUT 9
          INTEGER PTSOR,CAPAC,TRUCK,DATE,PTTRK,ONUMB AVSOUT 10
          INTEGER TRKSAV,SORDR AVSOUT 11
          ALPHA WHNAM AVSOUT 12
          REAL LTIME AVSOUT 13
          COMMON AVSOUT 14
          * /SAVFP/ NSAV(4),TRKSAV(4) AVSOUT 15
          * /SPORD/ ICNTRL,SAVTIM,SORDR AVSOUT 16
          * /WHINTG/ NWARE,NAREA(16) AVSOUT 17
          * /SCHEDL/ ONUMB(20),INFO(20),IDESTP(20),IAREA(20),LFRWD(20), AVSOUT 18
          * LBRWD(20),LCFWD(20),LCRWD(20),ATIME(20),STOPT(20) AVSOUT 19
          * /TRUCKS/ PTSOR(5),TRUCK(5),CAPAC(5),STIME(5),LTIME(5), AVSOUT 20
          * RTLIM(5),TLEFT(5),NT AVSOUT 21
          * /WHOUSE/ WHNAM(92) AVSOUT 22
          * /TIMTAB/ TTIME(3,45),TTIM2(6) AVSOUT 23
          * /MSGLNS/ NORDR,NTRKS(4),PTTRK(4) AVSOUT 24
          * /INOPT/ SHIFT,DATE AVSOUT 25
          REWIND L AVSOUT 26
          DO 200 I=1,4 AVSOUT 27
          IF (TRKSAV(I).GE.0) PTTRK(I)=TRKSAV(I) AVSOUT 28
200 NTRKS(I)=NTRKS(I)+NSAV(I) AVSOUT 29
          NT= NTRKS(1)+NTRKS(2)+NTRKS(3) AVSOUT 30
          NT=NT+NTRKS(4) AVSOUT 31
          WRITE(L,102) NWARE,(NAREA(I),I=1,16),(WHNAM(I),I=1,NWARE) AVSOUT 32
          WRITE(L,100) (ONUMB(I),INFO(I),IDESTP(I),IAREA(I), AVSOUT 33
          * LFRWD(I),LBRWD(I),LCFWD(I),LCRWD(I),ATIME(I),STOPT(I), AVSOUT 34
          * I=1,20) AVSOUT 35
          WRITE(L,101) (PTSOR(I),TRUCK(I),CAPAC(I),STIME(I),LTIME(I), AVSOUT 36
          * RTLIM(I),TLEFT(I),I=1,50) AVSOUT 37
          WRITE(L,104) ((TTIME(I,J),I=1,3),J=1,45),TTIM2 AVSOUT 38
          WRITE(L,105) PTTRK,NTRKS,NORDR ,SORDR AVSOUT 39
          WRITE(L,106) SHIFT,DATE AVSOUT 40
          MARK=999 AVSOUT 41
          WRITE(L,107) MARK AVSOUT 42
          ENDFILE L AVSOUT 43
          RETURN AVSOUT 44
          100 FORMAT(A18,2F6.1) AVSOUT 45
          101 FORMAT(3I8,4F6.1) AVSOUT 46
          102 FORMAT(10I8/7I8/(1X,10A6)) AVSOUT 47
          104 FORMAT(3F6.1) AVSOUT 48
          105 FORMAT(4I8/4I8/2I8) AVSOUT 49
          106 FORMAT(F6.1,I8) AVSOUT 50
          107 FORMAT(I6) AVSOUT 51
          END AVSOUT 52

```

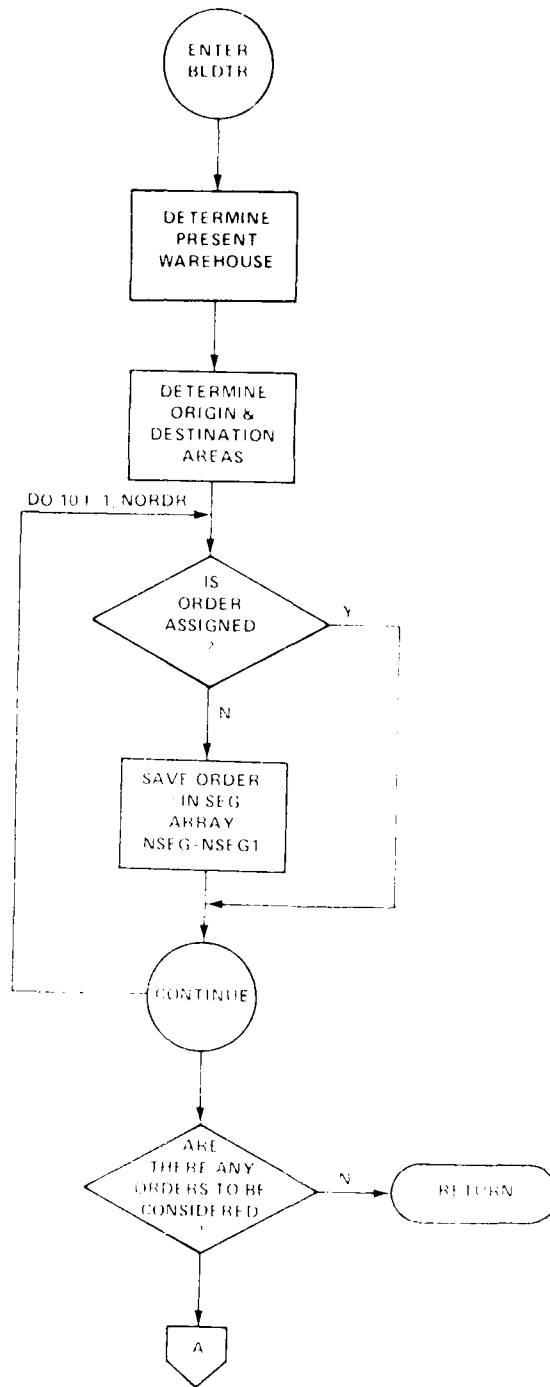
ROUTINE: BLDTR

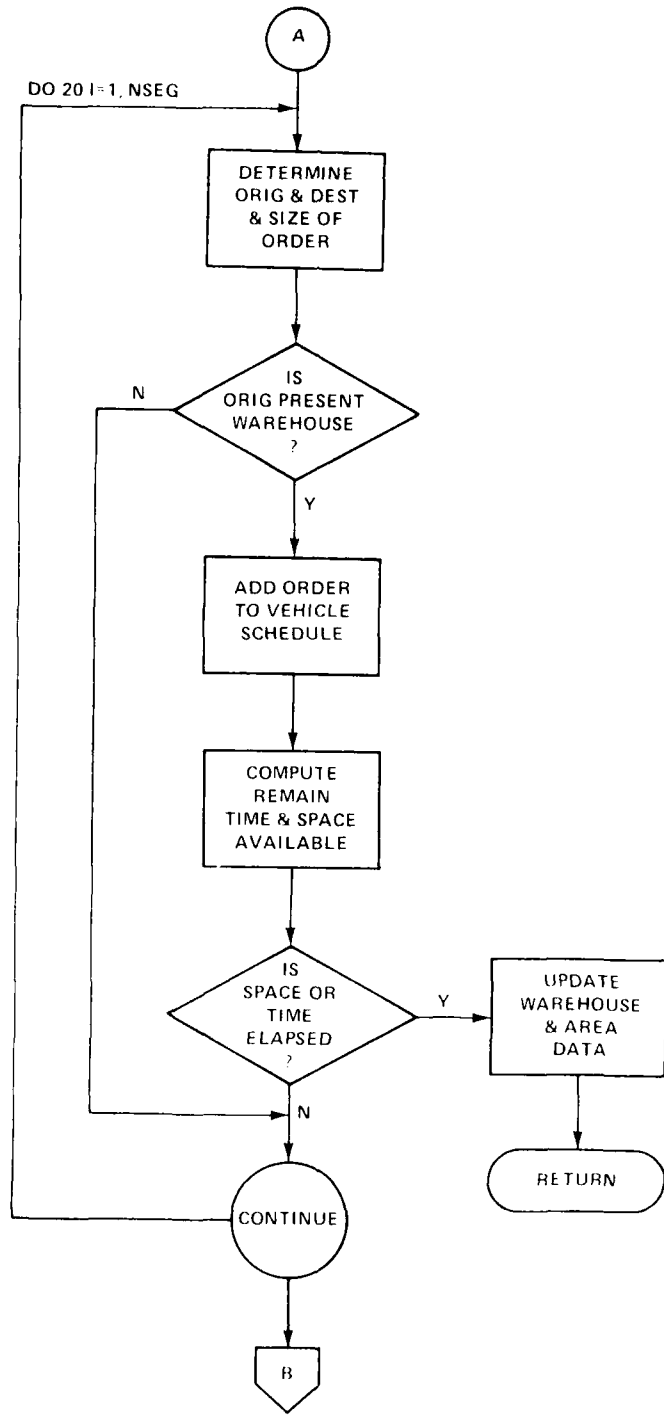
ARGUMENTS:

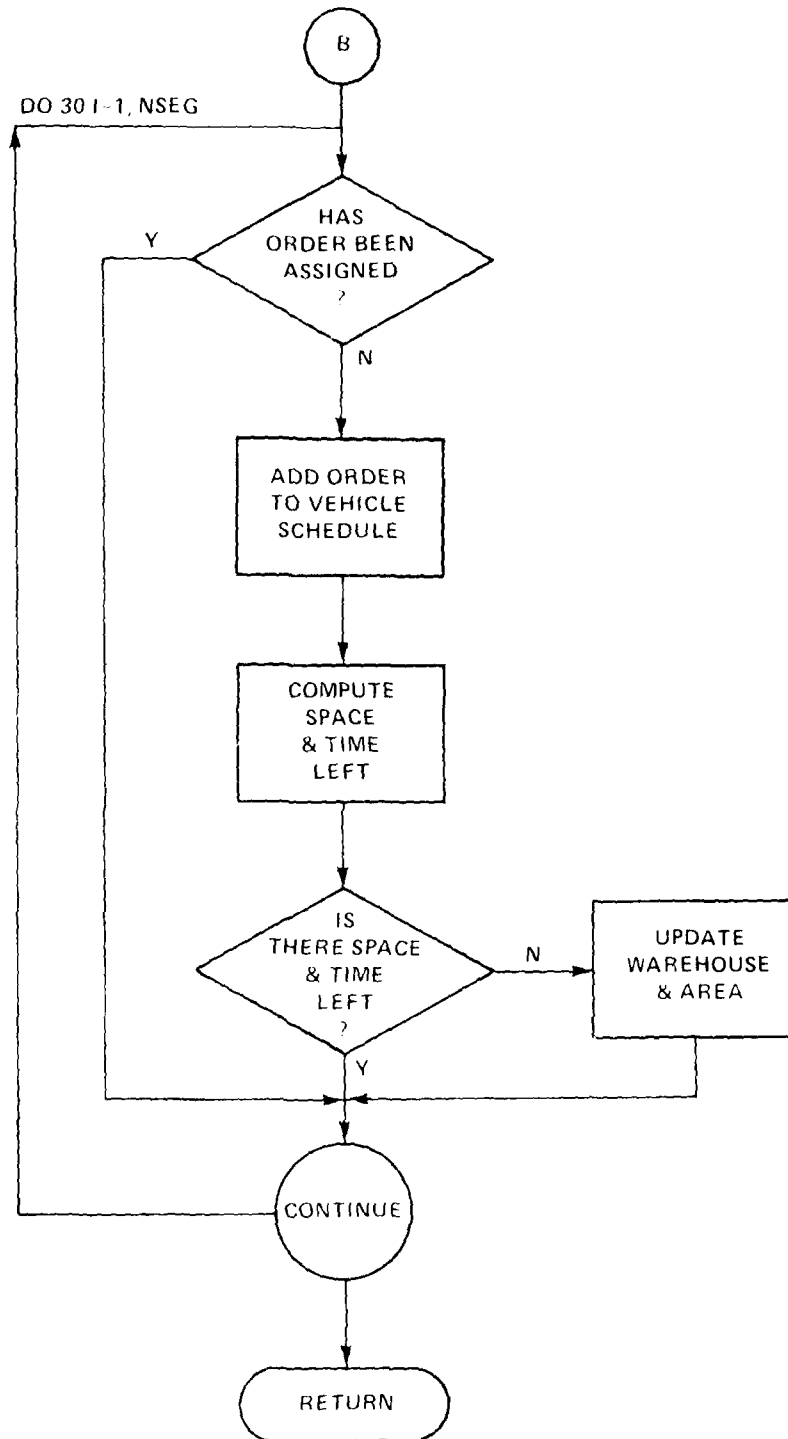
IFILL - Number of order parts assigned to the schedule segment
(Sign of IFILL indicates direction of delivery)
IRCAP - Remaining vehicle capacity given in pallets
LSTORD - List number of last order added to route segment

Description:

BLDTR builds schedule segments considering space available on the vehicle and the sequence of order delivery for specified origin and destination areas. Orders selected are set negative and linked to the vehicle list. IFILL is decremented and IRCAP is reduced by the size of the order loaded. BLDTR assigns orders to the vehicle in a "first on, last off" manner.







```

1      SUBROUTINE BLOTR(IFILL,IRCAP,LSTOR)          3L0TR      2
C.....                                     3L0TR      3
C      THIS SUBROUTINE BUILDS SCHEDULE SEGMENTS BY USING 3L0TR      4
C      THE TRANSPORTER TRUCK ORDER OF LOAD / UNLOAD 3L0TR      5
3      ----- FIRST ON - LAST OFF ----- 3L0TR      6
C      3L0TR      7
C      ITRUCK - CURRENT VEHICLE NO 3L0TR      8
C      ITYPE - VEHICLE TYPE 3L0TR      9
C      LASTW - LAST WHOUSE VISITED 3L0TR     10
C      LASTA - LAST AREA VISITED 3L0TR     11
C      NEXTA - NEXT AREA TO BE VISITED 3L0TR     12
C.....                                     3L0TR     13
C      3L0TR     14
C      INTEGER ONUMB,TRUCK,PTTRC 3L0TR     15
C      INTEGER SORDR(20),CAPAC,PTSOR 3L0TR     16
C      INTEGER SPLTSM(3) 3L0TR     17
C      ALPHA WHNAM 3L0TR     18
C      REAL LTIME 3L0TR     19
C      COMMON 3L0TR     20
C      *GEN/ RTTIM,MINLD(4) 3L0TR     21
C      *SCHFDL/ ONUMR(20),INF(200),IESTP(200),IAREA(200),LFR4D(200), 3L0TR     22
C      *LBKWD(200),LCFWD(200),LOR4D(200),ATIME(200),STOPT(200) 3L0TR     23
C      *TRUCKS/ PTSO(50),TRUCK(50),CAPAC(50),STIME(50),LTIME(50), 3L0TR     24
C      *RTTIM(50),LLEFT(50),NT 3L0TR     25
C      *MSCLNS/ NORDR,NTRKS(4),PTTRC(4) 3L0TR     26
C      *BLOS/ LASTW,LASTA,NEXTA,ITRUCK,ITYPE,IPASS,NPALTS,TIME 3L0TR     27
C      *SPORO/ ICNTRL,SAVTIM,ISORD 3L0TR     28
C      *WHOUSE/ WHNAM(92) 3L0TR     29
C      *SPLIT/ NSPLIT 3L0TR     30
C      *LCADSV/ LOAD 3L0TR     31
C      NPALTS=0 3L0TR     32
C      NSPLIT=NSPLIT 3L0TR     33
C      LSTOR=0 3L0TR     34
C      ISTART=1 3L0TR     35
C      IEAD=NORDR 3L0TR     36
C      NSEG - NO ORDERS ON SCHEDULE SEGMENT 3L0TR     37
C      SORDR - SAVE ORDERS OF SEGMENT 3L0TR     38
C      26 NSEG=0 3L0TR     39
C      ITSUM=LOAD 3L0TR     40
C      JAREA= LASTA*100+NEXTA 3L0TR     41
C      DO 100 I=ISTART,IEAD 3L0TR     42
C      IF (IPASS.GT.2) GO TO 35 3L0TR     43
C      IF (ONUMR(I).GE.0) GO TO 100 3L0TR     44
C      IF (ONUMR(I).GT.(-100)) GO TO 100 3L0TR     45
C      35 IF (INFO(I).LE.0) GO TO 100 3L0TR     46
C      IF (JAREA.NE. IAREA(I)) GO TO 100 3L0TR     47
C      IJ=INFO(I)/10000 3L0TR     48
C      IT=MOD (INFO(I)/100,100) 3L0TR     49
C      IF (NOW (ITYPE,WHNAM(I0),WHNAM(IT)).EQ.1) GO TO 100 3L0TR     50
C      75 NSEG=NSEG+1 3L0TR     51
C      ITSUM=ITSUM+MOD (INFO(I),100) 3L0TR     52
C      SORDR (NSEG)= I 3L0TR     53
C      100 CONTINUE 3L0TR     54
C      ITSUM=0 3L0TR     55
C      JAREA=NEXTA*100+NEXTA 3L0TR     56
C      IF (NSEG.LE.0) RETURN 3L0TR     57
C      IF (NSEG.NE.1) GO TO 101 3L0TR     58

```

	IORD=SORDR(1)	RLDTR	59
	ITSUM=MOD(INFO(IORD),100)	RLDTR	60
60	GO TO 102	RLDTR	61
	101 DO 175 I=1,MSEG	RLDTR	62
	IORD=SORDR(I)	RLDTR	63
	IT=MOD(INFO(IORD)/100,100)	RLDTR	64
	DO 180 II=1,NORDR	RLDTR	65
65	IF(IORD.EQ.II) GO TO 181	RLDTR	65
	IT=MOD(INFO(IORD)/100,100)	RLDTR	67
	IF(INFO(II)/10000.NE.IT) GO TO 180	RLDTR	68
	IF(MOD(IAREA(II),100).NE.NEXTA) GO TO 180	RLDTR	69
	IF(ITSAVE.EQ.0) GO TO 195	RLDTR	70
70	IF(ITSAVE.EQ.IT) GO TO 175	RLDTR	71
	185 ISUM=0	RLDTR	72
	DO 181 K=1,NORDR	RLDTR	73
	IF(INFO(K).LE.G) GO TO 191	RLDTR	74
	IF(JAREA.NE.IAREA(K)) GO TO 181	RLDTR	75
75	ISUM=ISUM+MOD(INFO(K),100)	RLDTR	76
	181 CONTINUE	RLDTR	77
	IF(ISUM.LT.MINLD(ITYPE)) GO TO 180	RLDTR	78
	ITSAVE=IT	RLDTR	79
	IF(MOD(IPASS,2).EQ.0) IPASS=IPASS-1	RLDTR	80
80	MSEG=I	RLDTR	81
	186 MSEG=MSEG-1	RLDTR	82
	IF(MSEG.LE.0) GO TO 175	RLDTR	83
	JORD=SORDR(MSEG)	RLDTR	84
	IF(MOD(INFO(JORD)/100,100).EQ.IT) GO TO 186	RLDTR	85
85	ITSUM=ITSUM+MOD(INFO(JORD),100)	RLDTR	86
	SORD(MSEG)=0	RLDTR	87
	GO TO 186	RLDTR	88
	180 CONTINUE	RLDTR	89
	175 CONTINUE	RLDTR	90
90	102 IF(ITSUM.GE.MINLD(ITYPE)) GO TO 103	RLDTR	91
	IF(NXTA.EQ.LASTA) RETURN	RLDTR	92
	IF(MOD(IPASS,2).NE.0) RETURN	RLDTR	93
	103 CONTINUE	RLDTR	94
	CONSIDER FIRST ORDERS WHERE ORIGIN =LASTA	RLDTR	95
95	JSAY=0	RLDTR	96
	LTH=0	RLDTR	97
	IF(MSEG.EQ.1) GO TO 350	RLDTR	98
	MSEG=MSEG-1	RLDTR	99
	DO 300 I=1,MSEG	RLDTR	100
	IF(SORDR(I).LE.0) GO TO 300	RLDTR	101
	IORD=SORDR(I)	RLDTR	102
	ISTART=I+1	RLDTR	103
	DO 400 J=ISTART,MSEG	RLDTR	104
	IF(SORDR(J).LE.0) GO TO 400	RLDTR	105
105	JORD=SORDR(J)	RLDTR	106
	IF(MOD(INFO(IORD),100).NE.MOD(INFO(JORD),100)) GO TO 801	RLDTR	107
	SUM(I)=JORD	RLDTR	108
	SORD(J)=IORD	RLDTR	109
	IORD=JORD	RLDTR	110
110	800 CONTINUE	RLDTR	111
	300 CONTINUE	RLDTR	112
	350 DO 1000 I=1,3	RLDTR	113
	1000 SPLTSM(I)=0	RLDTR	114
	DO 1100 I=1,MSEG	RLDTR	115

115	IF (SORDR(I).LE.0) GO TO 1100	RLDTR	116
	ICORR=SORDR(I)	RLDTR	117
	ISIZ=MOD(INFO(ICORR),100)	RLDTR	118
	IF (ISIZ.EQ.IRCAP) SPLTSM(1)=SPLTSM(1)+ISIZ	RLDTR	119
	IF (ISIZ.LT.IRCAP) SPLTSM(2)=SPLTSM(2)+ISIZ	RLDTR	120
121	SPLTSM(3)=SPLTSM(3)+ISIZ	RLDTR	121
1100	CONTINUE	RLDTR	122
	SPLTSM(1)=SPLTSM(1)+SPLTSM(2)	RLDTR	123
	NDEST=0	RLDTR	124
	DO 2000 ISPLIT=1,MSPLIT	RLDTR	125
125	IF (NEXTA.EQ.LASTA.AND.SPLTSM(ISPLIT).LT.MINLD(ITYPE)) GO TO 2000	RLDTR	126
	DO 200 I=1,NSEG	RLDTR	127
	IF (SORDR(I).LE.0) GO TO 200	RLDTR	128
	IORD=SORDR(I)	RLDTR	129
	IF (INFO(IORD)/10000.NE.LASTW) GO TO 200	RLDTR	130
131	C FROM PALETS	RLDTR	131
	IPLTS=MOD(INFO(IORD),100)	RLDTR	132
	GO TO (2001,2002,2003),ISPLIT	RLDTR	133
	2001 IF (IPLTS.EQ.IRCAP) GO TO 300	RLDTR	134
	GO TO 200	RLDTR	135
135	2002 IF (IPLTS.GT.IRCAP) GO TO 200	RLDTR	135
	GO TO 300	RLDTR	137
	2003 IF (IPLTS.LE.IRCAP) GO TO 300	RLDTR	138
	IF (NEXTA.NE.LASTA) GO TO 305	RLDTR	139
	IF (NDEST.EQ.0) NDEST=#)((INFO(IORD)/100,100)	RLDTR	140
140	IF (NDEST.NE.0) MOD(INFO(IORD)/100,100) GO TO 200	RLDTR	141
	305 ISAVE=IPLTS-IRCAP	RLDTR	142
	C FROM NEW INFO/ORDER ELEMENT	RLDTR	143
	INFO(ICORR)=INFO(IORD)/100*100+ISAVE	RLDTR	144
	NORDR=NORDR+1	RLDTR	145
145	IAREA(NORDR)=IAREA(IORD)	RLDTR	145
	INFO(NORDR)=-(INFO(IORD)/100*100+IRCAP)	RLDTR	147
	ONUM1(NORDR)=-IABS(ONUM1(IORD))	RLDTR	148
	ONUM2(NORDR)=ONUM2(IORD)	RLDTR	149
	NPALTS=NPALTS+IRCAP	RLDTR	150
151	C COMPUTE THE TIME REMAINING FOR VEHICLE	RLDTR	151
	TLEFT(ITRUCK)=TLEFT(ITRUCK)-ITIME*2.*(LYTIME(ITRUCK)*FLOAT(ITCAP)	RLDTR	152
	1* STIME(ITRUCK))	RLDTR	153
	IRCAP=0	RLDTR	154
	C LINK SEGMENT TO SCHEDULE	RLDTR	155
155	LSTART=NORDR	RLDTR	156
	IF (PTSOR(ITRUCK).LE.0) PTSOR(ITRUCK)=NORDR	RLDTR	157
	LSTART=PTSOR(ITRUCK)	RLDTR	158
	ISAVE=LSTART	RLDTR	159
	LINK=LEFWD(LSTART)	RLDTR	160
160	325 IF (LINK.LE.0) GO TO 350	RLDTR	161
	ISAVE=LINK	RLDTR	162
	LINK=LEFWD(LINK)	RLDTR	163
	GO TO 325	RLDTR	164
	350 IF (ISAVE.EQ.NORDR) GO TO 360	RLDTR	165
165	LEFWD(ISAVE)=NORDR	RLDTR	165
	LEFWD(NORDR)=ISAVE	RLDTR	167
	360 IF ILL=IFILL-1	RLDTR	168
	IJFSTP(NORDR)=IFILL	RLDTR	169
	IF (IFILL.EQ.(-1)) GO TO 370	RLDTR	170
170	TLEFT(ITRUCK)=TLEFT(ITRUCK)+TIME	RLDTR	171
	370 IJFST=MOD(IABS(INFO(NORDR)/100,100)	RLDTR	172

	IF (LTM.LE.0) LTM=IDEST	RL0TR	173
	IF (JSAVE.EQ.0) JSAVE=IDEST	RL0TR	174
	IF (JSAVE.NE.IDEST) TLEFT(ITRUCK)=TLEFT(ITRUCK)-2.	RL0TR	175
175	LAST=NEXTA	RL0TR	175
	LASTW=LTA	RL0TR	177
	RETURN	RL0TR	179
	300 IF (NEXTA.NE.LASTA) GO TO 375	RL0TR	179
	IF (NDEST.EQ.0) NDEST=NO((INFO(IORDR)/100,100)	RL0TR	180
180	IF (NDEST.NE.MOD(INFO(IORDR)/100,100)) GO TO 200	RL0TR	181
	375 IFILL=IFILL-1	RL0TR	182
	NPALTS=NPALTS+IPLTS	RL0TR	183
	IRCAP=IRCAP-IPLTS	RL0TR	184
	IDEST=MOD(INFO(IORDR)/100,100)	RL0TR	185
185	COMPUTE TIME REMAINING FOR VEHICLE	RL0TR	186
	TLEFT(ITRUCK)=TLEFT(ITRUCK)-(TIME+2.*(STIME(ITRUCK)+LTIME(ITRUCK)	RL0TR	187
	: *FLOAT(IPLTS)))	RL0TR	188
	IF (JSAVE.EQ.0) JSAVE=IDEST	RL0TR	189
	IF (JSAVE.NE.IDEST) TLEFT(ITRUCK)=TLEFT(ITRUCK)-2.	RL0TR	190
190	JSAVE=IDEST	RL0TR	191
	IF (IFILL.EQ.(-1)) GO TO 310	RL0TR	192
	TLEFT(ITRUCK)=TLEFT(ITRUCK)+TIME	RL0TR	193
	LINK=SEGMENT TO SCHEDULE	RL0TR	194
	340 INFO(IORDR)=-INFO(IORDR)	RL0TR	195
195	IF (LTM.LE.0) LTM=IDEST	RL0TR	196
	IDESTP(IORDR)=IFILL	RL0TR	197
	LSTO=IORDR	RL0TR	198
	IF (PTSOR(ITRUCK).LE.0) PTSOR(ITRUCK)=IORDR	RL0TR	199
	LSTA=PPTSOR(ITRUCK)	RL0TR	200
200	ISAVE=LSTART	RL0TR	201
	LINK=LFRWD(LSTART)	RL0TR	202
	410 IF (LINK.LE.0) GO TO 400	RL0TR	203
	ISAVE=LINK	RL0TR	204
	LINK=LFRWD(LINK)	RL0TR	205
205	GO TO 410	RL0TR	206
	400 IF (ISAVE.EQ.IORDR) GO TO 421	RL0TR	207
	LFRWD(ISAVE)=IORDR	RL0TR	208
	LAKWD(IORDR)=ISAVE	RL0TR	209
210	420 IF (IRCAP.LE.0) GO TO 421	RL0TR	210
	IF (TLEFT(ITRUCK).LE.0.0) GO TO 421	RL0TR	211
	IF (NO(IPASS,2).EQ.0) GO TO 210	RL0TR	212
	IFEST=NEXTA*100+NEXTA	RL0TR	213
	IF (45 JJ=1,NORDR	RL0TR	214
	IF (TAREALJJ).NE.JTEST) GO TO 425	RL0TR	215
215	IF (INFO(JJ)/10000.NE.LTA) GO TO 425	RL0TR	216
	GO TO 421	RL0TR	217
	425 CONTINUE	RL0TR	218
	GO TO 210	RL0TR	219
	421 LAST=NEXTA	RL0TR	220
220	LASTW=LTA	RL0TR	221
	RETURN	RL0TR	222
	200 CONTINUE	RL0TR	223
	2100 CONTINUE	RL0TR	224
	NDEST=0	RL0TR	225
225	DO 500 ISPLIT=1,MSPLIT	RL0TR	226
	IF (NEXTA.EQ.LASTA.AND.SPLTSM(SPLIT).LT.MINLJ(ITYPE)) GO TO 500	RL0TR	227
	DO 500 I=1,NSIG	RL0TR	228
	IOFD=-SORDR(I)	RL0TR	229

		IF(IORDR.EQ.0) GO TO 501	RLNTR	237
230		IF(INFA.NE.LASTA) GO TO 500	RLNTR	238
	C	ENUGH PALLETS	RLNTR	239
		IPLTS=MOD(INFO(IORDR),100)	RLNTR	240
		GO TO (5001,5002,5003),ISPLIT	RLNTR	241
235	5001	IF(IPLTS.EQ.IRCAP) GO TO 600	RLNTR	242
		GO TO 500	RLNTR	243
	5002	IF(IPLTS.GT.IRCAP) GO TO 500	RLNTR	244
		GO TO 601	RLNTR	245
	5003	IF(IPLTS.LE.IRCAP) GO TO 600	RLNTR	246
		IF(NEXTA.NE.LASTA) GO TO 605	RLNTR	247
240		IF(NDEST.EQ.0) NDEST=MOD(INFO(IORDR)/100,100)	RLNTR	248
		IF(NDEST.NE.MOD(INFO(IORDR)/100,100)) GO TO 500	RLNTR	249
	605	ISAVE=IPLTS-IRCAP	RLNTR	250
	C	FORM NEW INFO/ORDER ELEMENTS	RLNTR	251
		INFO(IORDR)=INFO(IORDR)/100*100+ISAVE	RLNTR	252
245		NORDR=NORDR+1	RLNTR	253
		INFO(NORDR)=-((INFO(IORDR)/100*100+IRCAP)	RLNTR	254
		ONUM(IORDR)=-IABS(ONUM(IORDR))	RLNTR	255
		ONUM(NORDR)=ONUM(IORDR)	RLNTR	256
		IAREA(NORDR)=IAREA(IORDR)	RLNTR	257
250		NPALTS=NPALTS+IRCAP	RLNTR	258
	C	COMPUTE THE TIME REMAINING FOR VEHICLE	RLNTR	259
		TLEFT(ITRUCK)=TLEFT(ITRUCK)-(TIME+2.*(LTIME(ITRUCK)*FLOAT(IPCAP)	RLNTR	260
		1*STIME(ITRUCK)))	RLNTR	261
		IRCAP=J	RLNTR	262
255	C	LINK SEGMENT TO SCHEDULE	RLNTR	263
		LSTART=NORDR	RLNTR	264
		IF(PTSOR(ITRUCK).LE.G) PTSOR(ITRUCK)=NORDR	RLNTR	265
		LSTART=PTSOR(ITRUCK)	RLNTR	266
		ISAVE=LSTART	RLNTR	267
260		LINK=LFRWD(LSTART)	RLNTR	268
	625	IF(LINK.LE.0) GO TO 650	RLNTR	269
		ISAVE=LINK	RLNTR	270
		LINK=LFRWD(LINK)	RLNTR	271
		GO TO 625	RLNTR	272
265	550	IF(ISAVE.EQ.NORDR) GO TO 660	RLNTR	273
		LFRWD(ISAVE)=NORDR	RLNTR	274
		LBACK(NORDR)=ISAVE	RLNTR	275
	660	IFILL=IFILL-1	RLNTR	276
		IDEST(NORDR)=IFILL	RLNTR	277
270		IF(IFILL.EQ.(-1)) GO TO 570	RLNTR	278
		TLEFT(ITRUCK)=TLEFT(ITRUCK)+TIME	RLNTR	279
	570	IDEST=MOD(IABS(INFO(NORDR))/100,100)	RLNTR	280
		IF(LTW.LE.0) LTW=IDEST	RLNTR	281
		IF(JSAVE.EQ.G) JSAVE=IDEST	RLNTR	282
275		IF(JSAVE.NE.IDEST) TLEFT(ITRUCK)=TLEFT(ITRUCK)-2.	RLNTR	283
		LASTA=NEXTA	RLNTR	284
		LASTW=LTW	RLNTR	285
		RETURN	RLNTR	286
280	600	IF(NEXTA.NE.LASTA) GO TO 675	RLNTR	287
		IF(NDEST.EQ.0) NDEST=MOD(INFO(IORDR)/100,100)	RLNTR	288
		IF(NDEST.NE.MOD(INFO(IORDR)/100,100)) GO TO 500	RLNTR	289
	675	IFILL=IFILL-1	RLNTR	290
		NPALTS=NPALTS+IPLTS	RLNTR	291
		IRCAP=IRCAP-IPLTS	RLNTR	292
285		IDEST=MOD(INFO(IORDR)/100,100)	RLNTR	293

	C	COMPUTE TIME REMAINING FOR VEHICLE	R.OTR	287
		TLEFT(ITRUCK)=TLEFT(ITRUCK)-(TIME*2.*(STIME(ITRUCK)+LTIME(ITRUCK)	R.OTR	288
		1 *FLOAT(IPLTS)))	R.OTR	289
293		IF (ISAVE.EQ.0) ISAVE=IDEST	R.OTR	290
		IF (ISAVE.NE.IDEST) TLEFT(ITRUCK)=TLEFT(ITRUCK)-2.	R.OTR	291
		ISAVE=IDEST	R.OTR	292
		TLEFT(ITRUCK)=TLEFT(ITRUCK)+TIME	R.OTR	293
	C	LINK SEGMENT TO SCHEDULE	R.OTR	294
	680	INFO(IORDR)=-INFO(IORDR)	R.OTR	295
295		IF (LTM.LE.0) LTM=IDEST	R.OTR	296
		IDEST(IORDR)=IFILL	R.OTR	297
		LSTAR=IORDR	R.OTR	298
		IF (PTSOR(ITRUCK).LE.0) PTSOR(ITRUCK)=IORDR	R.OTR	299
300		LSTAR=PTSOR(ITRUCK)	R.OTR	300
		ISAVE=LSTAR	R.OTR	301
		LINK=LFRWD(LSTAR)	R.OTR	302
	710	IF (LINK.LE.0) GO TO 700	R.OTR	303
		ISAVE=LINK	R.OTR	304
		LINK=LFRWD(LINK)	R.OTR	305
305		GO TO 710	R.OTR	306
	700	IF (ISAVE.EQ.IORDR) GO TO 720	R.OTR	307
		LFRWD(ISAVE)=IORDR	R.OTR	308
		LFRWD(IORDR)=ISAVE	R.OTR	309
	720	IF (ICAP.LE.0) GO TO 721	R.OTR	310
310		IF (TLEFT(ITRUCK).LE.0.0) GO TO 721	R.OTR	311
		GO TO 500	R.OTR	312
	721	LASTA=NEXTA	R.OTR	313
		LASTW=LTM	R.OTR	314
		RETURN	R.OTR	315
315	500	CONTINUE	R.OTR	316
	5000	CONTINUE	R.OTR	317
		IF (NPALTS.LE.0) RETURN	R.OTR	318
		LASTA=NEXTA	R.OTR	319
		LASTW=LTM	R.OTR	320
		RETURN	R.OTR	321
		END	R.OTR	322

1	BLUCK DATA	ALPHA	RLOCK	2
	* NORTH,CENTR,NSYN,WEST,NSYN,		RLOCK	3
	* NSYC,NSYS,SOUTH,MCRFT,X54,		RLOCK	4
5	* ABASE,NWS,DEYTN,BRASW,GSNWS,NMEDC		RLOCK	5
	INTEGER PTSOR,CAPAC,TRUCK,ONUMB		RLOCK	6
	INTEGER PTRK		RLOCK	7
	REAL LTIME		RLOCK	8
	COMMON		RLOCK	9
10	*/GEN/ RTTIM,MINLJ(4)		RLOCK	10
	*/TIMTAB/ TTIME(3,4,5),TTM(4)		RLOCK	11
	*/WHITG/ NWAR, NAREA(16)		RLOCK	12
	*/SCHDOL/ ONUMB(200),INF(200),JESTP(200),IAREA(200),LFRMD(200),		RLOCK	13
	*L3KMD(200),LCFMD(200),L340(200),ATIME(200),STOPT(200)		RLOCK	14
15	*/TRUCKS/ PTSOR(50),TRUCK(50),CAPAC(50),STIME(50),LTIME(50),		RLOCK	15
	* RTTIM(50),TLEFT(50),NT		RLOCK	16
	*/MCDLNC/ NORDC,NTPKS(4),PTRK(4)		RLOCK	17
	COMMON /4HOUSE/		RLOCK	18
	* NORTH(14),CENTR(17),NSYN(3),WEST(3),NSYN(4),		RLOCK	19
20	* NSYC(10),NSYS(12),SOUTH(13),MCRFT(6),X54,		RLOCK	20
	* ABASE,NWS,DEYTN,BRASW,GSNWS,NMEDC		RLOCK	21
			RLOCK	22
	DATA NAREA/14,17,3,3,4,10,12,10,6,7*1/		RLOCK	23
	DATA NWAR /92/		RLOCK	24
25	DATA NORTH / 3H131, 5H1601A, 5H1601B, 4H1602, 4+1603,		RLOCK	25
	* 4H1604, 4H1615, 4H1605, 4H1621, 4H1622,		RLOCK	26
	* 4H1628,1HA,4H1620,4H1157/		RLOCK	27
	DATA CENTR / 3H53C,3H64E,3H64W,3H66E,3H66W,3H67E,3H67W,		RLOCK	28
	* 3H198,4H1178,4H1127,2H5M,2H46,4H1138,		RLOCK	29
30	* 2H56,2H43,2H5F,3H5F2/		RLOCK	30
	DATA NSYN / 1H2,1H3,1H5,1H8,2H35,3H3C,2H44,2H53,3H223/		RLOCK	31
	DATA WEST / 4H1502, 4H1503, 4H1507 /		RLOCK	32
	DATA NSYN / 2H80,3H177,4H1143,4H1199 /		RLOCK	33
	DATA NSYC / 2H98,3H187,3H216,3H218,4H1175,4H1169,		RLOCK	34
35	* 4H1171,4H1172,4H1173,4H1174 /		RLOCK	35
	DATA NSYS / 3HX10,3H193,3H224,1HL,1HM,1HY,1HP,1HQ,		RLOCK	36
	* 1AK,1HS,1AT,3HX21/		RLOCK	37
	DATA SOUTH / 2H30,3H435,2H61,2H84,3H202,3H646,3H647,3H655		RLOCK	38
	* ,3H656,2H52/		RLOCK	39
40	DATA MCRFT / 1H1,1H7,2H15,2H23,2H26,3H535 /		RLOCK	40
	DATA X54 / 3HX54 /		RLOCK	41
	DATA ABASE / 4HABASE /		RLOCK	42
	DATA NWS / 3HNWS /		RLOCK	43
	DATA DEYTN / 5HDEYTN /		RLOCK	44
45	DATA BRASW / 5HBRASW /		RLOCK	45
	DATA GSNWS / 5HGSNWS /		RLOCK	46
	DATA NMEDC / 5HNMEDC /		RLOCK	47
	DATA NTPKS/4* /		RLOCK	48
	DATA TRUCK / 50 * 0 /		RLOCK	49
50	DATA PTRK/0,20,30,40/		RLOCK	50
	DATA CAPAC/20*7,10*12,10*14,10*10/		RLOCK	51
	DATA LTIME / 20 * 0.2, 10 * 0.8, 20 * 1.7 /		RLOCK	52
	DATA STIME / 20 * 1.7, 10 * 1.7, 20 * 5.0 /		RLOCK	53
	DATA PTSOR / 50*0 /		RLOCK	54
55	DATA TTIME / 4..7..+... 11..9..5.., 13..11..8..		RLOCK	55
	* 13..11..8.., 18..16..14.., 23..21..13..		RLOCK	56
	* 30..28..25.., 27..25..22.., 31..29..26..		RLOCK	57

	*	7..5..4..	6..4..4..	7..5..4..		BLOCK	59
	*	9..7..5..	12..10..7..	20..18..15..		BLOCK	60
60	*	26..24..21..	29..27..24..			BLOCK	61
	*	8..6..4..	7..5..4..	9..7..5..		BLOCK	62
	*	14..12..9..	22..20..17..	26..24..20..		BLOCK	63
	*	29..27..24..				BLOCK	64
	*	9..7..5..	12..10..7..	20..18..15..		BLOCK	65
65	*	26..24..21..	28..26..23..	32..30..27..		BLOCK	66
	*	6..4..4..	11..9..8..	16..14..12..		BLOCK	67
	*	21..19..17..	24..22..20..			BLOCK	68
	*	8..6..5..	14..12..9..	15..13..10..		BLOCK	69
70	*	18..16..14..				BLOCK	70
	*	9..7..5..	11..9..8..	12..10..9..		BLOCK	71
	*	8..5..4..	12..7..6..			BLOCK	72
	*	7..5..3..	/			BLOCK	73
		DATA TTIM2 / 30.. 45.. 50.. 25.. 35.. 15. /				BLOCK	74
		DATA ONUM8 / 200*0 /				BLOCK	75
75		DATA INFO/200*0/				BLOCK	76
		DATA IAREA/200*0/				BLOCK	77
		DATA IDESTP/200*0/				BLOCK	78
		DATA LFRWD / 200*0 /				BLOCK	79
		DATA LBRWD / 200*0 /				BLOCK	80
80		DATA LCFWD / 200*0 /				BLOCK	81
		DATA LCRWD / 200*0 /				BLOCK	82
		DATA ATIME / 200*0. /				BLOCK	83
		DATA STOPI / 200*0. /				BLOCK	84
		DATA RTTIM/485.0/				BLOCK	85
85		DATA MINLO/5.8.14.6/				BLOCK	86
		END				BLOCK	87

ROUTINE: BUILDS

ARGUMENTS:

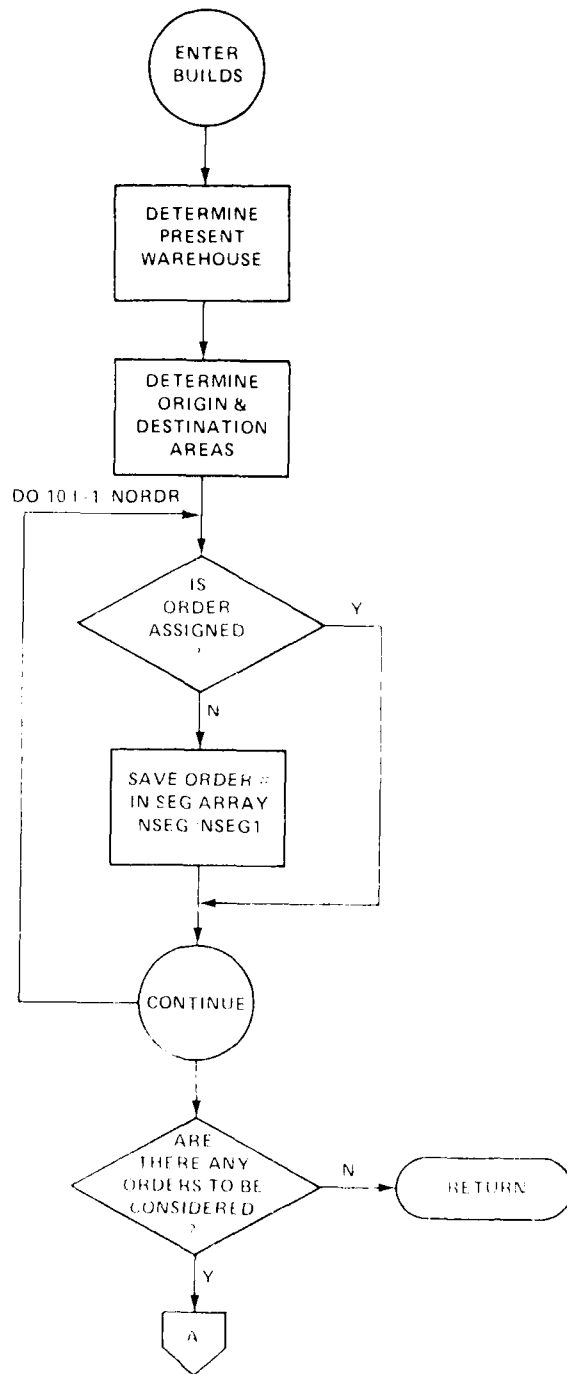
IFILL - Number of order parts assigned to schedule segment

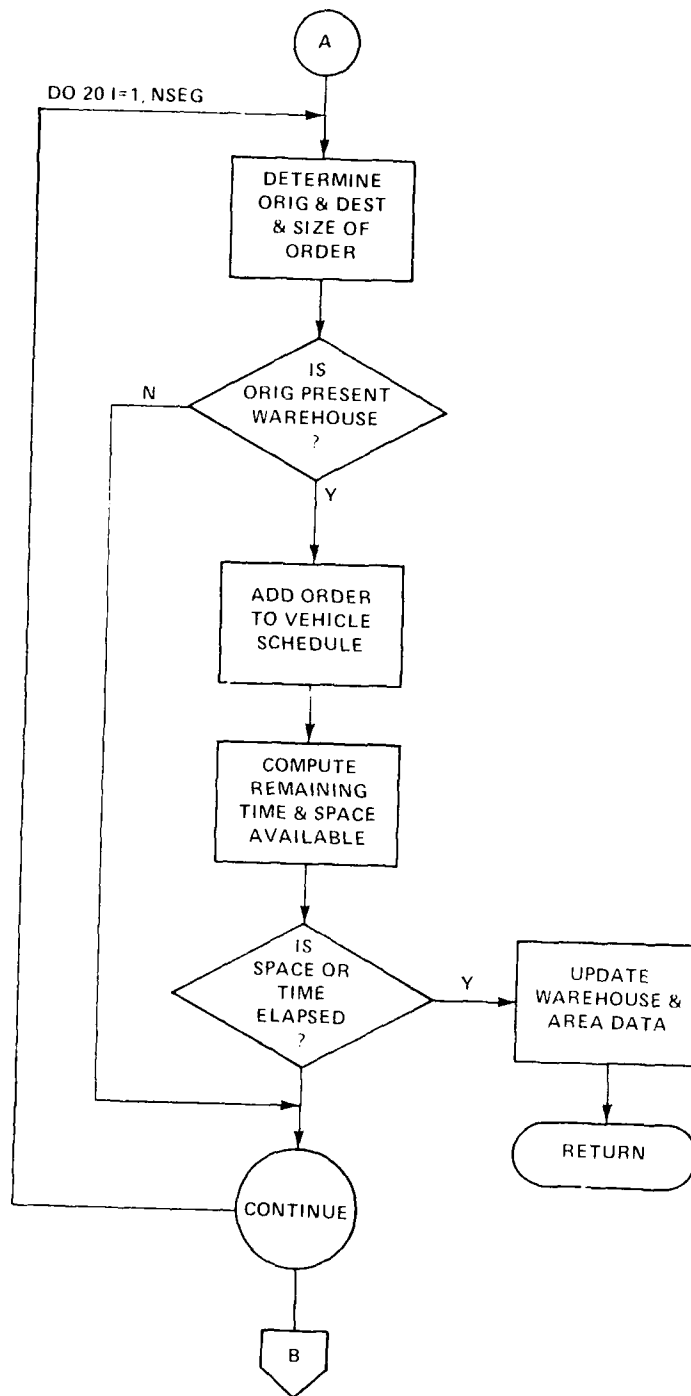
IRCRP - Remaining vehicle capacity given in pallets

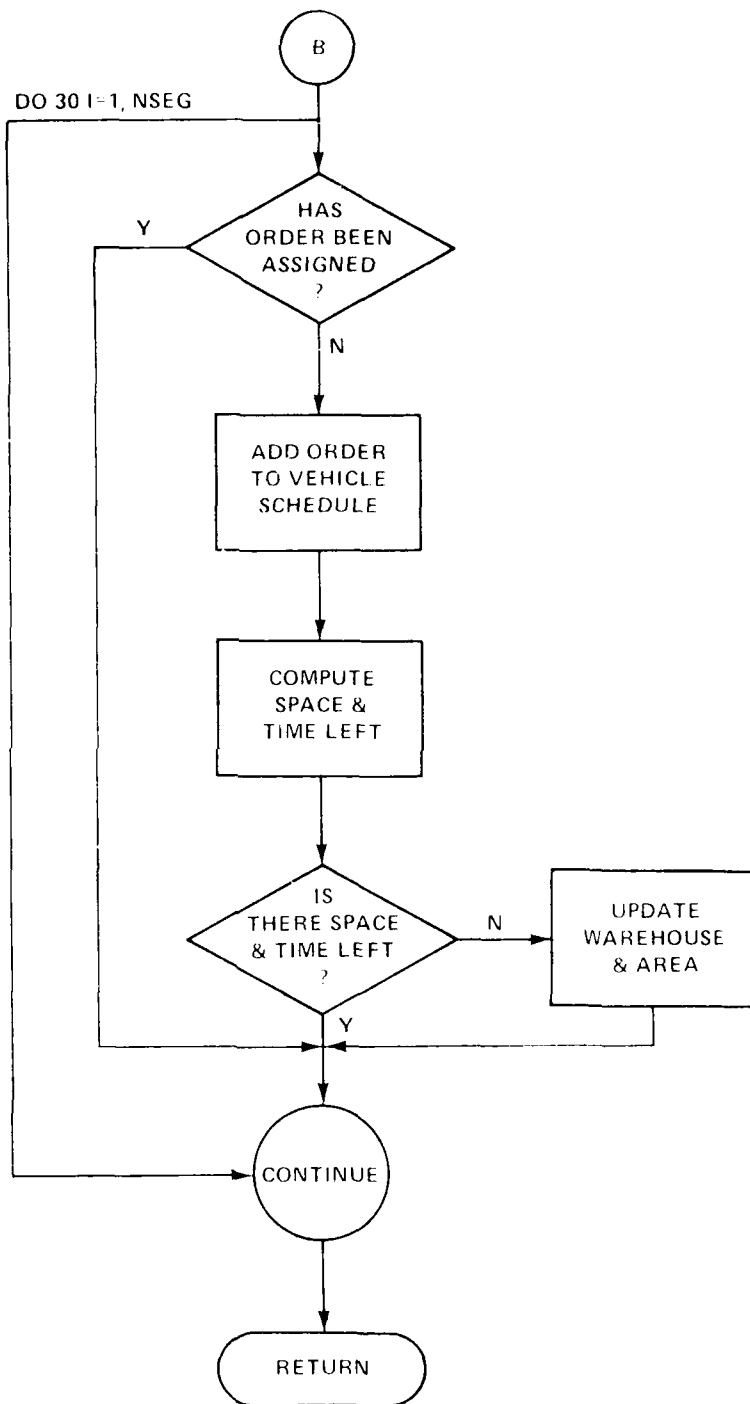
LSTORD - List number of last order added to route segment

Description:

BUILDS builds schedule segments considering space available on the vehicle and the sequence of order delivery for specified origin and destination areas. Selected orders are set negative and linked to the vehicle list. IFILL is incremented and IRCAP is reduced by the size of the order loaded. BUILDS assigns orders to a vehicle in a "first on, first off" manner.







1	SUBROUTINE BUILOS(IFILL,IRCAP,LSTORD,NDELC)	RUILOS	2
	RUILOS	3
	C THIS SUBROUTINE BUILOS SCHEDULES SEGMENTS BY USING	RUILOS	4
	C THE STRADDLE TRUCK ORDER OF LOAD / UNLOAD	RUILOS	5
5	----- FIRST ON - FIRST OFF -----	RUILOS	6
	C	RUILOS	7
	C ITRUCK - CURRENT VEHICLE NO	RUILOS	8
	C ITYPE - VEHICLE TYPE	RUILOS	9
	C LASTW - LAST WHOUSE VISITED	RUILOS	10
10	C LASTA - LAST AREA VISITED	RUILOS	11
	C NEXTA - NEXT AREA TO BE VISITED	RUILOS	12
	RUILOS	13
	C	RUILOS	14
	C INTEGER SORDR(20),CAPAC,PTSOR	RUILOS	15
15	* ,TRUCK,PTTRK,ONUMB	RUILOS	16
	C INTEGER SPLTSM(3)	RUILOS	17
	C ALPHA WHNAM	RUILOS	18
	C REAL LTIME	RUILOS	19
	C COMMON	RUILOS	20
20	* /GEN/ RTTIM,MINLO(4)	RUILOS	21
	* /SCHED/ ONUMH(20), INFO(20), IOESTP(20), IAREA(20), LFR4D(20),	RUILOS	22
	* L3KWD(20), LCFW(20), LCB4D(20), ATIME(20), STOPI(20)	RUILOS	23
	* /TRUCK/ PTSOR(50), TRUCK(50), CAPAC(50), STIME(50), LTIME(50),	RUILOS	24
	* RTTIM(50), TLEFT(50), NT	RUILOS	25
25	* /MSLNS/ NORDR, NTRKS(4), PTTRK(4)	RUILOS	26
	* /BLNS/ LASTW, LASTA, NEXTA, ITRUCK, ITYPE, IPASS, NPA, LTS, TIME	RUILOS	27
	* /SPLIT/ NSPLIT	RUILOS	28
	* /SPCKD/ IDNTRL, SAVTIM, ISORD	RUILOS	29
	* /WHOUSE/ WHNAM(92)	RUILOS	30
30	NPALTS=0	RUILOS	31
	LSTORD=0	RUILOS	32
	MSPLIT=NSPLIT	RUILOS	33
	ISTART=1	RUILOS	34
	IEND=NORDR	RUILOS	35
35	C NSEG - NO ORDERS ON SCHEDULE SEGMENT	RUILOS	36
	C SORDP - SAVE ORDERS OF SEGMENT	RUILOS	37
	26 NSEG=0	RUILOS	38
	JAREA= LASTA*100+NEXTA	RUILOS	39
	DO 100 I=ISTART, IEND	RUILOS	40
40	IF(IPASS.GT.2) GO TO 35	RUILOS	41
	IF(ONUMB(I).GE.0) GO TO 110	RUILOS	42
	IF(ONUMB(I).GT.(-100)) GO TO 100	RUILOS	43
	35 IF(INFO(I).LE.0) GO TO 100	RUILOS	44
	IF(JAREA.NE.IAREA(I)) GO TO 100	RUILOS	45
45	IO=INFO(I)/10000	RUILOS	46
	IT=MOD(INFO(I)/100,100)	RUILOS	47
	IF(NOW(ITYPE,WHNAM(IO),WHNAM(IT)).EQ.1) GO TO 100	RUILOS	48
	NSEG=NSEG+1	RUILOS	49
	SORDR(NSEG)= I	RUILOS	50
50	100 CONTINUE	RUILOS	51
	IF(NSEG.LE.0) RETURN	RUILOS	52
	C SORT BY DESTINATION WAREHOUSE	RUILOS	53
	C SUM PALLETS TO BE DELIVERED	RUILOS	54
	IF(NSEG.GT.1) GO TO 101	RUILOS	55
55	IORD=SORDR(1)	RUILOS	56
	IF(MOD(INFO(IORD),100).GE.MINLO(ITYPE)) GO TO 186	RUILOS	57
	NDELC=0	RUILOS	58

		IF (IPASS.GT.2) NDEL0=MOD(INFO(IORD),100)	RUTLOS	59
		RETURN	RUTLOS	60
83	101	LIMIT=NSEG-1	RUTLOS	61
		NDEL0=0	RUTLOS	62
		DO 45 I=1,LIMIT	RUTLOS	63
		ISTART=I+1	RUTLOS	64
		DO 40 J=ISTART,NSEG	RUTLOS	65
65		IORD=SORDR(I)	RUTLOS	66
		JORD=SORDR(J)	RUTLOS	67
		IF (MOD(INFO(IORD)/100,111).LE.MOD(INFO(JORD)/100,100)) GO TO 40	RUTLOS	68
		SORDR(J)=SORDR(I)	RUTLOS	69
		SORDR(I)=JORD	RUTLOS	70
70	40	CONTINUE	RUTLOS	71
	45	CONTINUE	RUTLOS	72
		IF (ICNTRL.EQ.0) GO TO 51	RUTLOS	73
		ISUM=0	RUTLOS	74
		ISTART=0	RUTLOS	75
75		DO 80 I=1,LIMIT	RUTLOS	76
		JORD=SORDR(I)	RUTLOS	77
		JORD=SORDR(I+1)	RUTLOS	78
		IF (ISTART.EQ.J) ISTART=I	RUTLOS	79
		ILAST=I	RUTLOS	80
83		ISUM=ISUM+MOD(INFO(IORD),100)	RUTLOS	81
		IF (MOD(INFO(IORD)/100,111).EQ.MOD(INFO(JORD)/100,100)) GO TO 55	RUTLOS	82
		IF (I.NE.LIMIT) GO TO 65	RUTLOS	83
		IF (MOD(INFO(JORD),100).GE.3) GO TO 65	RUTLOS	84
		SORDR(NSEG)=0	RUTLOS	85
85		NDEL0=NDEL0+MOD(INFO(JORD),100)	RUTLOS	86
	65	IF (ISUM.GE.3) GO TO 70	RUTLOS	87
		DO 75 K=ISTART,ILAST	RUTLOS	88
	75	SORDR(K)=0	RUTLOS	89
		NDEL0=NDEL0+ISUM	RUTLOS	90
90	70	ISUM=0	RUTLOS	91
		ISTART=0	RUTLOS	92
		ILAST=0	RUTLOS	93
		GO TO 50	RUTLOS	94
	55	IF (I.NE.LIMIT) GO TO 50	RUTLOS	95
95		ISUM=ISUM+MOD(INFO(JORD),100)	RUTLOS	96
		IF (ISUM.GE.3) GO TO 50	RUTLOS	97
		DO 60 K=ISTART,NSEG	RUTLOS	98
	60	SORDR(K)=0	RUTLOS	99
		NDEL0=NDEL0+ISUM	RUTLOS	100
100	50	CONTINUE	RUTLOS	101
	51	DO 80 I=1,NSEG	RUTLOS	102
		IF (SORDR(I).LF.0) GO TO 80	RUTLOS	103
		NDEL0=0	RUTLOS	104
	80	CONTINUE	RUTLOS	105
105		IF (IPASS.LE.2) NDEL0=0	RUTLOS	106
	0	CONSIDER FIRST ORDERS WHERE ORIGIN =LASTW	RUTLOS	107
		ITSAVE=0	RUTLOS	108
		NSEG=NSEG	RUTLOS	109
		DO 150 I=1,NSEG	RUTLOS	110
110		IF (SORDR(I).EQ.0) GO TO 150	RUTLOS	111
		IORD=SORDR(I)	RUTLOS	112
		IT=MOD(INFO(IORD)/100,100)	RUTLOS	113
		IF (ITSAVE.EQ.0) GO TO 150	RUTLOS	114
		IF (ITSAVE.EQ.IT) GO TO 150	RUTLOS	115

115	MSER=I-1	RUILOS	115
	GO TO 145	RUILOS	117
140	DO 175 II=1,NOROR	RUILOS	118
	IF(IORR.EQ.II) GO TO 175	RUILOS	119
	IF(II.NE.INFO(II)/1000) GO TO 175	RUILOS	120
120	IF(MOD(IAREA(II),100).NE.NEXTA) GO TO 175	RUILOS	121
	IF(MOD(IPASS,2).EQ.0) IPASS=IPASS-1	RUILOS	122
	ITSAVE=IT	RUILOS	123
	GO TO 150	RUILOS	124
175	CONTINUE	RUILOS	125
150	CONTINUE	RUILOS	126
185	NSPFR=NSFG	RUILOS	127
145	CONTINUE	RUILOS	128
	JSAVE=0	RUILOS	129
	NWARE=0	RUILOS	130
130	IF(NREG.EQ.1) GO TO 950	RUILOS	131
	MSGENSP=1	RUILOS	132
	DO 900 I=1,MS G	RUILOS	133
	IF(SORR(I).LE.0) GO TO 910	RUILOS	134
	IORD=SORR(I)	RUILOS	135
135	ISTAT=I*1	RUILOS	136
	DO 800 J=ISTAT,NSFG	RUILOS	137
	IF(SORR(J).LE.0) GO TO 900	RUILOS	138
	JORD=SORR(J)	RUILOS	139
140	IF(MOD(INFO(IORD),100).NE.MOD(INFO(JORD),100)) GO TO 800	RUILOS	140
	SORR(I)=JORD	RUILOS	141
	SORR(J)=IORD	RUILOS	142
	I)=JORD	RUILOS	143
800	CONTINUE	RUILOS	144
900	CONTINUE	RUILOS	145
145	950 DO 1000 I=1,3	RUILOS	146
	1000 SPLTM(I)=0	RUILOS	147
	DO 1100 I=1,NSFG	RUILOS	148
	IF(SORR(I).LE.0) GO TO 1100	RUILOS	149
	ICPR=SORR(I)	RUILOS	150
150	ISIZ=MOD(INFO(IORD),100)	RUILOS	151
	IF(ILIZ.EQ.ICAP) SPLTM(1)=SPLTM(1)+ISIZE	RUILOS	152
	IF(ISIZE.LT.ICAP) SPLTM(2)=SPLTM(2)+ISIZE	RUILOS	153
	SPLTM(3)=SPLTM(3)+ISIZE	RUILOS	154
1100	CONTINUE	RUILOS	155
155	SPLTM(2)=SPLTM(1)+SPLTM(3)	RUILOS	156
	NDFST=0	RUILOS	157
	DO 2000 ISPLIT=1,4SPLIT	RUILOS	158
	IF(SPLTM(ISPLIT).LT.MT(L)(ITYPE)) GO TO 2000	RUILOS	159
	DO 200 I=1,NSFG	RUILOS	160
160	IF(SORR(I).EQ.0) GO TO 200	RUILOS	161
	IORD=SORR(I)	RUILOS	162
	IF(INFO(IORD)/1000.NE.LASTW) GO TO 200	RUILOS	163
	ENOUGH PALLETS	RUILOS	164
	IF(NEXTA.NE.LASTA) GO TO 187	RUILOS	165
	IF(NWARE.EQ.0) NWARE=MOD(INFO(IORD)/100,100)	RUILOS	166
	IF(MOD(INFO(IORD)/100,100).NE.NWARE) GO TO 200	RUILOS	167
187	IPLTS=MOD(INFO(IORD),100)	RUILOS	168
	GO TO (2001,2002,2003).ISPLIT	RUILOS	169
2001	IF(IPLTS.EQ.ICAP) GO TO 300	RUILOS	170
170	GO TO 200	RUILOS	171
2002	IF(IPLTS.GT.ICAP) GO TO 200	RUILOS	172

```

        GO TO 300
2003 IF (IPLTS.LE.IRCAP) GO TO 301
      IF (NEXTA.NE.LASTA) GO TO 305
175 IF (NDEST.EQ.0) NDEST=MOD(INFO(IORDR)/100,100)
      IF (NDEST.NE.MOD(INFO(IORDR)/100,100)) GO TO 200
305 ISAVE=IPLTS-IRCAP
      C FORM NEW INFO/ORDER ELEMENT
      INFO(ICORD)=INFO(IORDR)/100*100+ISAVE
180 NORDR=NORDR+1
      INFO(NCORD)=-(INFO(IORDR)/100*100+IRCAP)
      ONUMH(IORDR)=-IABS(ONJMB(IORDR))
      IAREA(NORDR)=IAREA(IORDR)
      ONUMH(NORDR)=ONJMB(IORDR)
185 NPALTS=NPALTS+IRCAP
      C COMPUTE THE TIME REMAINING FOR VEHICLE
      TLEFT(ITRUCK)=TLEFT(ITRUCK)-(TIME+2.*(LTIME(ITRUCK)*FLOAT(IT?CAP)
190 1+STIME(ITRUCK)))
      IRCAP=0
      C LINK SEGMENT TO SCHEDULE
      LSTORD=NORDR
      IF (PTSOR(ITRUCK).LE.0) PTSOR(ITRUCK)=NORDR
      LSTART=PTSOR(ITRUCK)
195 ISAVE=LSTART
      LINK=LFRWD(LSTART)
325 IF (LINK.LE.0) GO TO 350
      ISAVE=LINK
      LINK=LFRWD(LINK)
      GO TO 325
200 350 IF (ISAVE.EQ.NORDR) GO TO 360
      LFRWD(ISAVE)=NORDR
      LBKWD(NORDR)=ISAVE
360 IFILL=IFILL+1
      IDESTP(NORDR)=IFILL
205 IF (IFILL.EQ.1) GO TO 370
      TLEFT(ITRUCK)=TLEFT(ITRUCK)+TIME
370 IDEST=MOD(IABS(INFO(NORDR)/100,100)
      IF (JSAVE.EQ.0) JSAVE=IDEST
210 IF (JSAVE.NE.IDEST) TLEFT(ITRUCK)=TLEFT(ITRUCK)-2.
      LASTA=NEXTA
      LASTW=IDEST
      RETURN
300 IF (NEXTA.NE.LASTA) GO TO 175
      IF (NDEST.EQ.0) NDEST=MOD(INFO(IORDR)/100,100)
215 IF (NDEST.NE.MOD(INFO(IORDR)/100,100)) GO TO 200
375 IFILL=IFILL+1
      NPALTS=NPALTS+IPLTS
      IRCAP=IRCAP-IPLTS
      IDEST=MOD(INFO(IORDR)/100,100)
220 C COMPUTE TIME REMAINING FOR VEHICLE
      TLEFT(ITRUCK)=TLEFT(ITRUCK)-(TIME+2.*(STIME(ITRUCK)+LTIME(ITRUCK)
      1*FLOAT(IPLTS)))
      IF (JSAVE.EQ.0) JSAVE=IDEST
225 IF (JSAVE.NE.IDEST) TLEFT(ITRUCK)=TLEFT(ITRUCK)-2.
      JSAVE=IDEST
      IF (IFILL.EQ.1) GO TO 340
      TLEFT(ITRUCK)=TLEFT(ITRUCK)+TIME
      C LINK SEGMENT TO SCHEDULE

```

	300	INFO(IORDR)=-INFO(IORDR)	RUTLOS	230
230		IDESTP(IORDR)=IFILL	RUTLOS	231
		LSTO=IORDR	RUTLOS	232
		IF(PTSOR(ITRUCK).LE.0) PTSOR(ITRUCK)=IORDR	RUTLOS	233
		LSTART=PTSOR(ITRUCK)	RUTLOS	234
		ISAVE=LSTART	RUTLOS	235
235		LINK=LFRWD(LSTART)	RUTLOS	236
	410	IF(LINK.LE.0) GO TO 400	RUTLOS	237
		ISAVE=LINK	RUTLOS	238
		LINK=LFRWD(LINK)	RUTLOS	239
		GO TO 410	RUTLOS	240
240	400	IF(ISAVE.EQ.IORDR) GO TO 420	RUTLOS	241
		LFRWD(ISAVE)=IORDR	RUTLOS	242
		LFRWD(IORDR)=ISAVE	RUTLOS	243
	420	IF(IACAP.LE.0) GO TO 421	RUTLOS	244
		IF(TLEFT(ITRUCK).LE.0.0) GO TO 421	RUTLOS	245
245		GO TO 200	RUTLOS	246
	421	LASTA=NEXTA	RUTLOS	247
		LASTW=IDEST	RUTLOS	248
		RETURN	RUTLOS	249
	200	CONTINUE	RUTLOS	250
250	2000	CONTINUE	RUTLOS	251
		NWARE=0	RUTLOS	252
		NDEST=0	RUTLOS	253
		DO 5000 ISPLIT=1,MSPLIT	RUTLOS	254
		IF(SPLTSM(ISPLIT).LT.MINLO(ITYPE)) GO TO 5000	RUTLOS	255
255		DO 500 I=1,NSLG	RUTLOS	256
		IORDR=SORDR(I)	RUTLOS	257
		IF(IORDR.EQ.0) GO TO 500	RUTLOS	258
		IF(INFC(IORDR).LE.J) GO TO 500	RUTLOS	259
	C	ENOUGH PALLETS	RUTLOS	260
260		IF(NEXTA.NE.LASTA) GO TO 505	RUTLOS	261
		IF(NWARE.EQ.0) NWARE=MOD(INFO(IORDR)/100,100)	RUTLOS	262
		IF(NWARE.NE.MOD(INFO(IORDR)/100,100)) GO TO 500	RUTLOS	263
	505	IPLTS=MOD(INFO(IORDR),100)	RUTLOS	264
		GO TO (5001,5002,5003),ISPLIT	RUTLOS	265
265	5001	IF(IPLTS.EQ.IRCAP) GO TO 600	RUTLOS	266
		GO TO 500	RUTLOS	267
	5002	IF(IPLTS.GT.IRCAP) GO TO 500	RUTLOS	268
		GO TO 600	RUTLOS	269
	5003	IF(IPLTS.LE.IRCAP) GO TO 500	RUTLOS	270
270		IF(NEXTA.NE.LASTA) GO TO 605	RUTLOS	271
		IF(NDEST.EQ.0) NDEST=MOD(INFO(IORDR)/100,100)	RUTLOS	272
		IF(NDEST.NE.MOD(INFO(IORDR)/100,100)) GO TO 500	RUTLOS	273
	605	ISAVE=IPLTS-IRCAP	RUTLOS	274
	C	FORM NEW INFO/ORDER ELEMENTS	RUTLOS	275
275		INFO(ICRDR)=INFO(IORDR)/100*100+ISAVE	RUTLOS	276
		NORDR=NORDR+1	RUTLOS	277
		INFO(NCRDR)=-(INFO(IORDR)/100*100+IRCAP)	RUTLOS	278
		ONUMB(IORDR)=IABS(ONUMB(IORDR))	RUTLOS	279
		ONUMB(NORDR)=ONUMB(IORDR)	RUTLOS	280
280		IAREA(NORDR)=IAREA(IORDR)	RUTLOS	281
		NPALTS=NPALTS+IRCAP	RUTLOS	282
	C	COMPUTE THE TIME REMAINING FOR VEHICLE	RUTLOS	283
		TLEFT(ITRUCK)=TLEFT(ITRUCK)-(TIME+2.*(LTIME(ITRUCK)*FLOAT(IPCAP)	RUTLOS	284
		+STIME(ITRUCK)))	RUTLOS	285
285		IRCAP=0	RUTLOS	286

	C	LINK SEGMENT TO SCHEDULE	RUILOS	287
		LSTORQ=NORDR	RUILOS	288
		IF (PTSOR(I TRUCK).LE.0) PTSOR(I TRUCK)=NORDR	RUILOS	289
290		LSTART=PTSOR(I TRUCK)	RUILOS	290
		ISAVE=LSTART	RUILOS	291
		LINK=LFRWD(LSTART)	RUILOS	292
	625	IF (LINK.LE.0) GO TO 650	RUILOS	293
		ISAVE=LINK	RUILOS	294
295		LINK=LFRWD(LINK)	RUILOS	295
		GO TO 625	RUILOS	296
	650	IF (ISAVE.EQ.NORDR) GO TO 660	RUILOS	297
		LFRWD(ISAVE)=NORDR	RUILOS	298
		LBRWD(NORDR)=ISAVE	RUILOS	299
	660	IF ILL=IFILL+1	RUILOS	300
300		IJESTP(NORDR)=I ILL	RUILOS	301
		IF (IFILL.EQ.1) GO TO 670	RUILOS	302
		TLEFT(I TRUCK)=TLEFT(I TRUCK)+TIME	RUILOS	303
	670	IDEST=MOD(ABS(INFO(NORDR)/100,100)	RUILOS	304
		IF (JSAVE.EQ.0) JSAVE=IDEST	RUILOS	305
305		IF (JSAVE.NE.IDEST) TLEFT(I TRUCK)=TLEFT(I TRUCK)-2.	RUILOS	306
		LASTA=NEXTA	RUILOS	307
		LASTW=IDEST	RUILOS	308
		RETURN	RUILOS	309
	600	IF (NEXTA.NE.LASTA) GO TO 675	RUILOS	310
310		IF (IDEST.EQ.0) IDEST=MOD(INFO(IORDR)/100,100)	RUILOS	311
		IF (IDEST.NE.MOD(INFO(IORDR)/100,100)) GO TO 600	RUILOS	312
	675	IF ILL=IFILL+1	RUILOS	313
		NPALTS=NPALTS+IPLTS	RUILOS	314
		IRCAP=IRCAP-IPLTS	RUILOS	315
315		IDEST=MOD(INFO(IORDR)/100,100)	RUILOS	316
	C	COMPUTE TIME REMAINING FOR VEHICLE	RUILOS	317
		TLEFT(I TRUCK)=TLEFT(I TRUCK)-(TIME+2.*(STIME(I TRUCK)+LTIME(I TRUCK)	RUILOS	318
		1 *FLOAT(IPLTS)))	RUILOS	319
		IF (JSAVE.EQ.0) JSAVE=IDEST	RUILOS	320
320		IF (JSAVE.NE.IDEST) TLEFT(I TRUCK)=TLEFT(I TRUCK)-2.	RUILOS	321
		JSAVE=IDEST	RUILOS	322
		IF (IFILL.EQ.1) GO TO 680	RUILOS	323
		TLEFT(I TRUCK)=TLEFT(I TRUCK)+TIME	RUILOS	324
	C	LINK SEGMENT TO SCHEDULE	RUILOS	325
325	680	INFO(IORDR)=-INFO(IORDR)	RUILOS	326
		IJESTP(IORDR)=IFILL	RUILOS	327
		LSTORQ=IORDR	RUILOS	328
		IF (PTSOR(I TRUCK).LE.0) PTSOR(I TRUCK)=IORDR	RUILOS	329
		LSTART=PTSOR(I TRUCK)	RUILOS	330
330		ISAVE=LSTART	RUILOS	331
		LINK=LFRWD(LSTART)	RUILOS	332
	710	IF (LINK.LE.0) GO TO 700	RUILOS	333
		ISAVE=LINK	RUILOS	334
335		LINK=LFRWD(LINK)	RUILOS	335
		GO TO 710	RUILOS	336
	700	IF (ISAVE.EQ.IORDR) GO TO 720	RUILOS	337
		LFRWD(ISAVE)=IORDR	RUILOS	338
		LBRWD(IORDR)=ISAVE	RUILOS	339
340	720	IF (IRCAP.LE.0) GO TO 721	RUILOS	340
		IF (TLEFT(I TRUCK).LE.0) GO TO 721	RUILOS	341
		GO TO 500	RUILOS	342
	721	LASTA=NEXTA	RUILOS	343
		LASTW=IDEST	RUILOS	344
		RETURN	RUILOS	345
345	500	CONTINUE	RUILOS	346
	5000	CONTINUE	RUILOS	347
		IF (NPALTS.LE.0) RETURN	RUILOS	348
		LASTA=NEXTA	RUILOS	349
		LASTW=IDEST	RUILOS	350
350		RETURN	RUILOS	351
		END	RUILOS	352

ROUTINE: MATCH

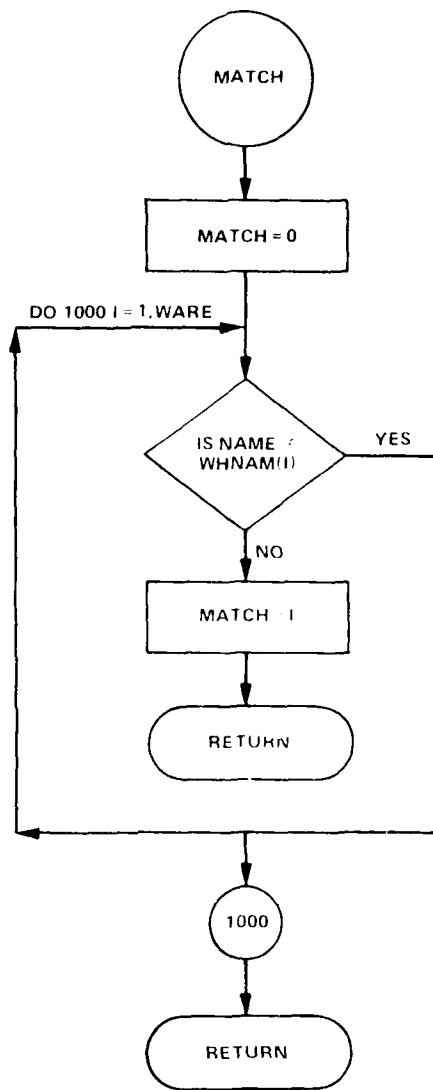
ARGUMENTS:

NAME - Alphanumeric warehouse name

IAREA - Area indicator of "NAME" warehouse

Description:

MATCH searches the warehouse name list for a match for NAME and returns its warehouse and area numbers.



FUNCTION NOM

74/74 OPT=0 ROUND=*/ TRACE

FTN 4.6+460

10/16/80 08.34.52

1	FUNCTION NOM(ITYPE,TO,IT)	AVS2 2
	ALPHA IO,IT	AVS2 3
	NOM=0	AVS2 4
	GO TO (100,200,300,400),ITYPE	AVS2 5
5	100 IF(IO.EQ.3H193.OR.IT.EQ.3H193) GO TO 9999	AVS2 6
	IF(IO.EQ.4H1172.OR.IT.EQ.4H1172) GO TO 9999	AVS2 7
	IF(IO.EQ.4H1605.OR.IT.EQ.4H1605) GO TO 9999	AVS2 8
	RETURN	AVS2 9
10	200 IF(TO.EQ.3H193.OR.IT.EQ.3H193) GO TO 9999	AVS2 10
	RETURN	AVS2 11
15	300 RETURN	AVS2 12
	400 IF(TO.EQ.4H1172.OR.IT.EQ.4H1172) GO TO 9999	AVS2 13
	RETURN	AVS2 14
20	9999 NOM=1	AVS2 15
	RETURN	AVS2 16
	END	AVS2 17

1	FUNCTION MATCH(NAME,IAREA)	MATCH 2
	C	MATCH 3
	C	MATCH 4
	C	MATCH 5
5	C	MATCH 6
	C	MATCH 7
	C	MATCH 8
	*** COMMONS ***	MATCH 9
	ALPHA WHNAM,NAME	MATCH 10
10	COMMON/WHINTG/ NWARE,NAREA(16)	MATCH 11
	COMMON/WHOUSE/ WHNAM(92)	MATCH 12
	C	MATCH 13
	AREA=0	MATCH 14
	MATCH = 0	MATCH 15
	DO 1000 I = 1, NWARE	MATCH 16
15	IF(NAME.EQ.WHNAM(I)) GO TO 2000	MATCH 17
	1000 CONTINUE	MATCH 18
	RETURN	MATCH 19
	2000 MATCH=I	MATCH 20
	I=I	MATCH 21
20	ICLK=0	MATCH 22
	DO 3000 J=1,16	MATCH 23
	ICLK=ICLK+NAREA(J)	MATCH 24
	IF(I.LE.ICLK) GO TO 4000	MATCH 25
	3000 CONTINUE	MATCH 26
	RETURN	MATCH 27
25	4000 IAREA=J	MATCH 28
	RETURN	MATCH 29
	END	

ROUTINE: ROUTE

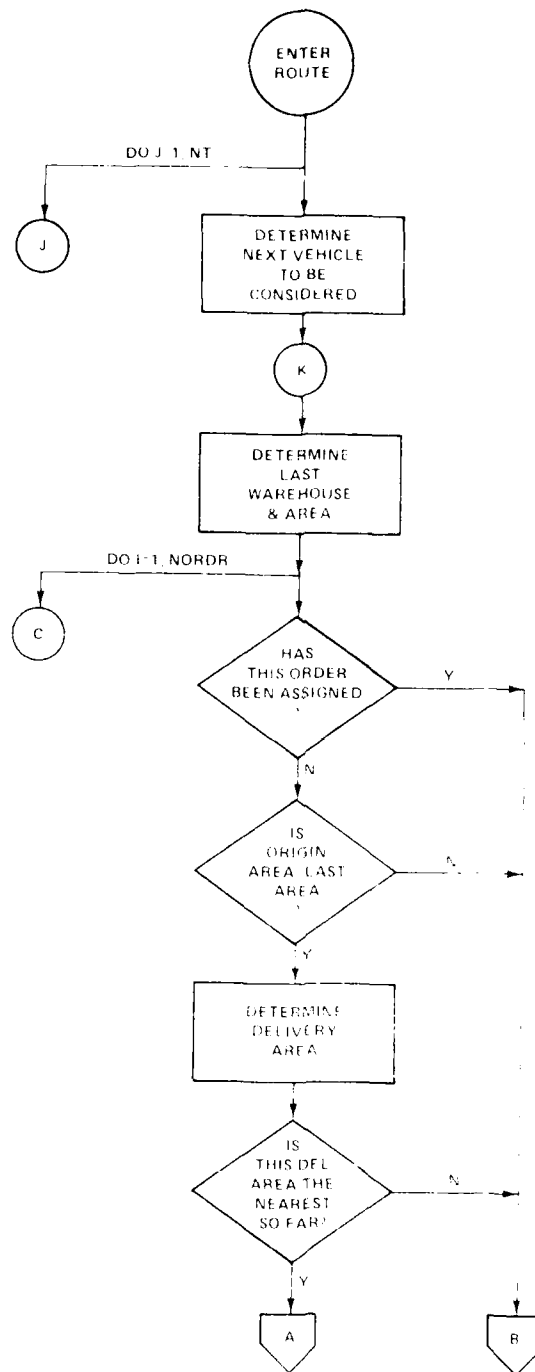
ARGUMENT: IAVS - Run option indicator

1, AVS1

2. AVS2

Description:

ROUTE applies the algorithm used in assigning orders to vehicles. It determines the next area of delivery for each available vehicle by searching the available orders, noting transfer and cargo movement times. The next area to be encountered by a vehicle is determined by least travel time for which a minimum quantity of cargo is to be moved. Other restrictions, such as the accessibility of the area to the vehicle and the order of cargo delivery, are also imposed on the selection of the next area. Each area selected may be an order origin area, destination area, or both. Once origin and destination areas are known, ROUTE calls BLDTR (first on, last off) or BUILDS (first on, last of.) to assign orders to the vehicle. The current area is updated and the next area selection process is repeated until the vehicle is out of time or the quantity of unassigned cargo does not meet delivery requirements.



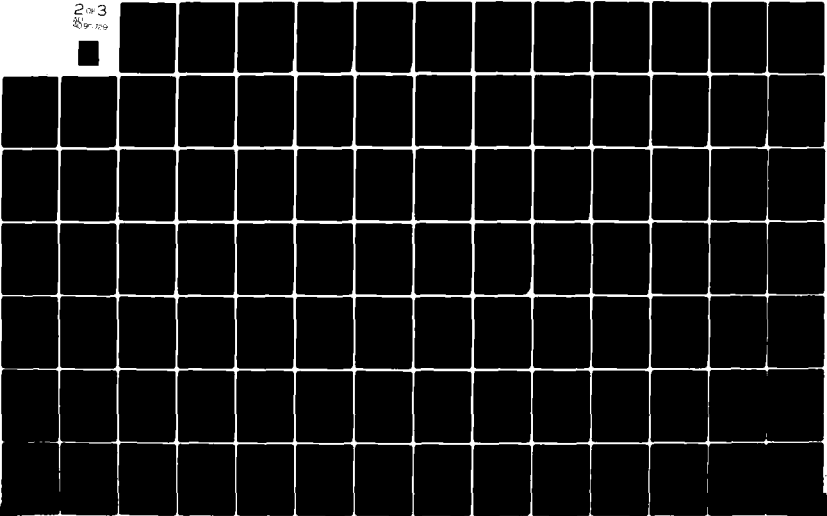
AD-A095 729

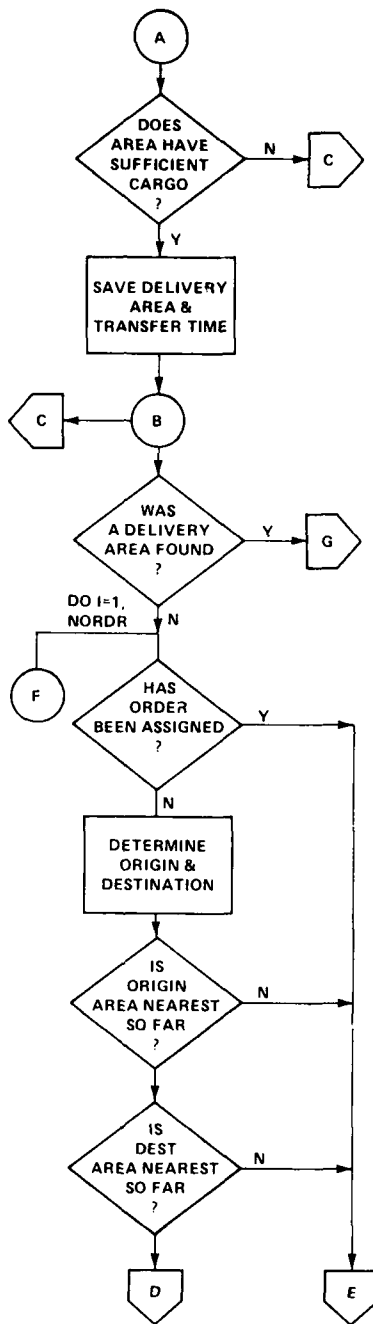
DAVID W TAYLOR NAVAL SHIP RESEARCH AND DEVELOPMENT CE--ETC F/G 12/2
AUTOMATIC VEHICLE SCHEDULING (AVS) PROGRAMMER'S INSTRUCTION MAN--ETC(U)
FEB 81 R WINCHELL, R MELTON, M NATRELLA
DTNSRDC-81/017

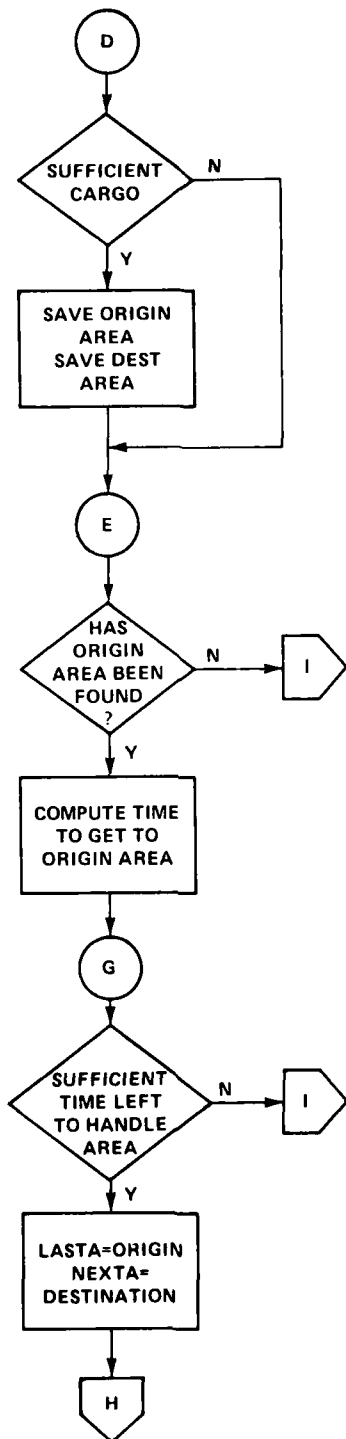
UNCLASSIFIED

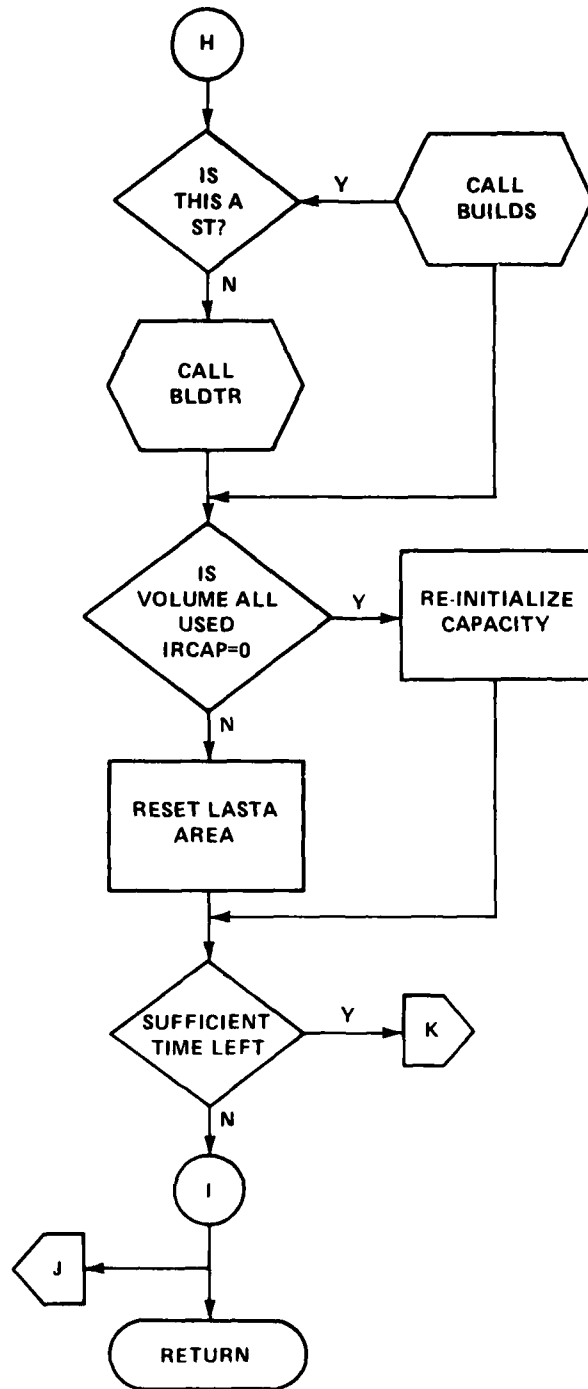
NL

2 of 3
97-219









```

1      SUBROUTINE ROUTE (IAVS)                                ROUTE  2
C .....                                                    ROUTE  3
C      ROUTE ASSIGNS ORDERS TO AVAILABLE VEHICLES           ROUTE  4
C      - COMMON AREA, ORIGIN AND DESTINATIONS              ROUTE  5
5      C      - NEAREST AREA TO LAST WAREHOUSE SERVICED    ROUTE  6
C .....                                                    ROUTE  7
      INTEGER LSTSV(16,16)                                  ROUTE  8
      INTEGER PTRK,ONUMB                                    ROUTE  9
      INTEGER PTSOR,CAPAC,TRUCK,DATE                       ROUTE 10
10     INTEGER TYPORD(4)                                    ROUTE 11
      ALPHA WWHNAM                                         ROUTE 12
      REAL LTIME                                           ROUTE 13
      COMMON                                               ROUTE 14
      * /GEN/ RTTIM,MINLD(4)                                ROUTE 15
15     * /WHINTG/ NWARE,NAREA(16)                          ROUTE 16
      * /SCHFDL/ ONUMB(20),INF(200),IDESTP(200),IAREA(200),LFR4D(200),
      * /LCKWD(200),LCFWD(200),LCBWD(200),ATIME(200),STOPF(200)
      * /TRUCKS/ PTSOR(50),TRUCK(50),CAPAC(50),STIME(50),LTIME(50),
20     * /RTIM(50),LEFT(50),NT
      * /WHOUSE/ WWHNAM(92)                                ROUTE 21
      * /TINTAB/ TTIME(3,45),TTIM(16)                     ROUTE 22
      * /MSCLNS/ NORDR,NTRKS(4),PTRK(4)                   ROUTE 23
      * /IAOPT/ SHIFT,DATE                                  ROUTE 24
25     * /BLOS/ LASTW,LASTA,NEXTA,ITRUCK,ITYPE,IPASS,NPALTS,TIME
      * /SPORD/ ICNTRL,SAVTIM,ISORD                        ROUTE 26
      * /SPLIT/ NSPLIT                                     ROUTE 27
      * /LOADSV/ LOAD                                       ROUTE 28
      DATA TYPORD/1,2,3,4/                               ROUTE 29
      RESET=0                                              ROUTE 30
30     C .....
      C CONSIDER NEXT AVAILABLE VEHICLE                     ROUTE 31
      C TYPE = 1, STRADDLES                                ROUTE 32
      C TYPE = 2, TRANSPORTERS                             ROUTE 33
      C TYPE = 3, TRACTOR TRAILERS                        ROUTE 34
35     C TYPE = 4, INDUSTRIAL TRACTOR                      ROUTE 35
      C .....
      C TOTAL NUMBER OF PALLETS TO BE MOVED IN SHIFT      ROUTE 36
      C .....
      NSPLIT=1                                             ROUTE 37
      ISPASS=0                                             ROUTE 38
40     IF (IAVS.EQ.1) ISPASS=2                             ROUTE 39
      LPA=2                                               ROUTE 40
      ISS=SHIFT                                            ROUTE 41
      XSHIFT=(ISS/100)*60+MOD(ISS,100)                   ROUTE 42
45     ISS=SAVTIM                                          ROUTE 43
      TME=(ISS/100)*60+MOD(ISS,100)                       ROUTE 44
      TME=TME-XSHIFT                                      ROUTE 45
      TME=RTTIM-TME                                       ROUTE 46
      NOFF=0                                              ROUTE 47
50     NJ 25 I=1,NOROR                                     ROUTE 48
      IF (INFC(I).LE.6) GO TO 25                          ROUTE 49
      IF (MOD(IAREA(I),100).GT.10) NOFF=NOFF+MOD(INFC(I),100)
      25 CONTINUE                                         ROUTE 50
55     ITYPE=0                                             ROUTE 51
      ICNT=0                                              ROUTE 52
100    ICNT=ICNT+1                                        ROUTE 53
      IF (ICNT.GT.4) GO TO 9000                           ROUTE 54

```

	ITYPE=TYPORD(ICNT)	ROUTE	59
	IF(ITYPE.GT.2.AND.ICNTR.LV.E.0) GO TO 100	ROUTE	60
60	MTRKS=NTRKS(ITYPE)	ROUTE	61
	IF(MTRKS.LE.0) GO TO 100	ROUTE	62
	DO 80 JJ=1,16	ROUTE	63
	DO 80 MM=1,16	ROUTE	64
80	LSTSV(JJ,MM)=0	ROUTE	65
65	DO 200 L=1,MTRKS	ROUTE	66
	ITRUCK=PTRK(ITYPE) *L	ROUTE	67
	LOCFF=0	ROUTE	68
	LOAD=0	ROUTE	69
	IFILL=0	ROUTE	70
70	IRCAP=CAPAC(ITRUCK)	ROUTE	71
	DETERMINE LAST WAREHOUSE OF VEHICLE	ROUTE	72
	LASTW=23	ROUTE	73
	LASTA=2	ROUTE	74
	IF(ICNTRL.NE.1) GO TO 140	ROUTE	75
75	IF(NSPLIT.EQ.1) GO TO 150	ROUTE	76
180	IF(PTSOR(ITRUCK).LE.0) GO TO 150	ROUTE	77
	LINK=PTSOR(ITRUCK)	ROUTE	78
	LASTA=MOD(IAREA(LINK),100)	ROUTE	79
	LASTW=MOD(IABS(INFO(LINK))/100,100)	ROUTE	80
80	185 LINKS=LFRWD(LINK)	ROUTE	81
	IF(LINKS.LE.0) GO TO 155	ROUTE	82
	IF(ITYPE.NE.1.AND.IDESTP(LINKS).GT.IDESTP(LINK)) GO TO 175	ROUTE	83
	IF(ITYPE.EQ.1.AND.IDESTP(LINKS).LT.IDESTP(LINK)) GO TO 175	ROUTE	84
	LASTA=MOD(IAREA(LINKS),100)	ROUTE	85
85	LASTW=MOD(IABS(INFO(LINKS))/100,100)	ROUTE	86
	175 LINK=LINKS	ROUTE	87
	GO TO 185	ROUTE	88
	155 IF(TLEFT(ITRUCK)-T(LASTA,LPA,ITYPE).LE.THE) GO TO 153	ROUTE	89
	TLEFT(ITRUCK)=THE	ROUTE	90
90	LASTA=2	ROUTE	91
	LASTW=23	ROUTE	92
	C DETERMINE NEAREST WAREHOUSE WITH UNFILLED ORDER	ROUTE	93
	C ORIGIN AREA = LASTA	ROUTE	94
95	C CHECK IF ANY ORDER ORIGINATES IN LAST AREA	ROUTE	95
	C	ROUTE	96
	C IPASS = 1 BUILD NON-DEPENDENT ORDERS	ROUTE	97
	C IPASS = 2 BUILD DEPENDENT ORDERS	ROUTE	98
	C	ROUTE	99
100	150 IPASS=IPASS	ROUTE	100
	NOELC=0	ROUTE	101
	IF(LOAD.LE.0) GO TO 300	ROUTE	102
311	DO 160 KK=1,16	ROUTE	103
	DO 160 MM=1,16	ROUTE	104
160	LSTSV(KK,MM)=0	ROUTE	105
105	300 IPASS=IPASS+1	ROUTE	106
	312 IF(IPASS.GT.4) GO TO 200	ROUTE	107
	307 NEXTA=0	ROUTE	108
	TIME=1000.0	ROUTE	109
	ISTART=1	ROUTE	110
110	110 IEND=NORJR	ROUTE	111
	C SEARCH TO FIND NEAREST DESTINATION AREA TO LASTA	ROUTE	112
302	DO 250 I=ISTART,IEND	ROUTE	113
	IF(IPASS.GT.2) GO TO 303	ROUTE	114
	IF(OMJH(I).GE.0) GO TO 250	ROUTE	115

115	IF (ONUMB(I).GT.(-100)) GO TO 250	ROUTE	115
303	IF (LASTA.NE.IAREA(I)/100) GO TO 250	ROUTE	117
	NXT=MOD(IAREA(I),100)	ROUTE	118
	IF (LSTSV(LASTA,NXT).NE.0) GO TO 250	ROUTE	119
	IF (INFO(I).LE.0) GO TO 251	ROUTE	120
120	IO=INFO(I)/10000	ROUTE	121
	IT=MOD(INFO(I)/100,100)	ROUTE	122
	IF (NOM(ITYPE,MHNA(I),4MNA(IT)).EQ.1) GO TO 251	ROUTE	123
	IDEST=MOD(IAREA(I),100)	ROUTE	124
	IF (ITYPE.NE.3) GO TO 305	ROUTE	125
125	IF (NOFF+LOFF.LT.MIN(3)) GO TO 310	ROUTE	126
	IF (NOFF+LOFF.LT.MIN(3)) GO TO 310	ROUTE	127
	IF (IDEST.LE.10) GO TO 251	ROUTE	128
	GO TO 305	ROUTE	129
310	IF (IDEST.GT.10) GO TO 250	ROUTE	130
130	308 IF (IFILL.EQ.0) GO TO 304	ROUTE	131
	IF (Y(IDEST,NSAVE,ITYPE).GT.10.0) GO TO 250	ROUTE	132
	304 CONTINUE	ROUTE	133
	TTEMP=T(LASTA,IDEST,ITYPE)	ROUTE	134
	IF (MOD(IPASS,2).EQ.0) GO TO 251	ROUTE	135
135	JSTART=1	ROUTE	136
	JEND=NOROR	ROUTE	137
	TTEMPS=1000.0	ROUTE	138
305	DO 220 II=JSTART,JEND	ROUTE	139
	IF (IPASS.GT.2) GO TO 305	ROUTE	140
140	IF (ONUMB(II).GE.0) GO TO 220	ROUTE	141
	IF (ONUMB(II).GT.(-100)) GO TO 220	ROUTE	142
306	IF (INFO(II).LE.0) GO TO 220	ROUTE	143
	IF (IAREA(II)/100.NE.IDEST) GO TO 220	ROUTE	144
	IO=INFO(II)/10000	ROUTE	145
145	IT=MOD(INFO(II)/100,100)	ROUTE	146
	IF (NOM(ITYPE,MHNA(I),4MNA(IT)).EQ.1) GO TO 220	ROUTE	147
	TTEMP1=T(IAREA(II)/100,40)(IAREA(II),100,ITYPE)	ROUTE	148
	IF (TTEMPS.GT.TTEMP1) TTEMPS=TTEMP1	ROUTE	149
220	CONTINUE	ROUTE	150
150	TTEMP=TTEMP+TTEMPS	ROUTE	151
251	IF (TTEMP.GE.TIME) GO TO 250	ROUTE	152
	NEXTA=IDEST	ROUTE	153
	TIME=TTEMP	ROUTE	154
250	CONTINUE	ROUTE	155
155	C TRANSFER TO BUILD NEXT SCHEDULE SEGMENT	ROUTE	156
	IF (NEXTA.LE.0) GO TO 210	ROUTE	157
	C	ROUTE	158
	C DETERMINE PALLETS TO BE MOVED FROM LASTA TO NEXTA	ROUTE	159
	C	ROUTE	160
160	ICHECK=LASTA*100+NEXTA	ROUTE	161
	NSAMF=0	ROUTE	162
	NPALTA=0	ROUTE	163
	DO 252 JJ=1,NOROR	ROUTE	164
	IF (INFO(JJ).LE.0) GO TO 252	ROUTE	165
165	IF (LASTA.NE.IAREA(JJ)/100) GO TO 252	ROUTE	166
	IO=INFO(JJ)/10000	ROUTE	167
	IT=MOD(INFO(JJ)/100,100)	ROUTE	168
	IF (NOM(ITYPE,MHNA(I),4MNA(IT)).EQ.1) GO TO 252	ROUTE	169
	IF (NSPLIT.EQ.1.AND.MOD(INFO(JJ),100).EQ.IRCA) GO TO 255	ROUTE	170
170	IF (NSPLIT.EQ.2.AND.MOD(INFO(JJ),100).LE.IRCA) GO TO 255	ROUTE	171
	IF (NSPLIT.EQ.3) GO TO 255	ROUTE	172

	GO TO 252	ROUTE	173
	255 IF (IAREA(JJ).NE.ICHECK) GO TO 254	ROUTE	174
	NPALTA=NPALTA+MOD(INFO(IJ),100)	ROUTE	175
175	GO TO 252	ROUTE	176
	254 IF (I(LASTA,MOD(IAREA(JJ),100),ITYPE).LE.10.0)	ROUTE	177
	NSAME=NSAME+MOD(INFO(IJ),100)	ROUTE	178
	252 CONTINUE	ROUTE	179
	NPALTA=NPALTA+LOAD	ROUTE	180
180	IF (NDEL.C.NE.0) NPALTA=0	ROUTE	181
	NDEL=0	ROUTE	182
	NSAME=NSAME+LOAD	ROUTE	183
	C	ROUTE	184
	C DETERMINE MIN LOAD FOR EACH TYPE	ROUTE	185
	NSAME=NSAME+NPALTA	ROUTE	186
185	GO TO (221,222,223,222),ITYPE	ROUTE	187
	221 IF (NPALTA.LT.MINLD(ITYPE)) GO TO 253	ROUTE	188
	GO TO 400	ROUTE	189
	222 IF (NPALTA.GE.MINLD(ITYPE)) GO TO 400	ROUTE	190
190	IF (NSAME.LT.MINLD(ITYPE)) GO TO 253	ROUTE	191
	GO TO 400	ROUTE	192
	223 IF (NOFF+LOFF.GE.MINLD(ITYPE)) GO TO 400	ROUTE	193
	IF (NSAME.GE.MINLD(ITYPE)) GO TO 400	ROUTE	194
	IF (NPALTA.GE.MINLD(ITYPE)) GO TO 400	ROUTE	195
195	C TRY TO FIND NEXT AREA WITH MIN LOAD TO BE MOVED	ROUTE	196
	253 LSTSV(LASTA,NEXTA)=LASTA	ROUTE	197
	GO TO 307	ROUTE	198
	C	ROUTE	199
	C DETERMINE NEAREST WAREHOUSE AREA TO LASTA	ROUTE	200
	SEARCH AREAS TO FIND NEAREST AREA TO LASTA	ROUTE	201
200	210 IF (ITYPE.NE.1) GO TO 212	ROUTE	202
	IF (IFILL.EQ.0) GO TO 212	ROUTE	203
	LASTA=NSAVE	ROUTE	204
	GO TO 4050	ROUTE	205
205	212 LSTA=0	ROUTE	206
	LSTW=0	ROUTE	207
	NEXTA=0	ROUTE	208
	TIMES=1000.0	ROUTE	209
	DO 260 I=1,16	ROUTE	210
	IF (I.EQ.LASTA) GO TO 260	ROUTE	211
210	ISTART=1	ROUTE	212
	IEND=NORDR	ROUTE	213
	TEMP1=T(LASTA,I,ITYPE)	ROUTE	214
	TEMP2=1000.0	ROUTE	215
	DO 265 II=ISTART,IEND	ROUTE	216
215	IF (IPASS.GT.2) GO TO 309	ROUTE	217
	IF (ONUMR(II).GE.0) GO TO 265	ROUTE	218
	IF (ONUMR(II).GT.(-100)) GO TO 265	ROUTE	219
	309 IF (INFO(II).LE.0) GO TO 265	ROUTE	220
	IF (I.NE.IAREA(II)/100) GO TO 265	ROUTE	221
220	ID=INFO(II)/10000	ROUTE	222
	IT=MOD(INFO(II)/100,100)	ROUTE	223
	IF (NOW(ITYPE,WHNAME(I),WHNAME(II)).EQ.1) GO TO 265	ROUTE	224
	IDEST=MOD(IAREA(II),100)	ROUTE	225
	IF (I,IDEST,ITYPE).GE.1000.0) GO TO 265	ROUTE	226
225	IF (IFILL.EQ.0) GO TO 274	ROUTE	227
	IF (I,IDEST,NSAVE,ITYPE).GT.10.0) GO TO 265	ROUTE	228
	274 IF (MOD(IPASS,2).EQ.0) GO TO 270	ROUTE	229
	DO 275 IIT=ISTART,IEND	ROUTE	230

	IF (INFO(III).LE.0) GO TO 275	ROUTE	231
230	IF (IPASS.GT.2) GO TO 276	ROUTE	231
	IF (ONUMR(III).GE.0) GO TO 275	ROUTE	232
	276 IF (IDEST.NE.IAREA(III)/101) GO TO 275	ROUTE	233
	GO TO 270	ROUTE	234
	275 CONTINUE	ROUTE	235
235	GO TO 260	ROUTE	236
	270 IF (TEMP2.LE.T(I.IDEST.ITYPE)) GO TO 265	ROUTE	237
C	JCES NEXT AREA HAVE MIN LOAD FOR THIS TYPE	ROUTE	238
	IF (ITYPE.NE.3) GO TO 205	ROUTE	239
	IF (NOFF*LOADOFF.LT.MINLOAD) GO TO 205	ROUTE	240
240	IF (IDEST.LE.10) GO TO 255	ROUTE	241
	GO TO 205	ROUTE	242
	205 IF (IDEST.GT.10) GO TO 255	ROUTE	243
	206 NSAVE=MOD(IAREA(II),100)	ROUTE	244
	IF (LSTSVI.NSVE).NE.0) GO TO 265	ROUTE	245
245	TEMP2=T(I.IDEST.ITYPE)	ROUTE	246
	LSAVEW=INFO(II)/10000	ROUTE	247
	NSAVEA=MOD(IAREA(II),101)	ROUTE	248
	265 CONTINUE	ROUTE	249
	TIME=TEMP1+TEMP2	ROUTE	250
250	IF (TIME.GE.TIMES) GO TO 250	ROUTE	251
	TIMES=TIME	ROUTE	252
	LSTA=LSAVEW	ROUTE	253
	NEXTA=NSAVEA	ROUTE	254
	LSTA=I	ROUTE	255
255	260 CONTINUE	ROUTE	256
	IF (TIMES.GE.1000.0) GO TO 300	ROUTE	257
	IF (LSTA.LE.0) GO TO 300	ROUTE	258
	NSAVE=0	ROUTE	259
	NPALTA=0	ROUTE	260
260	ICHECK=LSTA*100+NEXTA	ROUTE	261
	DO 261 JJ=1,NOROR	ROUTE	262
	IF (INFO(JJ).LE.0) GO TO 261	ROUTE	263
	IF (NSPLIT.EQ.1.AND.MOD(INFO(JJ),100).EQ.IRCAP) GO TO 263	ROUTE	264
	IF (NSPLIT.EQ.2.AND.MOD(INFO(JJ),100).LE.IRCAP) GO TO 263	ROUTE	265
265	IF (NSPLIT.EQ.3) GO TO 253	ROUTE	265
	GO TO 261	ROUTE	267
	263 IF (LSTA.NE.IAREA(JJ)/101) GO TO 261	ROUTE	268
	IQ=INFO(JJ)/10000	ROUTE	269
	IT=MOD(INFO(JJ)/100,100)	ROUTE	270
270	IF (NOW(ITYPE).WHNAM(IQ).WHNAM(IT)).EQ.1) GO TO 261	ROUTE	271
	IF (ICHECK.NE.IAREA(JJ)) GO TO 264	ROUTE	272
	NPALTA=NPALTA+MOD(INFO(JJ),100)	ROUTE	273
	GO TO 261	ROUTE	274
	264 IF ((LSTA+MOD(IAREA(JJ),100).ITYPE).LE.10.0)	ROUTE	275
275	* NSAVE=NSAVE+MOD(INFO(JJ),100)	ROUTE	276
	261 CONTINUE	ROUTE	277
C	DETERMINE MIN LOAD FOR EACH TYPE	ROUTE	278
	NSAME=NSAME+LOAD	ROUTE	279
	NPALTA=NPALTA+LOAD	ROUTE	280
280	NSAME=NSAME+NPALTA	ROUTE	281
	GO TO (225,226,227,226).ITYPE	ROUTE	282
	225 IF (NPALTA.LT.MINLOAD(ITYPE)) GO TO 262	ROUTE	283
	GO TO 240	ROUTE	284
	226 IF (NPALTA.GE.MINLOAD(ITYPE)) GO TO 240	ROUTE	285
285	IF (NSAME.LT.MINLOAD(ITYPE)) GO TO 262	ROUTE	285

		GO TO 280	ROUTE	247
	227	IF (NOFF+LDOFF.GE.MINLO(IITYPE)) GO TO 280	ROUTE	248
		IF (NSAME.GE.MINLO(IITYPE)) GO TO 280	ROUTE	249
		IF (NPALTA.GE.MINLO(IITYPE)) GO TO 280	ROUTE	250
290	C	TRY TO FIND NEXT AREA WITH MIN LOAD TO BE MOVED	ROUTE	291
	262	LSTSV(LSTA,NEXTA)=LSTA	ROUTE	292
		GO TO 211	ROUTE	293
	C	BUILD NEXT PORTION OF SCHEDULE BETWEEN AREAS	ROUTE	294
	C	LASTA AND NEXTA = TRAVEL TIME = TIME	ROUTE	295
295	C	CONSIDER ORDERS WITH SAME ORIGIN AND DESTINATION	ROUTE	296
	C	SAME DESTINATION - DIFFERENT ORIGINS	ROUTE	297
	C	DIFFERENT ORIGINS AND DESTINATIONS	ROUTE	298
	C	SELECT BEST SCHEDULE SEGMENT BY NO OF PALLETS / TIME	ROUTE	299
	C	ADD SEGMENT TO CURRENT VEHICLE SCHEDULE	ROUTE	300
300	C	UPDATE LAST AREA ,LASTA, AND LAST WAREHOUSE VISITED, LASTW	ROUTE	301
	280	TIME=T(LASTA,LSTA,IITYPE)	ROUTE	302
		IF (IFILL.NE.0.AND.LOAD.LT.MINLO(IITYPE)) GO TO 391	ROUTE	303
		TIMES=TIME+T(LSTA,LPA,IITYPE)+STIME(ITRUCK)	ROUTE	304
		TIMES=TIMES+STIME(ITRUCK)*2.*LTIME(ITRUCK)*FLOAT(MINLO(IITYPE))	ROUTE	305
305		IF (TLEFT(ITRUCK)-TIMES).LE.0.0) GO TO 200	ROUTE	306
		TIMES=TIME+STIME(ITRUCK)+T(NEXTA,LPA,IITYPE)	ROUTE	307
		IF (IFILL.NE.0.AND.NSAVE.EQ.NEXTA) GO TO 395	ROUTE	308
		IF (TLEFT(ITRUCK)-TIMES).LE.0.0) GO TO 200	ROUTE	309
	390	TLEFT(ITRUCK)=TLEFT(ITRUCK)-TIME	ROUTE	310
311		395 LASTA=LSTA	ROUTE	311
		LASTW=LSTW	ROUTE	312
	400	TIME=T(LASTA,NEXTA,IITYPE)	ROUTE	313
		LSAVE=LSTA	ROUTE	314
		NSAVE=NEXTA	ROUTE	315
315		GO TO (1000,2000,3000,3000),IITYPE	ROUTE	316
	C	BUILD SCHEDULES FOR STRADDLES	ROUTE	317
	1000	CALL BLDSTR(IFILL,IRCAP,LSTORD,NDELCO)	ROUTE	318
		IF (IRCAP.LE.0) GO TO 4050	ROUTE	319
		GO TO 3001	ROUTE	320
320	C	BUILD SCHEDULES FOR TRANSPORTERS	ROUTE	321
	2000	CALL BLDTR(IFILL,IRCAP,LSTORD)	ROUTE	322
		IF (IRCAP.LE.0) GO TO 4050	ROUTE	323
		GO TO 3001	ROUTE	324
	C	BUILD SCHEDULES FOR TRACTOR TRAILERS	ROUTE	325
325	3000	CALL BLDTR(IFILL,IRCAP,LSTORD)	ROUTE	326
		IF (LASTA.GT.10) NOFF=NOFF-NPALTS	ROUTE	327
		IF (IRCAP.LE.0) GO TO 4050	ROUTE	328
	3001	IF (NPALTS.LE.0) GO TO 4000	ROUTE	329
		IF (MOD(IPASS,2).NE.0) GO TO 4050	ROUTE	330
330		DO 3002 II=1,NORDR	ROUTE	331
		IF (INFC(II).LE.0) GO TO 3002	ROUTE	332
		IF (IAREA(II)/100.GT.10) GO TO 3002	ROUTE	333
		IF (NSAVE.NE.IAREA(II)/100) GO TO 3002	ROUTE	334
		GO TO 4050	ROUTE	335
335	3002	CONTINUE	ROUTE	336
		LJAO=LOAD+NPALTS	ROUTE	337
		IF (NEXTA.GT.10) LDOFF=LDOFF+NPALTS	ROUTE	338
		IF (LSAVE.EQ.NSAVE) GO TO 4050	ROUTE	339
		LASTA=LSAVE	ROUTE	340
340		GO TO 4000	ROUTE	341
	4050	JFILL=IABS(IFILL)	ROUTE	342
		IF (IITYPE.EQ.1) GO TO 4050	ROUTE	343

SUBROUTINE ROUTE 74/74 OPT=0 ROUND=9/ TRACE

FTN 4.6+461

12/12/80 08.07.55

	IF (JFILL.LT.3) GO TO 4060	ROUTE	344
	CALL SRTDST(LSTORD,I TYPE,JFILL)	ROUTE	343
345	4060 IRCAP=CAPAC(ITRJCK)	ROUTE	345
	IFILL=0	ROUTE	347
	LOAD=0	ROUTE	348
	LOOFF=0	ROUTE	349
	IF (TLEFT(ITRUCK).LE.0.0) GO TO 200	ROUTE	350
350	IF (NPALTS.GT.0) GO TO 150	ROUTE	351
	LSTSV(LSAVE,NSAVE)=1	ROUTE	352
	GO TO 150	ROUTE	353
	4000 LASTA=LSAVE	ROUTE	354
	NPALTL=NPALTL-NPALTS	ROUTE	355
355	IF (NPALTS.LE.0) LSTSV(LSAVE,NSAVE)=1	ROUTE	356
	IF (TLEFT(ITRUCK).LE.0.0) GO TO 200	ROUTE	357
	IF (IPASS.LT.4) GO TO 300	ROUTE	358
	IF (LSTSV(LSAVE,NSAVE).EQ.1) GO TO 150	ROUTE	359
	IPASS=IPASS	ROUTE	360
360	GO TO 300	ROUTE	361
	200 CONTINUE	ROUTE	362
	GO TO 100	ROUTE	363
	C CHECK IF ORDERS CAN BE ADDED TO BEGINNING OF SCHEDULE	ROUTE	364
	9000 IF (ICNTRL.NE.1) GO TO 9005	ROUTE	365
365	IF (NSPLIT.EQ.2) GO TO 9010	ROUTE	366
	9005 IF (NSPLIT.EQ.3) GO TO 9010	ROUTE	367
	NSPLIT=NSPLIT+1	ROUTE	368
	GO TO 50	ROUTE	369
	9010 IF (IRESET.NE.0) RETURN	ROUTE	370
370	IRESET=1	ROUTE	371
	ICNTRL=0	ROUTE	372
	ISPASS=0	ROUTE	373
	NSPLIT=1	ROUTE	374
	DO 9020 II=1,NORDR	ROUTE	375
375	IF (INFO(II).LE.0) GO TO 9020	ROUTE	376
	GO TO 50	ROUTE	377
	9020 CONTINUE	ROUTE	378
	RETURN	ROUTE	379
	END	ROUTE	380

ROUTINE: SRTDST

ARGUMENTS:

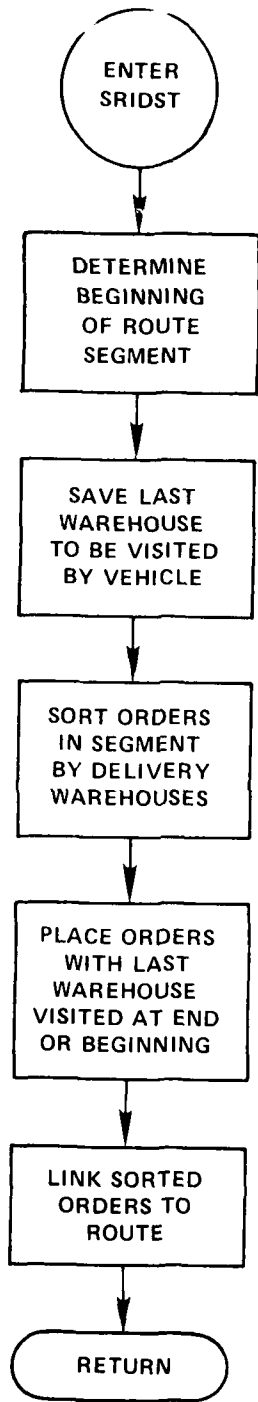
LSTORD - List number of last order added to route segment

ITYPE - Vehicle type number

JFILL - Number of orders in route segment

Description:

SRTDST re-orders the last route segment for a given vehicle. Order destinations are re-arranged to allow maximum vehicle utilization with a minimum travel time.



1	SUBROUTINE SRTDST(LSTOPD, ITYPE, JFILL)	SRTD	2
	C	SRTD	3
	C	SRTD	4
	C*****	SRTD	5
5	C	SRTD	6
	C SRTDST SORTS ENTRIES IN A ROUTE SEGMENT BY DESTINATION	SRTD	7
	C WAREHOUSE AND GROUPS COMMON DESTINATIONS ON A	SRTD	8
	C FIRST ON - LAST OFF BASIS	SRTD	9
	C	SRTD	10
11	C*****	SRTD	11
	C	SRTD	12
	C	SRTD	13
	C LSTOPD - LAST LIST ENTRY OF ROUTE SEGMENT	SRTD	14
	C ITYPE - VEHICLE TYPE	SRTD	15
15	C JFILL - NUMBER OF ORDERS IN ROUTE SEGMENT	SRTD	16
	C	SRTD	17
	C	SRTD	18
	C INTEGER ONUMB, WARE(14), PTR(14)	SRTD	19
	C COMMON	SRTD	20
21	C */SCHEDL/ ONUMB(200), INFO(200), IDESTP(200), IAREA(200),	SRTD	21
	C * LFRWD(200), LPKWD(200), IDUM(400), OUM(400)	SRTD	22
	C DO 50 I=1, 14	SRTD	23
	C WARE(I)=0	SRTD	24
	C 50 PTR(I)=0	SRTD	25
25	C FIND START OF SEGMENT	SRTD	26
	C ISTART=LSTOPD	SRTD	27
	C ICNT=0	SRTD	28
	C 100 WARE(JFILL-ICNT)=MOD(IARS(INFO(ISTART))/100, 100)	SRTD	29
	C PTR(JFILL-ICNT)=ISTART	SRTD	30
31	C ICNT=ICNT+1	SRTD	31
	C IF(JFILL.NE.ICNT) GO TO 200	SRTD	32
	C IFIRST=ISTART	SRTD	33
	C GO TO 300	SRTD	34
35	C 200 ISTART=LPKWD(ISTART)	SRTD	35
	C GO TO 100	SRTD	36
	C GROUP LIKE DESTINATIONS	SRTD	37
	C SORT ORDER SEGMENT	SRTD	38
	C 300 LIMIT=ICNT-1	SRTD	39
	C I=1	SRTD	40
41	C 350 JSTART=I+1	SRTD	41
	C MCNT=0	SRTD	42
	C DO 400 J=JSTART, ICNT	SRTD	43
	C IF(WARE(I).NE.WARE(J)) GO TO 400	SRTD	44
	C MCNT=MCNT+1	SRTD	45
45	C IF(J.EQ.JSTART) GO TO 400	SRTD	46
	C ISAVE=WARE(I+MCNT)	SRTD	47
	C JSAVE=PTR(I+MCNT)	SRTD	48
	C WARE(I+MCNT)=WARE(J)	SRTD	49
	C PTR(I+MCNT)=PTR(J)	SRTD	50
51	C WARE(J)=ISAVE	SRTD	51
	C PTR(J)=JSAVE	SRTD	52
	C 400 CONTINUE	SRTD	53
	C IF(MCNT.FO.0) I=I+1	SRTD	54
	C I=I+MCNT	SRTD	55
55	C IF(I.LE.LIMIT) GO TO 350	SRTD	56
	C ADJUST LINKED LISTS POINTERS WITH DELIVERY SEQUENCE	SRTD	57
	C JCNT=ICNT-1	SRTD	58

SUBROUTINE SRTDST 73/74 OPT=? ROUND=?/ TRACE

FTN 4.6+468

12/12/80 18.04.00

```
60      DO 500 I=1,JCNT
        LINK=PTR(I)
        LFRWD(LINK)=PTR(I+1)
        IF(I.GT.1) LBKWD(LINK)=PTR(I-1)
        IDESTP(LINK)=-I
500     CONTINUE
65     LINK=PTR(ICNT)
        IDESTP(LINK)=-ICNT
        LFRWD(LINK)=0
        LBKWD(LINK)=PTR(ICNT-1)
        RETURN
        END
```

```
SRTD 59
SRTD 60
SRTD 61
SRTD 62
SRTD 63
SRTD 64
SRTD 65
SRTD 66
SRTD 67
SRTD 68
SRTD 69
SRTD 70
```

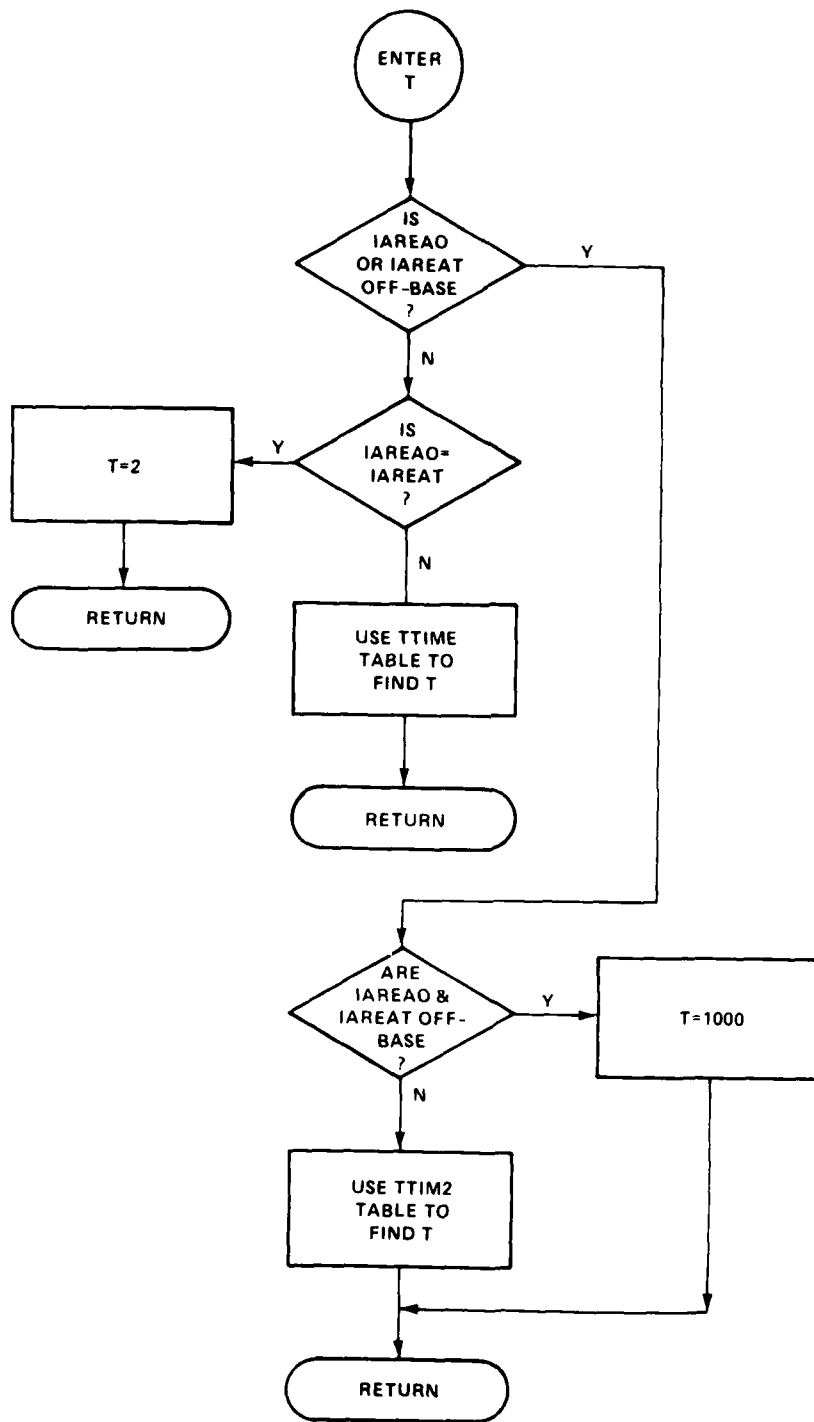

ROUTINE: T

ARGUMENTS:

IAREAO - Origin area number
IAREAT - Destination area number
K - Vehicle type number

Description:

T computes the travel time between two areas, IAREAO and IAREAT, for a vehicle of type K. If IAREAO equals IAREAT, T is set equal to 2 minutes. If IAREAO and IAREAT are both off base sites, T is set equal to 1000 minutes to prevent travel between the two areas.



FUNCTION T 76/76 OPT=8 ROUND=0/ TRACE FTH 4.6+600 10/17/80 10.04.17

```

1      FUNCTION T(IAREAD,IAREAT,K)
C
C      THIS FUNCTION GIVES THE TRAVEL TIME BETWEEN POINTS IO AND IT
C      FOR TRUCK TYPE K
5      TIMES ARE STORED IN TRIANGULAR ARRAYS WITH ZERO ON THE DIAGONAL
C
C      ALPHA MHNAN
C      *** COMMONS ***
10     COMMON/TINTAB/ TTIME(3,65),TTIM2(6)
COMMON/WHOUSE/ WHAAM(92)
C
C      NGRUP MUST BE CHANGED IF WAREHOUSE GROUPING CHANGES
C
15     NGRUP = 10
KSAVE = K
IF(K.EQ.4) K=1
1000  IF(IAREAD.EQ.IAREAT) GO TO 4500
1500  IF(IAREAD.GT.INGRUP.OR.IAREAT.GT.INGRUP) GO TO 5000
20     C
C      FIND TIME FOR SITES IN DIFFERENT AREAS
C
L=MIN0(IAREAD,IAREAT)
M=MAX0(IAREAD,IAREAT)
IF(L.EQ.1) GO TO 3000
N = L - 1
ISUM = 4
DO 2000 IX = 1, N
2000  ISUM = ISUM + (INGRUP - IX)
30     IL = ISUM - L
I=TTIME(K,IL)
GO TO 6000
3000  T = TTIME(K,M-1)
GO TO 6000
35     4000  T = 0.0
GO TO 6000
C
C      FIND TIME FOR SITES IN SAME AREA
C
40     4500  T = 2.0
GO TO 6000
C
C      FIND TIME FOR OFFBASE SITES
C
45     5000  IF(IAREAD.GT.INGRUP.AND.IAREAT.GT.INGRUP) GO TO 5500
IF(K.EQ.3) GO TO 5500
N=MAX0(IAREAD,IAREAT)
N = N - NGRUP
T = TTIM2(N)
50     GO TO 6000
C
C      T = 1000. PREVENTS TRAVEL BETWEEN OFFBASE SITES
C
55     5500  T=1000.0
6000  K=KSAVE
RETURN
END

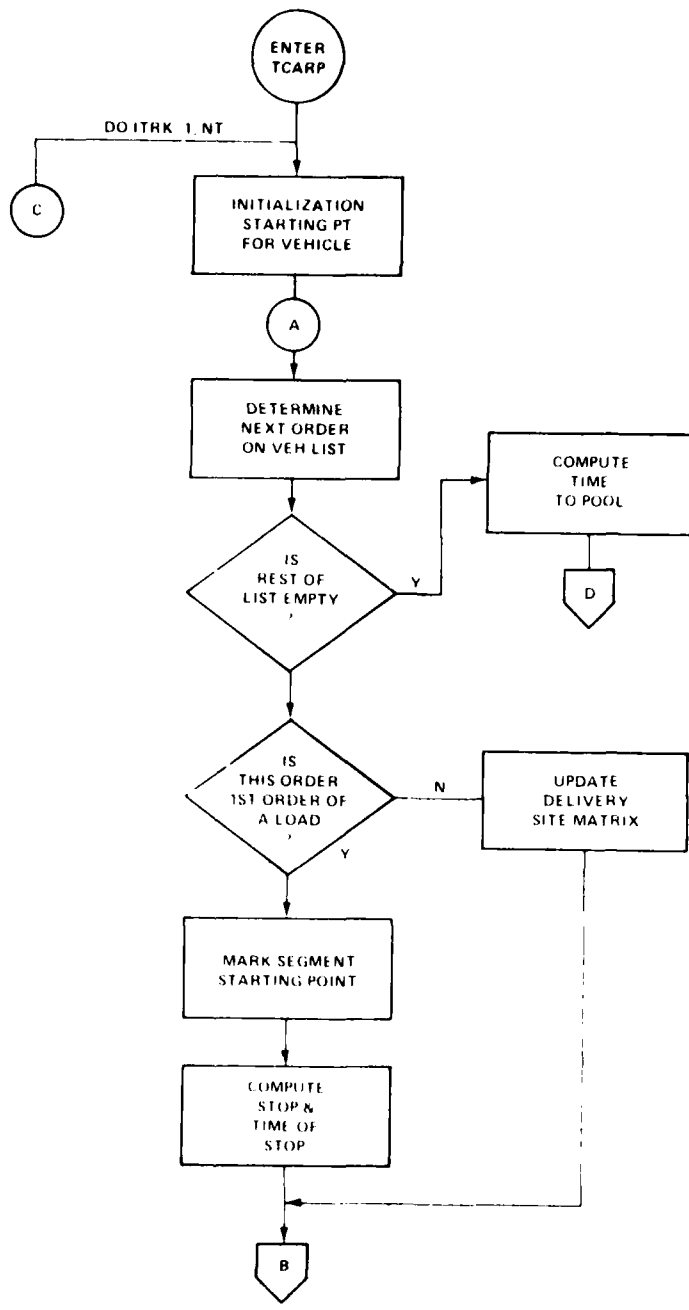
```

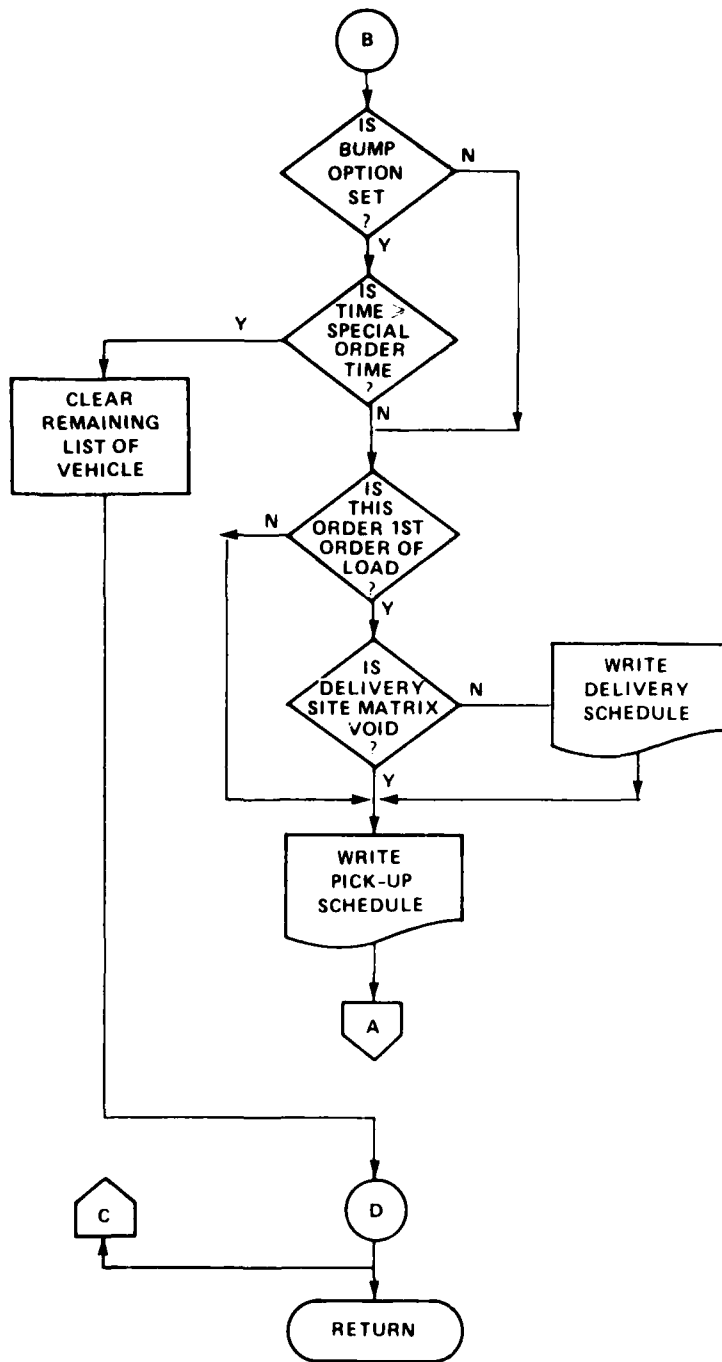
ROUTINE: TCARP

Description:

The primary purpose of TCARP is to translate the linked lists notation for each vehicle in service into readable schedules.

When the bump option is set for a special order run (AVS2), TCARP traces through each vehicle's schedule to determine whether the vehicle is suitable to move special orders. Each suitable vehicle is made available for the special order run.





```

1          SUBROUTINE TCA6P                                TCARP  2
C          .....                                        TCARP  3
C          TCARP - DECODES SCHEDULES FROM THE LINKED LISTS TCARP  4
C          FOR EACH VEHICLE USED                          TCARP  5
C          IT ALSO PRINTS ALL UNMOVED ORDERS             TCARP  6
C          .....                                        TCARP  7
C          .....                                        TCARP  8
C          .....                                        TCARP  9
10         C
          INTEGER ONUMB,TRUCK,PTSQR,PTTRK,NDEST(15)        TCARP 10
          INTEGER TRKSAV,CAPAC                             TCARP 11
          ALPHA MHNAM,TYPE(4),PRTS(2),IPRT               TCARP 12
          REAL LTIME                                       TCARP 13
          COMMON                                           TCARP 14
15         * /GEN/ RTTIM,MIALD(4)                          TCARP 15
          * /MSCLNS/ NORDR,NTRKS(4),PTTRK(4)             TCARP 16
          * /SAV/P/ NSAV(4),TRKSAV(4)                    TCARP 17
          * /SPORD/ ICNTRL,SAVTIM,ISORD                   TCARP 18
          * /HOUSE/ MHNAM(92)                             TCARP 19
20         * /SCHEDL/ ONUMB(200),INFO(200),IOESTP(200),   TCARP 20
          AREA(200),LFRWD(200),
          LCKWD(200),LCFWD(200),LCBWD(200),ATIME(200),    TCARP 21
          STOPT(200)
          * /TRUCKS/ PTSQR(50),TRUCK(50),CAPAC(50),STIME(50),LTIME(50), TCARP 22
          * /RIL/ RILIN(50),TLEFT(50),NT                 TCARP 23
          * /INOPT/ SHIFT,IOATE                           TCARP 24
25         DATA TYPE/2MST,2MTR,2HIT,2MIT/              TCARP 25
          DATA PRTS/6H      .6H*SPCL* /                TCARP 26
          C          LIST ALL UNMOVED ORDERS              TCARP 27
          PRTSHF=SHIFT                                     TCARP 28
          SAVSHF=0.0                                      TCARP 29
30         IPCOL=23                                       TCARP 30
          JA1=2                                           TCARP 31
          DJ 100J JJ=1.4                                  TCARP 32
          IF(NTRKS(JJ),LE,0) GO TO 1000                  TCARP 33
          MSTART=NTRKS(JJ)                               TCARP 34
35         DJ 2000 I=1,MSTART                             TCARP 35
          ITRK=PTTRK(IJ)+1
          IF(PTSQR(ITRK),LE,0) GO TO 2000                TCARP 36
          NPALTS=0                                       TCARP 37
          ISTOP=0                                         TCARP 38
          TIME=0.0                                        TCARP 39
40         LSTOP=IPOOL                                    TCARP 40
          IA1=JA1                                         TCARP 41
          NUFL=0                                          TCARP 42
          LINK=PTSQR(ITRK)                                TCARP 43
          LNKSAV=LINK                                     TCARP 44
45         C          LEFT: DETERMINE VEHICLE TYPE        TCARP 45
          ITYPE=1                                         TCARP 46
          IF(ITRK.GT,PTTRK(2)) ITYPE=2                  TCARP 47
          IF(ITRK.GT,PTTRK(3)) ITYPE=3                  TCARP 48
          IF(ITRK.GT,PTTRK(4)) ITYPE=4                  TCARP 49
50         JWRIG=0                                        TCARP 50
          ITIME=SHIFT                                     TCARP 51
          IVEH=ITRK-TRKSAV(ITYPE)                        TCARP 52
          IF(TRKSAV(ITYPE),LT,0) IVEH=ITRK-PTTRK(ITYPE) TCARP 53
55         IF(ICNTRL.EQ,0) GO TO 2010                    TCARP 54
          IPHT=PRTS(1)                                    TCARP 55
          WRITE(6,602) TYPE(ITYPE),IVEH,SHIFT,IOATE     TCARP 56
          TCARP 57
          TCARP 58

```

	2010	INFO(LINK)=IAES(INFO(LINK))	TCARP 59
		IF(ICNTRL.NE.0) GO TO 2014	TCARP 60
60		IF(IABS(IDESEP(LINK)).NE.1) GO TO 2015	TCARP 61
		IF(itime.LT.IFIX(SAVTIM)) GO TO 2015	TCARP 62
	C	CLEAR VEHICLE SCHEDULE	TCARP 63
	2025	INFO(LINK)=IABS(INFO(LINK))	TCARP 64
		IDESTP(LINK)=0	TCARP 65
65		LAST=LKNO(LINK)	TCARP 66
		LFRWD(LAST)=0	TCARP 67
		LKNO(LINK)=0	TCARP 68
		STOPT(LINK)=0.0	TCARP 69
		ATIME(LINK)=0.0	TCARP 70
70		LINK=LFRWD(LINK)	TCARP 71
		IF(LINK.GT.0) GO TO 2025	TCARP 72
	C	UPDATE TIME REMAINING FOR VEHICLE	TCARP 73
		ISS=SHIFT	TCARP 74
		XSHFT=(ISS/100)*50+MOD(ISS,100)	TCARP 75
75		TLEF=(itime/100)*60+MOD(itime,100)	TCARP 76
		TLEF=RTTIM-(TLEF-XSHFT)	TCARP 77
		TLEFT(ITRK)=TLEF	TCARP 78
		GO TO 2005	TCARP 79
	2015	LINKS=LFRWD(LINK)	TCARP 80
80		IF(LINKS.GT.0) GO TO 2014	TCARP 81
		ISS=SHIFT	TCARP 82
		XSHFT=(ISS/100)*60+MOD(ISS,100)	TCARP 83
		ISS=SAVTIM	TCARP 84
		TLEFT(ITRK)=(ISS/100)*60+MOD(ISS,100)	TCARP 85
85		TLEFT(ITRK)=RTTIM-(TLEFT(ITRK)-XSHFT)	TCARP 86
	2014	IF(ATIME(LINK).GT.TIME) TIME=ATIME(LINK)	TCARP 87
		IF(ATIME(LINK).NE.0.0) GO TO 2016	TCARP 88
		IF(SAVSHF.NE.0.0) GO TO 2016	TCARP 89
		SAVSHF=SHIFT	TCARP 90
90		ISS=SHIFT	TCARP 91
		XSHFT=(ISS/100)*60+MOD(ISS,100)	TCARP 92
		SHIFT=SAVTIM	TCARP 93
		ISS=SAVTIM	TCARP 94
		TME=(ISS/100)*60+MOD(ISS,100)	TCARP 95
95		TME=TME-XSHFT	TCARP 96
		IF(TME.LE.TIME) GO TO 2016	TCARP 97
		IA1=?	TCARP 98
		TIME=TME	TCARP 99
	2016	CONTINUE	TCARP100
100		IORIG=INFO(LINK)/10000	TCARP101
		IDEST=MOD(INFO(LINK)/100,100)	TCARP102
		ISIZE=MOD(INFC(LINK),100)	TCARP103
		NPALTS=NPALTS+ISIZE	TCARP104
		NDEL=NDEL+1	TCARP105
105		NDEST(NDEL)=IDEST	TCARP106
		IA2=IAREA(LINK)/100	TCARP107
		IF(IORIG.EQ.JGRIG) GO TO 4005	TCARP108
		ATIME(LINK)=TIME+I(IA1,IA2,ITYPE)	TCARP109
		TIME=ATIME(LINK)	TCARP110
110		ITIME=TCNV(TIME,PRTSHF)	TCARP111
		STAY=LTIME(ITRK)*FLOAT(ISIZE)+STIME(ITRK)	TCARP112
		TIME=TIME+STAY	TCARP113
		ISTAY=ISTAY+.9	TCARP114
		ISTOP=ISTOP+1	TCARP115

115	IF(ICNTRL.EQ.0) GO TO 8000	TCARP116
	NUM=IABS(ONUMB(LINK))	TCARP117
	IF(ONUMB(LINK).GE.0) GO TO 101	TCARP118
	IPRT=PRTS(2)	TCARP119
	IF(NUM.LE.100) IPRT=6H*SPLIT	TCARP120
120	NUM=MOD(NUM,100)	TCARP121
	101 WRITE(6,603) ISTOP,MHNM(IORIG),ITIME,ISIZE,NUM,ISTAY,IPRT	TCARP122
	IPRT=PRTS(1)	TCARP123
	8000 JORIG=JORIG	TCARP124
	GO TO 4010	TCARP125
125	4005 STAY=LTIME(ITRK)*FLOAT(ISIZE)	TCARP126
	ISTAY=STAY+.9	TCARP127
	ATIME(LINK)=TIME	TCARP128
	TIME=TIME+STAY	TCARP129
	IF(ICNTRL.EQ.0) GO TO 4010	TCARP130
130	NUM=IABS(ONUMB(LINK))	TCARP131
	IF(ONUMB(LINK).GE.0) GO TO 102	TCARP132
	IPRT=PRTS(2)	TCARP133
	IF(NUM.LE.100) IPRT=6H*SPLIT	TCARP134
	NUM=MOD(NUM,100)	TCARP135
135	102 WRITE(6,605) ISIZE,NUM,ISTAY,IPRT	TCARP136
	IPRT=PRTS(1)	TCARP137
	4010 JDEST=0	TCARP138
	NEXT=LFRWD(LINK)	TCARP139
	NXTSAV=NEXT	TCARP140
140	IF(NEXT.LE.0) GO TO 2020	TCARP141
	IF(IJESTP(LINK).EQ.IDESTP(NEXT)) GO TO 3000	TCARP142
	IF(ITYPE.EQ.1) GO TO 4000	TCARP143
	IF(IJESTP(LINK).LT.IDESTP(NEXT)) GO TO 3200	TCARP144
	GO TO 5000	TCARP145
145	4000 IF(IJESTP(LINK).GT.IDESTP(NEXT)) GO TO 3100	TCARP146
	5000 LINK=NEXT	TCARP147
	LSTOP=JORIG	TCARP148
	IA1=IA2	TCARP149
	GO TO 2010	TCARP150
150	C COMPUTE TIME ONE DESTINATION	TCARP151
	3000 IA2=MOG(IAREA(LINK),100)	TCARP152
	TIME=TIME+T(IA1,IA2,ITYPE)	TCARP153
	STOPT(LINK)=TIME	TCARP154
	ISTOP=ISTOP+1	TCARP155
155	ITIME=CONV(TIME,PRTSMF)	TCARP156
	STAY=LTIME(ITRK)*FLOAT(ISIZE)+STIME(ITRK)	TCARP157
	TIME=TIME+STAY	TCARP158
	ISTAY=STAY+.9	TCARP159
	IF(ICNTRL.EQ.0) GO TO 8001	TCARP160
160	NUM=IABS(ONUMB(LINK))	TCARP161
	IF(ONUMB(LINK).GE.0) GO TO 103	TCARP162
	IPRT=PRTS(2)	TCARP163
	IF(NUM.LE.100) IPRT=6H*SPLIT	TCARP164
	NUM=MOD(NUM,100)	TCARP165
165	103 WRITE(6,604) ISTOP,MHNM(IDEST),ITIME,ISIZE,NUM,ISTAY,IPRT	TCARP166
	IPRT=PRTS(1)	TCARP167
	8001 JDEST=0	TCARP168
	JORIG=IDEST	TCARP169
	LINK=NEXT	TCARP170
170	LNKSAV=NEXT	TCARP171
	NOEL=0	TCARP172

		IA1=IA2	TCARP173
		LSTOP=IDEST	TCARP174
		GO TO 2010	TCARP175
175	C	FIRST ON - FIRST OFF	TCARP176
		2020 IF (ITYPE.EQ.1) GO TO 3100	TCARP177
		IA1=IA2	TCARP179
		LNKSAV=LINK	TCARP179
		GO TO 3210	TCARP180
180		3100 IF (NOEL.EQ.0) GO TO 7000	TCARP181
		DO 6000 II=1,NOEL	TCARP182
		IDEST=NOEST(II)	TCARP183
		IA2=MOD(IAREA(LNKSAV),100)	TCARP184
		ISIZE=MOD(INFC(LNKSAV),100)	TCARP185
185		IF (JDEST.NE.IDEST) GO TO 3105	TCARP186
		STOPT(LNKSAV)=TIME	TCARP187
		STAY=LTIME(ITRK)*FLOAT(ISIZE)	TCARP188
		TIME=TIME+STAY	TCARP189
		ISTAY=STAY*.9	TCARP190
190		IF (ICNTRL.EQ.0) GO TO 3110	TCARP191
		NUM=IABS(ONUMB(LNKSAV))	TCARP192
		IF (ONUMB(LNKSAV).GE.0) GO TO 104	TCARP193
		IPRT=PRTS(2)	TCARP194
		IF (NUM.LE.100) IPRT=6H*SPLIT	TCARP195
195		104 WRITE(6,607) ISIZE,NUM,ISTAY,IPRT	TCARP196
		IPRT=PRTS(1)	TCARP197
		GO TO 3110	TCARP198
		3105 ISTOP=ISTOP+1	TCARP199
		JDEST=IDEST	TCARP200
200		TIME=TIME+(IA1,IA2,ITYPE)	TCARP201
		STOPT(LNKSAV)=TIME	TCARP202
		STAY=LTIME(ITRK)*FLOAT(ISIZE)+STIME(ITRK)	TCARP203
		TIME=TIME+STAY	TCARP204
		ISTAY=STAY*.9	TCARP205
205		ITIME=TCONV(STOPT(LNKSAV),PRTSHF)	TCARP206
		IF (ICNTRI.EQ.0) GO TO 3110	TCARP207
		NUM=IABS(ONUMB(LNKSAV))	TCARP208
		IF (ONUMB(LNKSAV).GE.0) GO TO 105	TCARP209
		IPRT=PRTS(2)	TCARP210
210		IF (NUM.LE.100) IPRT=6H*SPLIT	TCARP211
		NUM=MOD(NUM,100)	TCARP212
		105 WRITE(6,604) ISTOP,WHNAM(IDEST),ITIME,ISIZE,NUM,ISTAY,IPRT	TCARP213
		IPRT=PRTS(1)	TCARP214
215		3110 IA1=IA2	TCARP215
		LNKSAV=LFRWD(LNKSAV)	TCARP216
		LSTOP=IDEST	TCARP217
		6000 CONTINUE	TCARP218
		NOEL=0	TCARP219
220		IF (LNKSAV.LE.0) GO TO 7000	TCARP220
		LINK=LNKSAV	TCARP221
		JORIG=IDEST	TCARP222
		GO TO 2010	TCARP223
	C	FIRST ON - LAST OFF	TCARP224
		3200 LNKSAV=LORKND(NEXT)	TCARP225
225		3210 IA2=MOD(IAREA(LNKSAV),100)	TCARP226
		IDEST=MOD(INFC(LNKSAV)/100,100)	TCARP227
		ISIZE=MOD(INFC(LNKSAV),100)	TCARP228
		IF (JDEST.NE.IDEST) GO TO 3205	TCARP229

230	STOPT(LNKSAV)=TIME	TCAPP230
	STAY=LTIME(ITRK)*FLOAT(ISIZE)	TCAPP231
	ISTAY=STAY*.9	TCAPP232
	TIME=TIME+STAY	TCAPP233
	IF(ICNTRL.EQ.0) GO TO 3220	TCAPP234
	NUM=IABS(ONUMB(LNKSAV))	TCAPP235
235	IF(ONUMB(LNKSAV).GE.0) GO TO 106	TCAPP236
	IPRT=PRTS(2)	TCAPP237
	IF(NUM.LE.100) IPRT=6H*SPLIT	TCAPP238
	NUM=MOD(NUM,100)	TCAPP239
106	WRITE(6,607) ISIZE,NUM,ISTAY,IPRT	TCAPP240
240	IPRT=PRTS(1)	TCAPP241
	GO TO 3220	TCAPP242
3205	IOEST=IOEST	TCAPP243
	ISTOP=ISTOP+1	TCAPP244
	TIME=TIME+T(IA1,IA2,ITYPE)	TCAPP245
245	STOPT(LNKSAV)=TIME	TCAPP246
	STAY=LTIME(ITRK)*FLOAT(ISIZE)+STIME(ITRK)	TCAPP247
	TIME=TIME+STAY	TCAPP248
	ISTAY=STAY*.9	TCAPP249
	ITIME=TCONV(STOPT(LNKSAV),PRTSHF)	TCAPP250
250	IF(ICNTRL.EQ.C) GO TO 3220	TCAPP251
	NUM=IABS(ONUMB(LNKSAV))	TCAPP252
	IF(ONUMB(LNKSAV).GE.0) GO TO 107	TCAPP253
	IPRT=PRTS(2)	TCAPP254
	IF(NUM.LE.100) IPRT=6H*SPLIT	TCAPP255
255	NUM=MOD(NUM,100)	TCAPP256
107	WRITE(6,604) ISTOP,WHNAM(IOEST),ITIME,ISIZE,NUM,ISTAY,IPRT	TCAPP257
	IPRT=PRTS(1)	TCAPP258
3220	IA1=IA2	TCAPP259
	LSTOP=IOEST	TCAPP260
260	NDEL=NDEL-1	TCAPP261
	NEXT=LNKSAV	TCAPP262
	IF(NDEL.GT.0) GO TO 3200	TCAPP263
	IF(NXTSAV.LE.0) GO TO 7000	TCAPP264
	LINK=NXTSAV	TCAPP265
265	LNKSAV=NXTSAV	TCAPP266
	JORIG=IOEST	TCAPP267
	GO TO 2010	TCAPP268
C	END ROUTE FOR VEHICLE	TCAPP269
7000	TIME=TIME+T(IA1,JA1,ITYPE)	TCAPP270
270	ITIME=TCONV(TIME,PRTSHF)	TCAPP271
	IF(ICNTRL.EQ.C) GO TO 2005	TCAPP272
	WRITE(6,606) WHNAM(IPOOL),ITIME,NPALTS	TCAPP273
2005	IF(SAVSHF.LE.0.0) GO TO 2003	TCAPP274
	SHIF=SAVSHF	TCAPP275
275	SAVSHF=0.0	TCAPP276
2000	CONTINUE	TCAPP277
1000	CONTINUE	TCAPP278
	DO 9000 I=1,NCRRD	TCAPP279
	INFO(I)=IABS(INFO(I))	TCAPP280
280	IF(IDESTP(I).EQ.0) GO TO 9000	TCAPP281
	INFO(I)=-INFO(I)	TCAPP282
9000	CONTINUE	TCAPP283
	IF(ICNTRL.EQ.0) RETURN	TCAPP284
	WRITE(6,600)	TCAPP285
285	DO 9001 I=1,NCRRD	TCAPP286

SUBROUTINE TCARP

74/74

OPT=0 ROUND=* / TRACE

FTN 4.6460

10/17/80 10.04.12

```

      IF(INFC(I).LE.0) GC TO 9001
      IQ=IABS(INFC(I))/1000
      IT=MOD(IABS(INFC(I))/100,100)
      JSIZE=MOD(IABS(INFC(I)),100)
290   WRITE(6,601) CAUME(I),JSIZE,WHNAM(IO),WHNAM(IT)
      9001 CONTINUE
      RETURN
      600 FORMAT(1H1,50(1H*)/5X,18HORDERS NOT MOVED /1X,50(1H*)///)
      601 FORMAT(1X,6HORDER ,I4,2H, ,I2,6HFROM ,A6,4H TO ,A6)
295   602 FORMAT(1H1/// 20X,13HVEHICLE - ,A2,I2/20X,12HSTART TIME - ,
      * F5.0/20X,4HDATE,9X,I6///1H0.76(1H-)/6X,4HSTOP,2X,4HSITE,3X,4HTIME
      * 5X,7HDELIVER,7X,7HPICK UP ,5X,5HORDEF,3X,15HSTAY TIME (MIN) /
      * 1X,76(1H-))
      603 FORMAT(7X,I3,1X,A6,1X,I5,17X,I2,8H PALLETS ,4X,I3,4X,I5,1X,A6)
300   604 FORMAT(7X,I3,1X,A6,1X,I5,3X,I2,8H PALLETS ,18X,I3,4X,I5,1X,A6)
      605 FORMAT(40X,I2,8H PALLETS ,4X,I3,4X,I5,1X,A6)
      606 FORMAT(//20X,11HROUTE ENDED /20X,13HLOCATION = ,A6/
      * 20X,6HTIME = ,I5/20X,21HNO OF PALLETS MOVED = ,I6)
305   607 FORMAT(26X,I2,8H PALLETS ,18X,I3,4X,I5,1X,A6)
      END
      TCARP287
      TCARP288
      TCARP289
      TCARP290
      TCARP291
      TCARP292
      TCARP293
      TCARP294
      TCARP295
      TCARP296
      TCARP297
      TCARP298
      TCARP299
      TCARP300
      TCARP301
      TCARP302
      TCARP303
      TCARP304
      TCARP305
      TCARP306
```

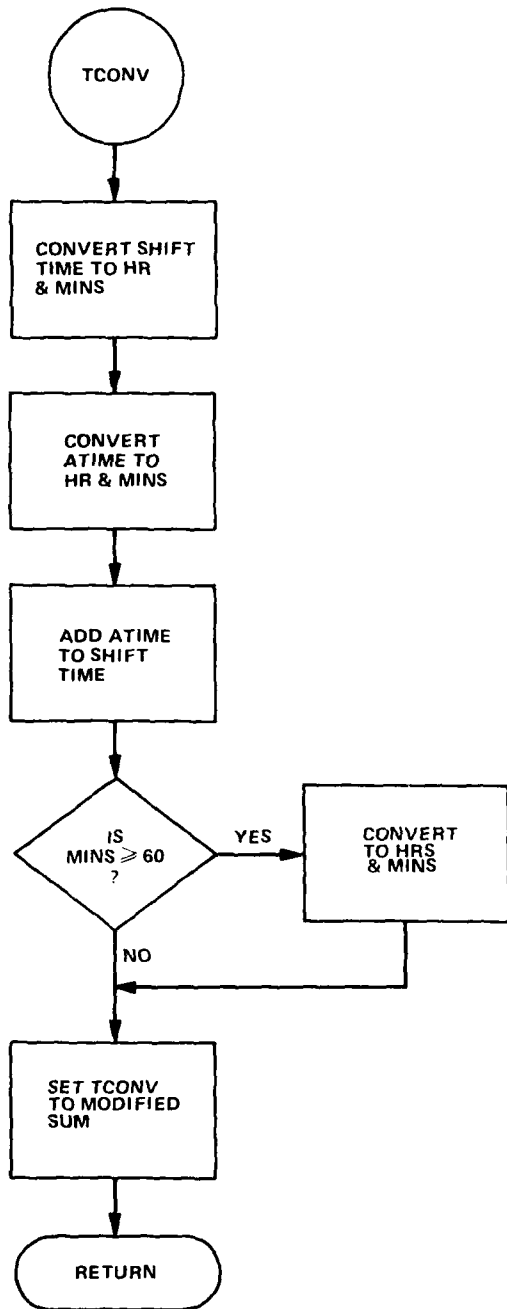
ROUTINE: TCONV

ARGUMENTS:

- A - Relative time in minutes
- S - Start of work shift (24-hour clock)

Description

TCONV adds the relative time to the start time of the shift. The sum is converted to 24-hour clock time. TCONV is then set equal to the result.



FUNCTION TCCAV

7-774

OPT=0 ROUNO=*/ TRACE

FTN 4.6+660

10/17/80

10.04.12

```
1      FUNCTION TCONV(A,S)                                TCONV 2
      C                                                    TCONV 3
      C THIS FUNCTION TAKES REAL INPUTS A = REL. TIME IN MINUTES TCONV 4
      C AND S = START OF SHIFT (24 HR. CLOCK) TCONV 5
5      C ADDS A TO S AND OUTPUTS THE RESULT, TCONV, AS 24 HR. CLOCK TIME TCONV 6
      C                                                    TCONV 7
      SHR = FLOAT(IFIX(S/100.)) TCONV 8
      SMIN = S - SHR*100. TCONV 9
      XHR = FLOAT(IFIX(A/60.)) TCONV 10
10     XMIN = A - XHR*60. TCONV 11
      YMIN = SMIN + XMIN TCONV 12
      YHR = SHR + XHR TCONV 13
      TCONV = 100.*YHR + YMIN TCONV 14
      IF(YMIN.LT.50.) RETURN TCONV 15
15     TCONV = TCONV + 48. TCONV 16
      RETURN TCONV 17
      END TCONV 18
```

APPENDIX C - AVSIN1, A SINR COBOL DRIVER ROUTINE, LISTINGS

001030 IDENTIFICATION DIVISION.
 001050 PROGRAM-ID.
 001070 AVSIN1.
 001090 AUTHOR.
 001110 R. MELTON
 001130 INSTALLATION.
 001150 DTNSRDC.
 001170 DATE-WRITTEN.
 001190 14 DEC 78.
 001210 DATE-COMPIL'D.
 001230 14 DEC 78.
 001250 SECURITY.
 001270 UNCLASSIFIED.
 001290 REMARKS.
 001310 COMBINED INPUT-FRAME PROGRAM FOR AVS1 AND AVS2.
 001330 ENVIRONMENT DIVISION.
 001350 CONFIGURATION SECTION.
 001370 SOURCE-COMPUTER.
 001390 R3500.
 001410 OBJECT-COMPUTER.
 001430 R3500.
 001450 INPUT-OUTPUT SECTION.
 001470 FILE-CONTROL.
 001490 SELECT SQFILE ASSIGN TO DISK
 001510 RESERVE NO ALTERNATE AREAS
 001530 ACCESS MODE IS RANDOM
 001550 ACTUAL KEY IS SQFILE-KEY.
 001610 SELECT VS2 IN ASSIGN TO DISK
 001630 RESERVE 1 ALTERNATE AREA.
 001650 SELECT VS3 IN ASSIGN TO DISK
 001670 RESERVE 1 ALTERNATE AREA.
 001690 DATA DIVISION.
 001710 FILE SECTION.
 001730 FD SQFILE COPY SQFILE.
 001750 01 AVS-RECORD.
 001770 03 AVS-PREFIX PIC X(24).
 001790 03 AVS-DATA PIC X(284).
 001810 01 INPUT-DATA.
 001830 03 FILLER PIC X(2-1).
 001850 03 COMMON-PREFIX.
 001870 05 OCCID PIC XXX.
 001890 05 FRAME-NR PIC X.
 001910 03 FILLER PIC X(280).
 001930 01 FRAME-1-INPUT.
 001950 03 FILLER PIC X(2-1).
 001970 03 FILLER PIC X(4).
 001990 03 ORDER OCCURS 2) TIMES.
 002010 05 O-SIZE PIC XX.
 002030 05 O-ORIG PIC KXXXXX.
 002050 05 O-DEST PIC X(6).
 002070 01 FRAME-2-INPUT.
 002090 03 FILLER PIC X(2-1).
 002110 03 FILLER PIC X(4).
 002130 03 TRUCK OCCURS 30 TIMES.

002150 05 V-USF PIC X.
002170 05 CAFAC PIC XX.
002190 05 TIME-LIM PIC XXX.
002210 03 FILLER PIC X(130).
002230 01 FRAME-3-INPUT.
002250 03 FILLER PIC X(24).
002270 03 FILLER PIC X(4).
002290 03 I-DATE PIC X(6).
002310 03 START-TIME PIC X(4).
002330 03 MAX-ROUTE PIC XXX.
002350 03 FILLER PIC X(256).
002370 01 FRAME-6-INPUT.
002390 03 FILLER PIC X(24).
002410 03 FILLER PIC X(4).
002430 03 M-OPTION PIC X.
002450 03 P-OPTION PIC X.
002470 03 FILLER PIC X(268).
002490 01 FRAME-7-INPUT.
002510 03 FILLER PIC X(24).
002530 03 FILLER PIC X(4).
002550 03 FCN-TYPE PIC X.
002570 03 START-DATE PIC X(6).
002590 03 H-OPTION PIC X.
002610 03 END-DATE PIC X(6).
002630 03 H-SHIFT PIC X(-).
002650 03 TODAY-DATE PIC X(6).
002670 03 TODAY-SHIFT PIC X(4).
002690 03 UPDATE-RECS OCCURS 6 TIMES.
002710 05 ORIG PIC X(6).
002730 05 DEST PIC X(6).
002750 05 SIZEH PIC XX.
002770 05 VEH PIC X(4).
002790 05 EMFRH PIC X.
002810 05 RPLEH PIC X.
002830 03 FILLER PIC X(122).
002930 FD VS2IN FILE CONTAINS 20 BY 1000 RECORDS
002950 RECCRD CONTAINS 80 CHARACTERS
002970 DATA RECCRD IS CARD-IMAGE.
002990 01 CARD-IMAGE PIC X(80).
003010 FD VS3IN FILE CONTAINS 20 BY 1000 RECORDS
003030 RECORD CONTAINS 80 CHARACTERS
003050 DATA RECCRD IS CARD-IMAGE2.
003070 01 CARD-IMAGE2 PIC X(80).
003090 WORKING-STORAGE SECTION.
003110 77 PROGRAM-NAME PIC X(6) VALUE "AVSIN1".
003130 77 ERR-CNT PIC 9999 COMP.
003150 77 LLAST PIC 9999 COMP VALUE ZEROES.
003170 77 JJ PIC 9999 COMP VALUE ZEROES.
003190 77 HIST-CHECK PIC X VALUE SPACE.
003210 77 STRAD-CAFAC PIC XX VALUE "05".
003230 77 TRANS-CAFAC PIC XX VALUE "13".
003250 77 TRAILER-CAPAC PIC XX VALUE "14".
003270 77 MAX-CAPAC PIC XX.
003290 77 EMER-CHECK PIC X VALUE SPACE.
003310 77 SAME-REG PIC X VALUE SPACE.

003330 77 J-DSP PIC 99.
003350 77 J PIC 9999 COMP.
003370 77 M PIC 99 COMP.
003390 77 K PIC 99 COMP.
003410 77 JLAST PIC 99 COMP.
003430 77 KLAST PIC 99 COMP.
003450 77 J-FLAG PIC 9 COMP.
003470 77 K-FLAG PIC 9 COMP.
003490 77 GROERS-FULL PIC 9 COMP.
003510 77 ONE PIC 9 COMP VALUE 012.
003530 77 KCHECK PIC 99 COMP.
003550 77 BASE-NR PIC 99.
003570 77 TYPE-CT PIC 99.
003590 77 TEST-NAME PIC X(6).
003610 77 TRUCK-TYPE PIC XX.
003630 77 OPTICN PIC X VALUE "2".
003650 01 SQSTOR COPY SQSTOR.
003670 01 I-WORK-AREA COPY ISTORE.
003690 01 AVS1-EXECUTE PIC X(40)
003710 VALUE "EX AVS1 CG 162904 .".
003730 01 AVS2-EXECUTE PIC X(40)
003750 VALUE "EX AVS2 CG 162904 .".
003770 01 AVS3-EXECUTE PIC X(40) VALUE
003790 "EX AVS3 CG 162904 .".
003810 01 TYPE-CT-CSP.
003830 03 FIRST-DIG PIC 9.
003850 03 SECOND-DIG PIC 9.
003870 01 ORDER-STORE.
003890 03 ORDER-OUTPUT OCCURS 99 TIMES.
003910 05 OUT-SIZE PIC XX.
003930 05 OUT-ORIG PIC X(6).
003950 05 OUT-NEST PIC X(6).
003970 01 TRUCK-STORE.
003990 03 TRUCK-OUTPUT OCCURS 50 TIMES.
003995 05 TRK-USE PIC X.
004010 05 TRUCK-NUM PIC XX.
004030 05 TRUCK-CAPAC PIC XX.
004050 05 TRUCK-TIME-LIM PIC XXX.
004070 01 OPT-STORE.
004090 03 M-OPTS PIC X.
004110 03 P-OPTS PIC X.
004130 01 FRAME-1-OUTPUT.
004150 03 LAST-OPD-PROC PIC XX.
004170 03 ERR-FIELD-1 OCCURS 10 TIMES.
004190 05 ORDER-NR PIC XX.
004210 05 EXPLANATION-1 PIC X(20).
004230 01 FRAME-2-OUTPUT.
004250 03 ERR-FIELD-2 OCCURS 8 TIMES.
004270 05 V-NAME.
004290 07 V-TYPE PIC XX.
004310 07 V-NUMB PIC X.
004330 05 EXPLANATION-2 PIC X(20).
004350 03 LAST-TRUCK-FRCC.
004370 05 LAST-TYPE PIC XX.
004390 05 LAST-NUMB PIC X.

004410 01 FRAME-3-OUTPUT.
004430 03 ERR-FIELD-3 OCCURS 6 TIMES.
004450 05 INPUT-ERROR PIC X(4).
004470 05 EXPLANATION-3 PIC X(20).
004490 01 FRAME-6-OUTPUT.
004510 03 OUT-MIX PIC X.
004530 03 OUT-CYCLIC PIC X.
004550 01 FRAME-7-OUTPUT.
004570 03 ERR-FIELD-4 OCCURS 6 TIMES.
004590 05 INPUT-ERR PIC X(4).
004610 05 EXPLANATION-4 PIC X(19).
004630 01 OPT-CARD.
004650 03 O-OPT PIC X.
004670 03 FILLER PIC X(74) VALUE SPACES.
004690 01 DATE-CARD.
004710 03 O-DATE PIC X(6).
004730 03 FILLER PIC X(74) VALUE SPACES.
004750 01 SHIFT-CARD.
004770 03 O-SHIFT PIC X(4) VALUE SPACES.
004790 03 FILLER PIC X(76) VALUE SPACES.
004810 01 OPT-CARD-E.
004830 03 O-MIX PIC X VALUE "0".
004850 03 FILLER PIC X VALUE SPACE.
004870 03 PROPT PIC X VALUE "0".
004890 03 FILLER PIC X(77) VALUE SPACES.
004910 01 ORDER-CARD.
004930 03 O-OSIZE PIC XX.
004950 03 FILLER PIC X VALUE SPACE.
004970 03 O-ORIGIN PIC X(6).
004990 03 FILLER PIC X VALUE SPACE.
005010 03 O-DESTIN PIC X(6).
005030 03 FILLER PIC X(64) VALUE SPACES.
005050 01 ORDER-CARD-E.
005070 03 O-TIME PIC X(4).
005090 03 FILLER PIC XX VALUE SPACES.
005110 03 O-SIZ PIC XX.
005130 03 FILLER PIC X VALUE SPACE.
005150 03 O-ORIGN PIC X(6).
005170 03 FILLER PIC X VALUE SPACE.
005190 03 O-DESTN PIC X(6).
005210 03 FILLER PIC X(64) VALUE SPACES.
005230 01 END-CARD.
005250 03 O-END-SYMBOL PIC XX VALUE "-1".
005270 03 FILLER PIC X(74) VALUE SPACES.
005290 01 TRUCK-HEADER-CARD-R.
005310 03 O-TRUCK-TITLE-R PIC X(16) VALUE "TRUCKS".
005330 03 FILLER PIC X VALUE SPACE.
005350 03 O-NUMB-STS-R PIC XX.
005370 03 FILLER PIC X VALUE SPACE.
005390 03 O-NUMB-TRS-R PIC XX.
005410 03 FILLER PIC X VALUE SPACE.
005430 03 O-NUMB-TTS-R PIC XX.
005450 03 FILLER PIC X VALUE SPACE.
005470 03 O-ALL-LIN-R PIC XXXX.
005490 03 FILLER PIC X(60) VALUE SPACES.

```

005510 01 TRUCK-HEADER-CARD.
005530 03 O-TRUCK-TITLE PIC X(6) VALUE "TRUCKS".
005550 03 FILLER PIC X VALUE SPACE.
005570 03 O-NUMB-STS PIC X.
005590 03 FILLER PIC X VALUE SPACE.
005610 03 O-NUMB-TNS PIC X.
005630 03 FILLER PIC X VALUE SPACE.
005650 03 C-NUMB-TTS PIC X.
005670 03 FILLER PIC X VALUE SPACE.
005690 03 O-ALL-LIM PIC XXXX.
005710 03 FILLER PIC X(60) VALUE SPACES.
005730 01 TRUCK-CARD.
005735 03 TRK-USE-IT PIC X.
005750 03 O-TRUCK-CODE PIC XX.
005770 03 FILLER PIC X VALUE SPACE.
005790 03 O-TRUCK-TIME PIC XXX.
005810 03 FILLER PIC X VALUE SPACE.
005830 03 O-TRUCK-CAPAC PIC XX.
005850 03 FILLER PIC X(70) VALUE SPACES.
005870 01 FCN-2-SAVE.
005890 03 FCN-2-PLCS OCCURS 6 TIMES.
005910 05 ORIG PIC X(6).
005930 05 DESTS PIC X(6).
005950 05 SIZE PIC XX.
005970 05 VEH PIC X(4).
005990 05 EMERS PIC X.
006010 05 RPLS PIC X.
01 AVS1-HEADER-CARD.
03 FILLER PIC X(5) VALUE "AVS1".
03 FILLER PIC X(75) VALUE SPACES.
01 AVS2-HEADER-CARD.
03 FILLER PIC X(5) VALUE "AVS2".
03 FILLER PIC X(75) VALUE SPACES.
006030 01 FCN-ALL-CARD.
006050 03 FCN PIC X.
006070 03 DATE1 PIC X(6) VALUE SPACES.
006090 03 DATE2 PIC X(6) VALUE SPACES.
006110 03 SHIFT-ALL PIC X(4) VALUE SPACES.
006130 03 OPTION-H PIC X VALUE SPACE.
006150 03 FILLER PIC X(67) VALUE SPACES.
006170 01 FCN-2-CARD.
006190 03 DATE-SAVE PIC X(6).
006210 03 SHIFT-SAVE PIC X(5).
006230 03 FILLER PIC X VALUE SPACE.
006250 03 VEH-NO PIC X(4).
006270 03 ORIG-SAVE PIC X(6).
006290 03 DEST-SAVE PIC X(6).
006310 03 FILLER PIC X(3) VALUE SPACES.
006330 03 SIZE-SAVE PIC XX.
006350 03 EMERG-SAVE PIC X.
006370 03 RPLS-SAVE PIC X.
006390 03 FILLER PIC X(45).
006410 01 WAREHOUSE-DATA.
006430 03 FILLER PIC X(114) VALUE
006450 "191 1601A 16013 1602 1603 1604 1605 1606 1621 1622

```

```

006470-   "46   53C   64E   64W   66E   66W   67E   67W   198   ".
006490   03 FILLER PIC X(114) VALUE
006510   "1127 113A SM   2     3     5     8     35    43C   44
006530-   "59   223   1502  1503  1507  80    177   1143  1199   ".
006550   03 FILLER PIC X(114) VALUE
006570   "187   21b   218   1013  1169  1171  1173  1172  1174  X10
006590-   "193   224   3C    43S   61    84    202   646   647   ".
006610   03 FILLER PIC X(102) VALUE
006630   "656   1     7     16    23    26    53S   X54   ABASE NWS
006650-   "DEYTN BRASH CSNWS NMEDC 1078 98    655   ".
006670   03 FILLER PIC X(4A) VALUE
006690   "L     M     N     P     Q     R     S     T     ".
006710 01 WAREHOUSE-NAME. REDEFINES WAREHOUSE-DATA.
006730   02 W-NAME, OCCURS 82 TIMES, PIC X(6).
006750 01 PATCH-AREA PIC X(200).
006770 PROCEDURE DIVISION.
006790 010-START-PROGRAM SECTION.
006810 011-START.
006830   PERFORM I-START.
006850 020-OPEN-FILE SECTION 51.
006870 021-OPEN.
006890   COPY SQCPIN.
006910 030-PROCESS SECTION.
006930 031-INITIALIZE.
006950   MOVE ZERO TO JLAST.
006970   MOVE ZERO TO KLAST.
006990   MOVE ZERO TO ORDERS-FULL.
006995   MOVE ZERO TO LLAST.
007010 040-READ.
007030   MOVE ONE TO K.
007050   MOVE "00" TO J-DSP.
007070   MOVE SPACES TO FRAME-1-INPUT.
007090   MOVE SPACES TO FRAME-2-INPUT.
007110   MOVE SPACES TO FRAME-3-INPUT.
007130   MOVE SPACES TO FRAME-6-INPUT.
007150   MOVE SPACES TO FRAME-7-INPUT.
007170   PERFORM SQREAD.
007190   IF FRAME-NR EQUALS "1" GO TO 100-FRAME-1.
007210   IF FRAME-NR EQUALS "2" GO TO 200-FRAME-2.
007230   IF FRAME-NR EQUALS "3" GO TO 300-FRAME-3.
007250   IF FRAME-NR EQUALS "4" GO TO 400-OUTPUT.
007270   IF FRAME-NR EQUALS "5" GO TO 600-EMERGENCY-INPUT.
007290   IF FRAME-NR EQUALS "6" GO TO 599-CLEAR-RESTART.
007310   IF FRAME-NR EQUALS "7" GO TO 700-HISTORY.
007330   GO TO 040-READ.
007350 100-FRAME-1.
007370   IF ORDERS-FULL EQUALS ONE MOVE "99" TO J-DSP
007390   GO TO 120-LAST-ORDER.
007410   MOVE ZERO TO K-FLAG.
007430   MOVE ONE TO J.
007450   MOVE "01" TO J-DSP.
007470   MOVE ONE TO K.
007490 110-NEXT-ORDER.
007510   MOVE ZERO TO J-FLAG.
007530   IF 0-SIZE (J) EQUALS SPACES

```

```

007550     IF O-ORIG (J) EQUALS SPACES
007570     IF C-DEST (J) EQUALS SPACES
007590     SUBTRACT 1 FROM J-DSP
007610     GO TO 120-LAST-ORDER.
007630     IF O-SIZE (J) NOT NUMERIC
007650     IF K-FLAG EQUALS ONE GO TO 120-LAST-ORDER
007670     ELSE MOVE J-DSP TO ORDER-NR (K)
007690     MOVE "SIZE NOT NUMERIC " TO EXPLANATION-1 (K)
007710     MOVE ONE TO J-FLAG
007730     ADD ONE TO K
007750     IF K GREATER THAN 10 MOVE ONE TO K-FLAG.
007770     MOVE O-ORIG (J) TO TEST-NAME.
007790     PERFORM 500-NAME-TEST.
007810     IF TEST-NAME NOT EQUAL O-ORIG (J)
007830     IF K-FLAG EQUALS ONE GO TO 120-LAST-ORDER
007850     ELSE MOVE J-DSP TO ORDER-NR (K)
007870     MOVE "ORIGIN MISMATCH" TO EXPLANATION-1 (K)
007890     MOVE ONE TO J-FLAG
007910     ADD ONE TO K
007930     IF K GREATER THAN 10 MOVE ONE TO K-FLAG.
007950     MOVE O-DEST (J) TO TEST-NAME.
007970     PERFORM 500-NAME-TEST.
007990     IF TEST-NAME NOT EQUAL O-DEST (J)
008010     IF K-FLAG EQUALS ONE GO TO 120-LAST-ORDER
008030     ELSE MOVE J-DSP TO ORDER-NR (K)
008050     MOVE "DESTINATION MISMATCH" TO EXPLANATION-1 (K)
008070     MOVE ONE TO J-FLAG
008090     ADD ONE TO K
008110     IF K GREATER THAN 10 MOVE ONE TO K-FLAG.
008130     IF J-FLAG NOT EQUAL ONE
008150     ADD ONE TO JLAST
008170     IF JLAST GREATER THAN 99 MOVE ONE TO ORDERS-FULL
008190     GO TO 120-LAST-ORDER
008210     ELSE MOVE O-SIZE (J) TO OUT-SIZE (JLAST)
008230     MOVE O-ORIG (J) TO OUT-ORIG (JLAST)
008250     MOVE O-DEST (J) TO OUT-DEST (JLAST).
008270     ADD ONE TO J.
008290     IF J GREATER THAN 20 GO TO 120-LAST-ORDER.
008310     ADD 1 TO J-DSP.
008330     GO TO 110-NEXT-ORDER.
008350 120-LAST-ORDER.
008370     MOVE J-DSP TO LAST-ORD-PROC.
008390     MOVE 226 TO SQFILE-RECORD-SIZE.
008410     MOVE 15 TO SQFILE-ROW.
008430 150-WRITE.
008450     COPY SQWRIT REPLACING SQWRIT001 BY SQFILE-PREFIX
008470     SQWRIT002 BY FRAME-1-OUTPUT.
008490     MOVE SPACES TO FRAME-1-OUTPUT.
008510     GO TO 040-READ.
008530 200-FRAME-2.
008550     MOVE ZERO TO K-FLAG.
008570     MOVE ONE TO J.
008590     MOVE "01" TO J-DSP.
008610     MOVE ONE TO K.
008630     MOVE "ST" TO TRUCK-TYPE.

```

```

008650      MOVE STRAD-CAPAC TO MAX-CAPAC.
008670 210-NEXT-TRUCK.
008690      MOVE ZERO TO J-FLAG.
008710      IF V-USE (J) EQUAL "S" MOVE "S" TO SAME-REG
008730          MOVE ZERO TO SECCNC-DIG
008750          GO TO 230-LAST-TRUCK.
008770      IF V-USE (J) EQUALS SPACES GO TO 220-INCREMENT-J.
008790 212-USE-TRUCK.
008810      IF CAPAC (J) EQUALS SPACES MOVE MAX-CAPAC
008830          TO CAPAC (J).
008850      IF CAPAC (J) NOT NUMERIC
008870          IF K-FLAG EQUALS ONE GO TO 230-LAST-TRUCK
008890              ELSE PERFORM 213-DISP-VEH-NUM
008910                  MOVE SECOND-DIG TO V-NUMB (K)
008930                  MOVE TRUCK-TYPE TO V-TYPE (K)
008950                  MOVE "CAPAC NOT NUMERIC" TO EXPLANATION-2 (K)
008970                  MOVE ONE TO J-FLAG
008990                  ADD ONE TO K
009010                      IF K GREATER THAN 8 MOVE ONE TO K-FLAG.
009030      IF CAPAC (J) NUMERIC
009050          IF CAPAC (J) GREATER THAN MAX-CAPAC
009070              IF K-FLAG EQUALS ONE GO TO 230-LAST-TRUCK
009090                  ELSE PERFORM 213-DISP-VEH-NUM
009110                      MOVE SECOND-DIG TO V-NUMB (K)
009130                      MOVE TRUCK-TYPE TO V-TYPE (K)
009150                      MOVE "CAPAC TOO LARGE" TO EXPLANATION-2 (K)
009170                      MOVE ONE TO J-FLAG
009190                      ADD ONE TO K
009210                          IF K GREATER THAN 8 MOVE ONE TO K-FLAG.
009230      IF TIME-LTM (J) EQUALS SPACES
009250          MOVE "240" TO TIME-LIM (J).
009270      IF TIME-LIM (J) NOT NUMERIC
009290          IF K-FLAG EQUALS ONE GO TO 230-LAST-TRUCK
009310              ELSE PERFORM 213-DISP-VEH-NUM
009330          MOVE SECOND-DIG TO V-NUMB (K)
009350          MOVE TRUCK-TYPE TO V-TYPE (K)
009370          MOVE " TIME NOT NUMERIC" TO EXPLANATION-2 (K)
009390          MOVE ONE TO J-FLAG
009410          ADD ONE TO K
009430              IF K GREATER THAN 8 MOVE ONE TO K-FLAG.
009450      IF J-FLAG EQUALS ONE GO TO 220-INCREMENT-J.
009470      MOVE ONE TO KCHECK.
009490 213-DISP-VEH-NUM.
009510      MOVE J-DSP TO BASE-NR.
009530      MOVE J-DSP TO TYPE-CT.
009550      IF J GREATER THAN 12 ADD 8 TO BASE-NR
009570          SUBTRACT 12 FROM TYPE-CT.
009590      IF J GREATER THAN 18 ADD 4 TO BASE-NR
009610          SUBTRACT 6 FROM TYPE-CT.
009630      MOVE TYPE-CT TO TYPE-CT-DSP.
009650 215-CHECK-LIST.
009670      IF KCHECK GREATER THAN KLAST GO TO 21A-NEW-TRUCK.
009690      IF TRUCK-NUM (KCHECK) NOT EQUAL BASE-NR
009710          ADD ONE TO KCHECK
009730          GO TO 215-CHECK-LIST.

```



```

009750 MOVE CAPAC (J) TO TRUCK-CAPAC (KCHECK).
009770 MOVE TIME-LIM (J) TO TRUCK-TIME-LIM (KCHECK).
009790 GO TO 220-INCREMENT-J.
009810 218-NEW-TRUCK.
009830 ADD ONE TO KLAST.
009850 MOVE BASE-NR TO TRUCK-NUM (KLAST).
009870 MOVE CAPAC (J) TO TRUCK-CAPAC (KLAST).
009890 MOVE TIME-LIM (J) TO TRUCK-TIME-LIM (KLAST).
009892 IF V-USE (J) EQUALS "*"
009894 MOVE "10" TO TRUCK-CAPAC (KLAST)
009896 MOVE "*" TO TRK-USE (KLAST)
009898 ELSE MOVE SPACE TO TRK-USE (KLAST).
009910 220-INCREMENT-J.
009930 ADD ONE TO J.
009950 ADD 1 TO J-DSP.
009970 IF J GREATER THAN 12
009990 MOVE TRANS-CAPAC TO MAX-CAPAC
010010 MOVE "TR" TO TRUCK-TYPE.
010030 IF J GREATER THAN 18
010050 MOVE TRAILER-CAPAC TO MAX-CAPAC
010070 MOVE "TT" TO TRUCK-TYPE.
010090 IF J GREATER THAN 30 GO TO 230-LAST-TRUCK.
010110 GO TO 210-NEXT-TRUCK.
010130 230-LAST-TRUCK.
010150 MOVE TRUCK-TYPE TO LAST-TYPE.
010170 MOVE SECOND-DIG TO LAST-NUMB.
010190 MOVE 191 TO SQFILE-RECORD-SIZE.
010210 MOVE 17 TO SQFILE-ROW.
010230 250-WRITE.
010250 COPY SQWRIT REPLACING SQWRIT001 BY SQFILE-PREFIX
010270 SQWRIT002 BY FRAME-2-OUTPUT.
010290 MOVE SPACES TO FRAME-2-OUTPUT.
010310 GO TO 040-READ.
010330 300-FRAME-3.
010350 MOVE ONE TO K.
010370 IF I-DATE NOT NUMERIC
010390 MOVE "DATE" TO INPUT-ERROR (K)
010410 MOVE "NOT NUMERIC" TO EXPLANATION-3 (K)
010430 ADD ONE TO K.
010450 IF START-TIME NOT NUMERIC
010470 MOVE "TIME" TO INPUT-ERROR (K)
010490 MOVE "NOT NUMERIC" TO EXPLANATION-3 (K)
010510 ADD ONE TO K.
010530 IF MAX-ROUTE EQUALS SPACES MOVE "2400" TO MAX-ROUTE.
010550 IF MAX-ROUTE NOT NUMERIC
010570 MOVE "MXRT" TO INPUT-ERROR (K)
010590 MOVE "NOT NUMERIC" TO EXPLANATION-3 (K)
010610 ADD ONE TO K.
010630 IF K EQUALS ONE MOVE I-DATE TO O-DATE
010650 MOVE START-TIME TO O-SHIFT
010670 MOVE OPTION TO O-OPT
010690 MOVE MAX-ROUTE TO C-ALL-LIM
010710 MOVE "NO ERRORS" TO EXPLANATION-3 (K).
010730 MOVE 96 TO SQFILE-RECORD-SIZE.
010750 MOVE 11 TO SQFILE-ROW.

```

```

010770 350-WRITE.
010790 COPY SQWRIT REPLACING SQWRIT001 BY SQFILE-PREFIX
010810 SQWRIT002 BY FRAME-3-OUTPUT.
010830 MOVE SPACES TO FRAME-3-OUTPUT.
010850 GO TO 040-READ.
010870 600-EMERGENCY-INPUT.
010890 MOVE "E" TO EMER-CHECK.
010910 MOVE "0" TO M-OPTS.
010930 IF M-OPTION EQUALS "Y"
010950 MOVE "1" TO M-OPTS.
010970 MOVE "J" TO P-OPTS.
010990 IF P-OPTION EQUALS "Y"
011010 MOVE "1" TO P-OPTS.
011030 MOVE P-OPTS TO OUT-OPTION.
011050 MOVE M-OPTS TO OUT-MIX.
011070 MOVE 96 TO SQFILE-RECORD-SIZE.
011090 MOVE 11 TO SQFILE-ROW.
011110 650-WRITE.
011130 COPY SQWRIT REPLACING SQWRIT001 BY SQFILE-PREFIX
011150 SQWRIT002 BY FRAME-6-OUTPUT.
011170 MOVE SPACES TO FRAME-6-OUTPUT.
011190 GO TO 040-READ.
011210 700-HISTORY.
011230 MOVE "H" TO HIST-CHECK
011250 MOVE FCN-TYPE TO FCN
011270 MOVE ZEROS TO ERR-CNT.
011290 IF FCN-TYPE EQUALS "2" GO TO 750-FUNCTION-2.
011291 IF FCN-TYPE EQUALS "1"
011292 MOVE ZEROS TO START-DATE
011293 MOVE ZEROS TO END-DATE
011294 MOVE ZEROS TO H-SHIFT.
011310 IF START-DATE NOT NUMERIC
011330 ADD ONE TO ERR-CNT
011370 TO EXPLANATION-> (ERR-CNT)
011390 MOVE ERR-CNT TO INPUT-ERR (ERR-CNT).
011410 IF END-DATE NOT NUMERIC
011430 ADD ONE TO ERR-CNT
011450 MOVE "END DATE NOT NUMERIC"
011490 MOVE ERR-CNT TO INPUT-ERR (ERR-CNT).
011491 IF START-DATE NOT= END-DATE
011492 MOVE ZERO TO H-SHIFT.
011510 IF H-SHIFT NOT NUMERIC
011530 ADD ONE TO ERR-CNT
011550 MOVE "SHIFT NOT NUMERIC"
011570 TO EXPLANATION-> (ERR-CNT)
011590 MOVE ERR-CNT TO INPUT-ERR (ERR-CNT)
011610 GO TO 799-DISPLAY-ERRORS.
011630 MOVE START-DATE TO DATE1
011650 MOVE END-DATE TO DATE2
011670 MOVE H-SHIFT TO SHIFT-ALL.
011690 IF H-OPTION EQUALS "Y"
011710 MOVE "1" TO OPTION-H ELSE
011730 MOVE "0" TO OPTION-H.
011750 GO TO 799-DISPLAY-ERRORS.
011770 750-FUNCTION-2.

```

```

011790 MOVE ZERCF5 TO JJ
011810 MOVE ZERCF5 TO ERR-CNT.
011830 IF TODAY-DATE NOT NUMERIC
011850 ADD ONE TO ERR-CNT
011870 MOVE "DATE NOT NUMERIC" TO EXPLANATION-4 (ERR-CNT)
011890 MOVE ERR-CNT TO INPUT-ERR (ERR-CNT).
011910 IF TODAY-SHIFT NOT NUMERIC
011930 ADD ONE TO ERR-CNT
011950 MOVE "SHIFT NOT NUMERIC" TO EXPLANATION-4 (ERR-CNT)
011970 MOVE ERR-CNT TO INPUT-ERR (ERR-CNT).
011990 IF ERR-CNT GREATER THAN ZERO GO TO 799-DISPLAY-ERRORS.
012010 MOVE TODAY-DATE TO DATE-SAVE
012030 MOVE TODAY-SHIFT TO SHIFT-SAVE.
012050 760-INCREMENT.
012055 IF LLAST EQUALS 6 GO TO 799-DISPLAY-ERRORS.
012070 ADD ONE TO JJ.
012090 IF JJ GREATER THAN 6
012110 GO TO 799-DISPLAY-ERRORS.
012115 IF CRIGH(JJ) EQUALS SPACES GO TO 799-DISPLAY-ERRORS.
012130 MOVE ORIGH (JJ) TO TEST-NAME.
012150 PERFORM 500-NAME-TEST.
012170 IF TEST-NAME NOT EQUAL ORIGH (JJ)
012190 ADD ONE TO ERR-CNT
012210 MOVE "ORIGIN MISMATCH" TO EXPLANATION-4 (ERR-CNT)
012230 MOVE JJ TO INPUT-ERR (ERR-CNT).
012250 IF ERR-CNT EQUALS 6 GO TO 799-DISPLAY-ERRORS.
012270 MOVE DESTH (JJ) TO TEST-NAME.
012290 PERFORM 500-NAME-TEST.
012310 IF TEST-NAME NOT EQUAL DESTH (JJ)
012330 ADD ONE TO ERR-CNT
012350 MOVE "DEST MISMATCH" TO EXPLANATION-4 (ERR-CNT)
012370 MOVE JJ TO INPUT-ERR (ERR-CNT).
012390 IF ERR-CNT EQUALS 6 GO TO 799-DISPLAY-ERRORS.
012410 IF SIZEH (JJ) NOT NUMERIC
012430 ADD ONE TO ERR-CNT
012450 MOVE "SIZE NOT NUMERIC" TO EXPLANATION-4 (ERR-CNT)
012470 MOVE JJ TO INPUT-ERR (ERR-CNT).
012490 IF ERR-CNT EQUALS 6 GO TO 799-DISPLAY-ERRORS.
012510 IF ERR-CNT GREATER THAN ZERO
012530 GO TO 760-INCREMENT.
012550 ADD ONE TO LLAST.
012570 MOVE ORIGH (JJ) TO ORIG5 (LLAST)
012590 MOVE DESTH (JJ) TO DEST5 (LLAST)
012610 MOVE SIZEH (JJ) TO SIZE5 (LLAST)
012630 MOVE VEHH (JJ) TO VEH5 (LLAST).
012650 IF EMERH (JJ) EQUALS "Y"
012670 MOVE "1" TO EMER5 (LLAST) ELSE
012690 MOVE "0" TO EMER5 (LLAST).
012710 IF KPLEH (JJ) EQUALS "Y"
012730 MOVE "1" TO KPLE5 (LLAST) ELSE
012750 MOVE "0" TO KPLE5 (LLAST).
012770 GO TO 760-INCREMENT.
012790 799-DISPLAY-ERRORS.
012810 IF ERR-CNT EQUALS ZEROES
012830 MOVE "NO ERRORS" TO EXPLANATION-4 (1)

```

```

012850      MOVE ZEROES TO INPUT-ERR (1).
012870      MOVE 226 TO SQFILE-RECORD-SIZE.
012890      MOVE 16 TO SQFILE-ROW.
012910 799-WRITE.
012930      COPY SQWRIT REPLACING SQWRIT001 BY SQFILE-PREFIX
012950      SQWRIT002 BY FRAME-7-OUTPUT.
012970      MOVE SPACES TO FRAME-7-OUTPUT.
012990      GO TO 040-READ.
013010 400-OUTPUT.
013030      IF HIST-CHECK EQUALS "H" GO TO 400-HISTORY-CARDS.
013050      IF EMER-CHECK NOT EQUAL "E" GO TO 400-OUTPUT-REGULAR.
013070      OPEN OUTPUT VS2IN.
013090      MOVE ONE TO J.
013110 410-ORDER-OUTPUT-E.
013130      MOVE O-SHIFT TO O-TIME.
013150      MOVE OUT-SIZE (J) TO O-SIZ.
013170      MOVE OUT-ORIG (J) TO O-ORIGN.
013190      MOVE OUT-DEST (J) TO O-DESTN.
013210      MOVE ORDER-CARD-E TO CARD-IMAGE.
013230      WRITE CARD-IMAGE.
013250      ADD ONE TO J.
013270      IF J GREATER THAN JLAST NEXT SENTENCE
013290          ELSE GO TO 410-ORDER-OUTPUT-E.
013310      MOVE END-CARD TO CARD-IMAGE.
013330      WRITE CARD-IMAGE.
013350      MOVE M-OPTS TO O-MIX.
013370      MOVE P-OPTS TO PROFIT.
013390      MOVE OPT-CARD-E TO CARD-IMAGE.
013410      WRITE CARD-IMAGE.
013430      MOVE ZERO TO O-NUMB-STS.
013450      MOVE ZERO TO O-NUMB-TRS.
013470      MOVE ZERO TO O-NUMB-TTS.
013490      IF SAME-REG EQUALS "S"
013510          MOVE "A" TO C-NUMB-STS.
013530      MOVE TRUCK-HEADER-CARD TO CARD-IMAGE.
013550      WRITE CARD-IMAGE.
013570      IF SAME-REG EQUALS "S"
013590          GO TO 421-END-CARD.
013610      MOVE ONE TO J.
013630 420-TRUCK-OUTPUT-E.
013635      MOVE TRK-USE (J) TO TRK-USE-IT
013650      MOVE TRUCK-NUM (J) TO O-TRUCK-CODE.
013670      MOVE TRUCK-TIME-LIM (J) TO O-TRUCK-TIME.
013690      MOVE SPACES TO O-TRUCK-CAPAC.
013710      MOVE TRUCK-CARD TO CARD-IMAGE.
013730      WRITE CARD-IMAGE.
013750      ADD ONE TO J.
013770      IF J GREATER THAN KLAST GO TO 421-END-CARD
013790          ELSE GO TO 420-TRUCK-OUTPUT-E.
421-END-CARD.
013870      CLOSE VS2IN WITH RELEASE.
013890      ZIP AVS2-EXECUTE.
013910      GO TO 040-READ.
013930 400-OUTPUT-REGULAR.
          OPEN OUTPUT VS2IN.

```

```

MOVE DATE-CARD TO CARD-IMAGE.
WRITE CARD-IMAGE.
MOVE SHIFT-CARD TO CARD-IMAGE.
WRITE CARD-IMAGE.
MOVE OPT-CARD TO CARD-IMAGE.
WRITE CARD-IMAGE.
014090 MOVE ONE TO J.
014110 410-ORDER-OUTPUT.
014130 MOVE OUT-SIZE (J) TO O-SIZE.
014150 MOVE OUT-ORIG (J) TO O-ORIG.
014170 MOVE OUT-DEST (J) TO O-DESTIN.
MOVE ORDER-CARD TO CARD-IMAGE.
WRITE CARD-IMAGE.
014230 ADD ONE TO J.
014250 IF J GREATER THAN JLAST NEXT SENTENCE
014270 ELSE GO TO 410-ORDER-OUTPUT.
MOVE END-CARD TO CARD-IMAGE.
WRITE CARD-IMAGE.
014330 MOVE ZEROES TO O-NUMB-STS-R
014350 MOVE ZEROES TO O-NUMB-TRS-R
014370 MOVE ZEROES TO O-NUMB-TTS-R
014390 MOVE O-ALL-LIM TO O-ALL-LIM-R.
MOVE TRUCK-HEADER-CARD-R TO CARD-IMAGE.
WRITE CARD-IMAGE.
014450 MOVE ONE TO J.
014470 420-TRUCK-OUTPUT.
014475 MOVE TRK-USE (J) TO TRK-USE-IT.
014490 MOVE TRUCK-NUM (J) TO O-TRUCK-CODE.
014510 MOVE TRUCK-TIME-LIM (J) TO O-TRUCK-TIME.
014530 MOVE TRUCK-CAPAC (J) TO C-TRUCK-CAPAC.
MOVE TRUCK-CARD TO CARD-IMAGE.
WRITE CARD-IMAGE.
014590 ADD ONE TO J.
014610 IF J GREATER THAN KLAST NEXT SENTENCE
014630 ELSE GO TO 420-TRUCK-OUTPUT.
CLOSE VS2IN WITH RELEASE.
ZIP AVS2-EXECUTE.
014730 GO TO 040-READ.
014750 400-HISTORY-CARDS.
014770 OPEN OUTPUT VS3IN.
014790 MOVE FCN-ALL-CARD TO CARD-IMAGE2.
014810 WRITE CARD-IMAGE2.
014830 IF FCN ACT EQUAL "2"
014850 GO TO 410-RUN-AVS3.
014870 MOVE ZERCS TO JJ.
014890 405-INCREMENT.
014910 ADD ONE TO JJ.
014930 IF JJ GREATER THAN LLAST
014950 GO TO 410-RUN-AVS3.
014970 MOVE ORIGS (JJ) TO ORIG-SAVE
014990 MOVE DESTS (JJ) TO DEST-SAVE
015010 MOVE SIZES (JJ) TO SIZE-SAVE
015030 MOVE EMERS (JJ) TO EMERG-SAVE
015050 MOVE RPLES (JJ) TO RPLE-SAVE.
015070 MOVE VEHs (JJ) TO VEH-NO.

```

```

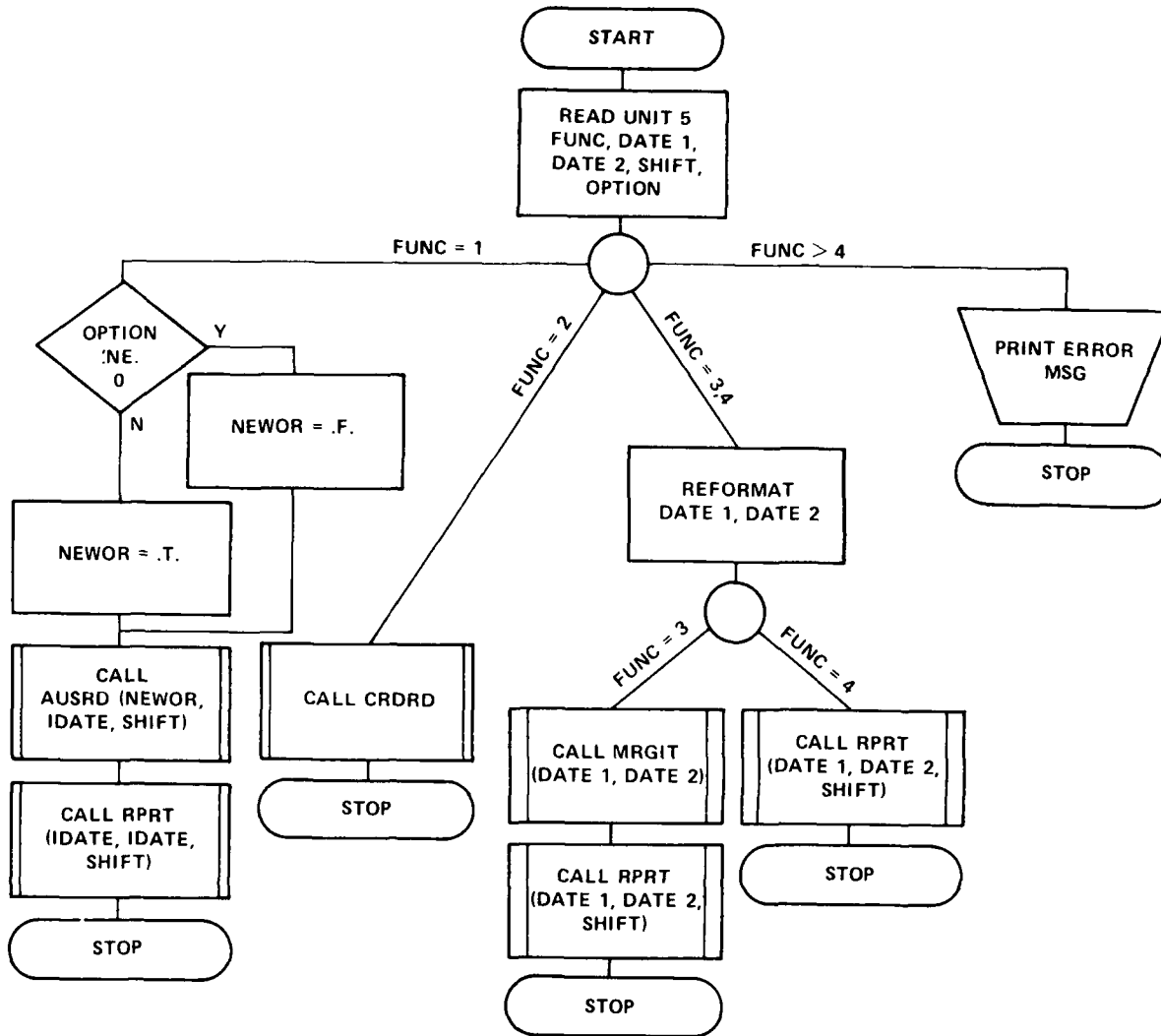
015090     MOVE FCA-2-CARD TO CARD-IMAGE2.
015110     WRITE CARD-IMAGE2.
015130     GC TO 405-INCREMMENT.
015150 410-RUN-AVS3.
015170     CLOSE VS3IN WITH RELEASE.
015190     ZIP AVS3-EXECUTE.
015210     GO TO 040-READ.
015230 500-NAMF-TEST SECTION.
015250 510-START.
015270     MOVE ONE TO M.
015290 520-LIST-LOOP.
015310     IF TEST-NAME EQUALS W-NAME (M) GC TO 530-EXIT.
015330     ADD ONE TO M.
015350     IF M GREATER THAN 82 .
015370         MOVE SPACES TO TEST-NAME.
015390     GO TO 530-EXIT.
015410     GO TO 520-LIST-LOOP.
015430 530-EXIT.
015450     EXIT.
015470 599-CLEAR-RESTART SECTION.
015490 599-START.
015510     MOVE SPACE TO SAME-REG
015530         MOVE SPACE TO EMER-CHECK.
015550     MOVE SPACE TO HIST-CHECK.
015570     MOVE ONE TO J.
015590     MOVE ONE TO K.
015591     MOVE ONE TO JJ.
015592 598-LOOP.
015593     MOVE SPACES TO ORIG(S(JJ))
015594     MOVE SPACES TO DESIS(JJ)
015595     MOVE SPACES TO SIZES(JJ)
015596     MOVE SPACES TO EMERS(JJ)
015597     MOVE SPACES TO RPLES(JJ)
015598     MOVE SPACES TO VEHS(JJ).
015599     ADD ONE TO JJ.
015600     IF JJ GREATER THAN 6 GO TO 599-LOOP ELSE GO TO 598-LOOP.
015610 599-LOOP.
015630     IF ORDER-OUTPUT (J) NOT EQUAL SPACES
015650         MOVE SPACES TO ORDER-OUTPUT (J).
015670     ADD ONE TO J.
015690     IF J GREATER THAN 99 GO TO 031-INITIALIZE.
015710     IF K GREATER THAN 50 GO TO 599-LOOP.
015730     IF TRUCK-OUTPUT (K) NOT EQUAL SPACES
015750         MOVE SPACES TO TRUCK-OUTPUT (K).
015770     ADD ONE TO K.
015790     GO TO 599-LOOP.
015810 CLOSE-FILE SECTION 54.
015830 05-CLOSE-IN.
015850     COPY SACLIN.
015870 06-SACLIN.
015890 06-CLOSE-OUT.
015910     GO TO I-EOJ.
015930 OPEN-SUBR SECTION 51.
015950 07-OPEN-SUBR.
015970     COPY IOPENR.

```

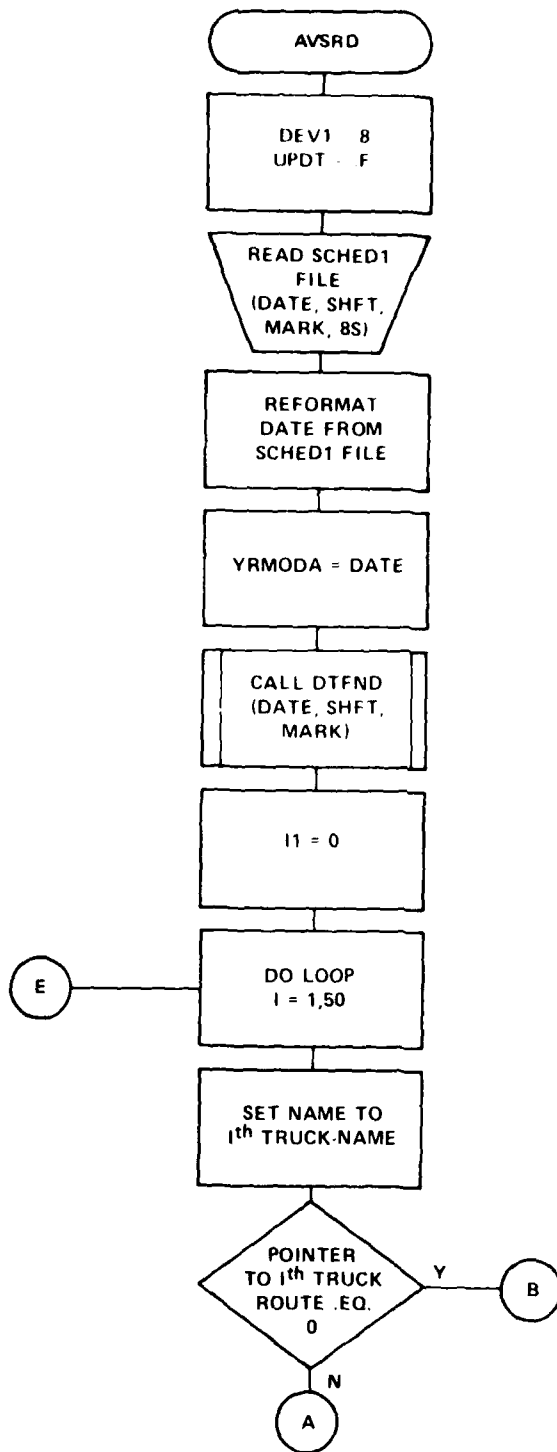
015990 BASE-SUBR SECTION.
016010 08-READ-SUBR.
016030 COPY SQRED1 REPLACING SQRED1002 BY 05-CLOSE-IN.
016050 09-WRITE-SUBR.
016070 COPY SQCLT1 REPLACING SQCLT1002 BY 15-OUTPUT-ERRCR.
016090 SNCH-SUBR SECTION 52.
016110 10-DCH-INPUT.
016130 COPY SQPEN2.
016150 11-DCH-OUTPUT.
016170 COPY SQOUT2.
016190 DUMMY-PARA SECTION.
016210 SQRED3.
016230 SQOUT3.
016250 CLOSE-SUBR SECTION 54.
016270 15-CLOSE-SUBR.
016290 COPY ICLCSR.
016310 ERR-PROC SECTION 59.
016330 16-OUTPUT-ERROR.
016350 MOVE "CLTPUT ERRCR" TO I-ABORT-MSG.
016370 TRACE 20.
016390 GO TO I-ABORT.
016410 COMMON-SUBR SECTION 59.
016430 17-COMMON-SUBR.
016450 COPY ICCOMN.
016470 END-OF-JOB.

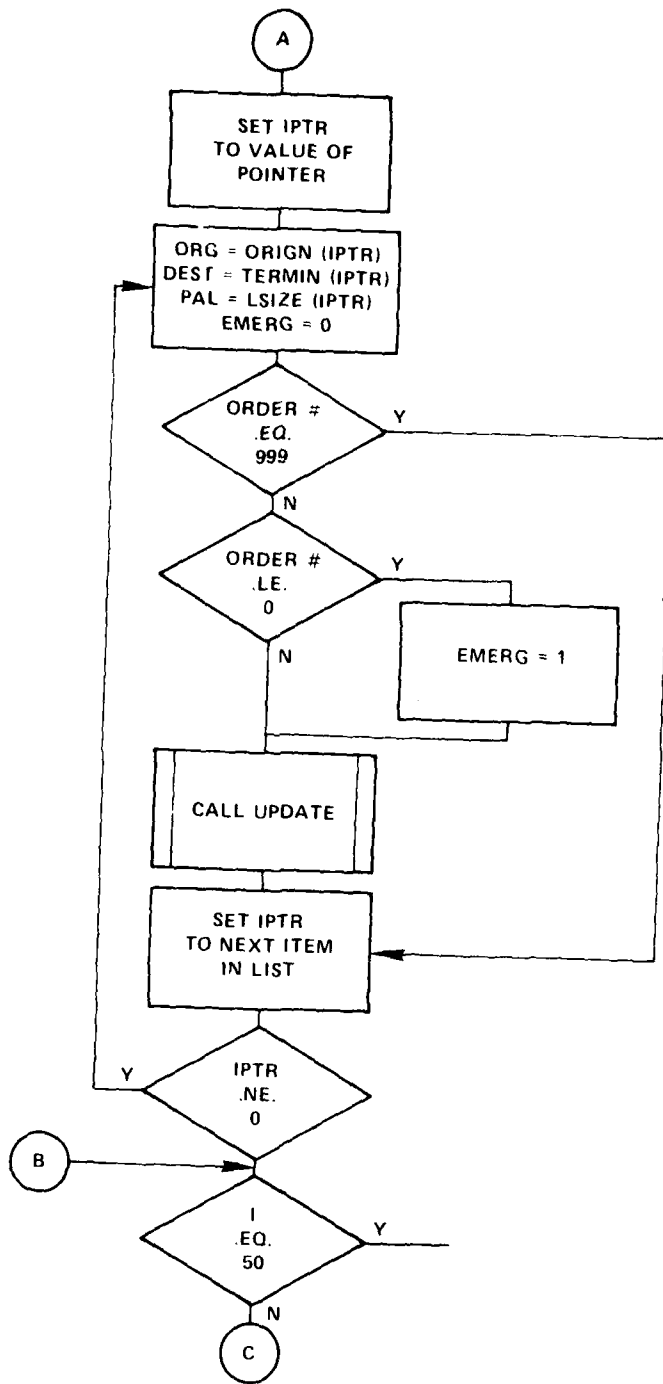
APPENDIX D - AVS 3, HISTORY/UPDATE, FLOWCHARTS AND LISTINGS

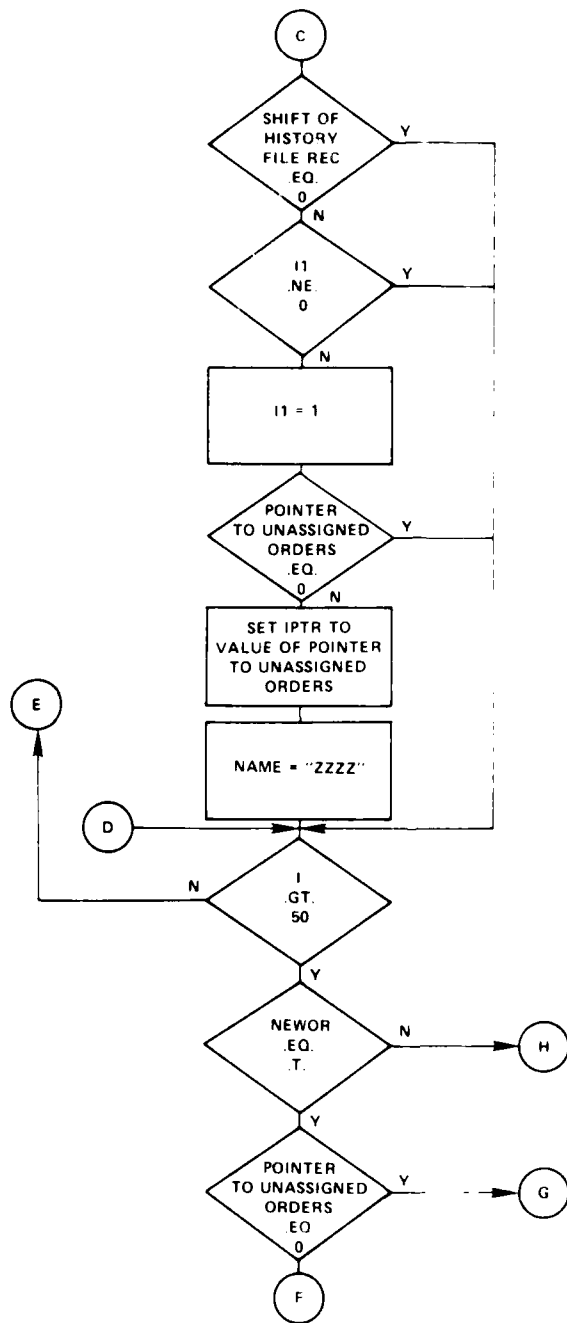
MAIN PROGRAM

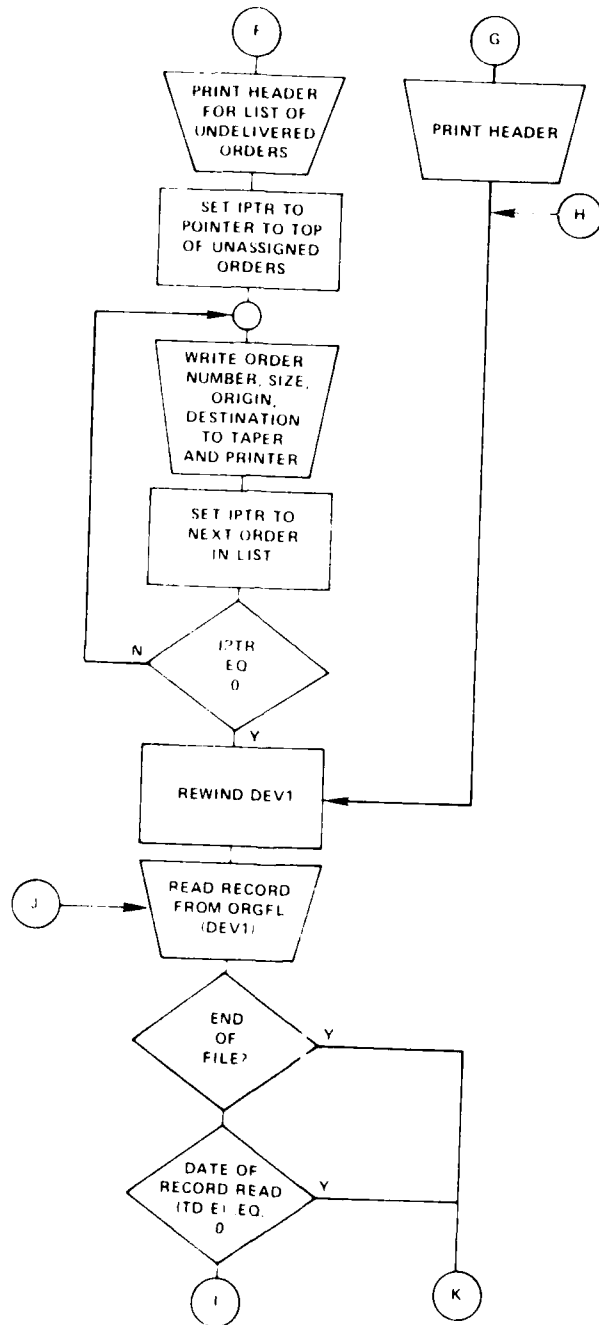


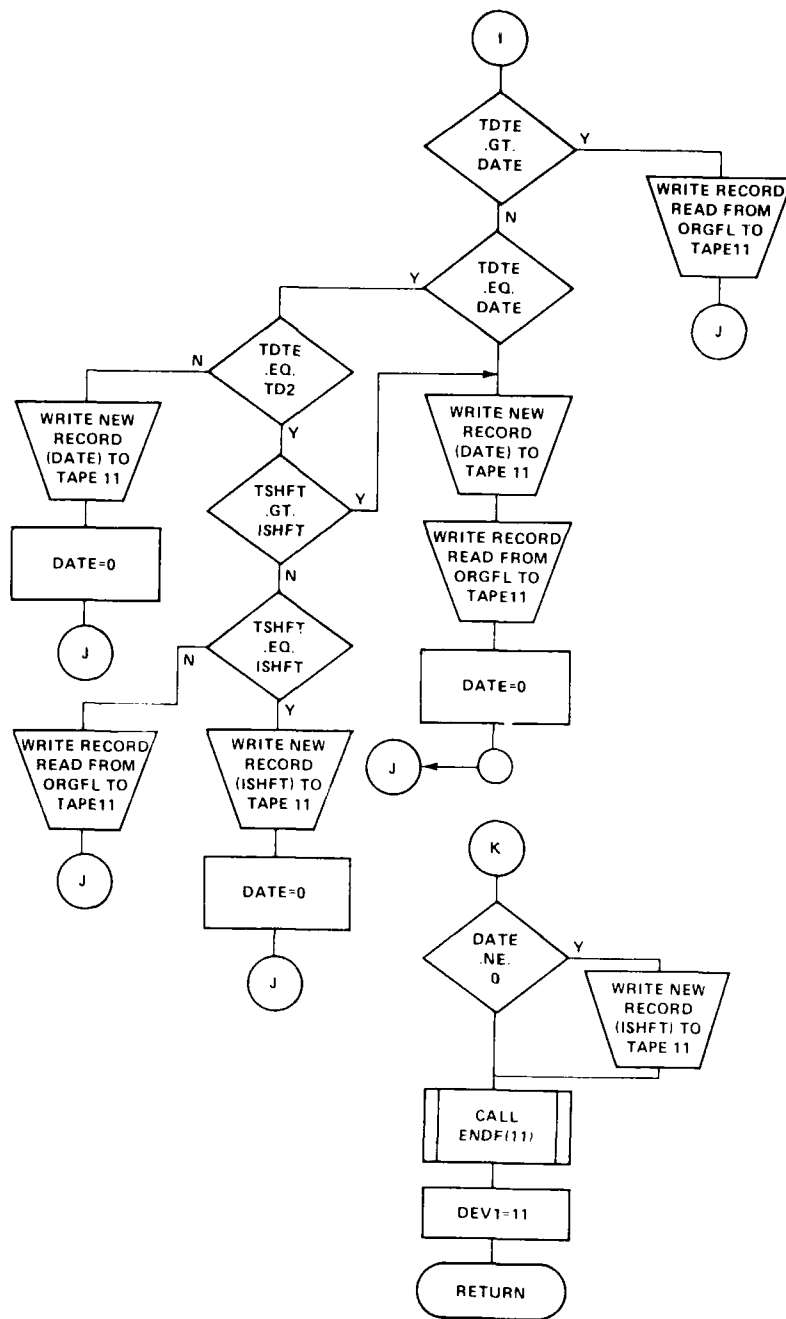
1	IDENT AVS3	AVS3	2
	SIZE INTEGER=7	AVS3	3
	SEGMENT AVSRD,MRGIT,MERGE,CROAD,RPLCL,DTFND,RPRT,DMFIT	AVS3	4
	FILE 7=SCHED1,UNIT=DISK,BLOCKING=1,RECORD=80	AVS3	5
2	FILE 8=ORGF1,UNIT=DISK,BLOCKING=1,RECORD=1424	AVS3	6
	FILE 9=TEMP,UNIT=DISK,BLOCKING=1,RECORD=1424	AVS3	7
	FILE 11=PPMF1E,UNIT=DISK,BLOCKING=1,RECORD=1+24,LOCK	AVS3	8
	FILE 12=AVIA,UNIT=DISK,BLOCKING=1,RECORD=83	AVS3	9
	FILE 5=VS4IN,UNIT=DISK,BLOCKING=1,RECORD=80	AVS3	10
10	REAL SHIFT	AVS3	11
	INTEGER DEV2	AVS3	12
	LOGICAL UPDT,UPDTE	AVS3	13
	INTEGER DATE1,DATE2,FUNC,OPTION,DAPRY(10),DEV1	AVS3	14
	LOGICAL NEWOR	AVS3	15
15	COMMON/CTRL/CFUT,CARRY,DEV1,DEV2,UPDTE	AVS3	16
	READ(5,1000)FUNC,DATE1,DATE2,INSHF,OPTION	AVS3	17
	1000 FORMAT(11,210,14,11)	AVS3	18
	SHIFT=FLCAT(INSHF)	AVS3	19
	GOTO(10,20,30,30,5,5,5,5,60),FUNC	AVS3	20
20	PRINT 1100	AVS3	21
	1100 FORMAT(" INVALID OPTION-PROGRAM TERMINATED")	AVS3	22
	STOP	AVS3	23
10	NEWOR=.F.	AVS3	24
	IF(OPTION.NE.C) NEWOR=.T.	AVS3	25
20	CALL AVSRD(NEWOR,ICATE,SHIFT)	AVS3	26
	CALL RPRT(IDATE,IDATE,SHIFT)	AVS3	27
	STOP	AVS3	28
20	CALL CTRD	AVS3	29
	STOP	AVS3	30
30	30 IDATE=DATE1/100	AVS3	31
	IYR=MOD(CATE1,100)	AVS3	32
	F1YR=FLOAT(IYR)	AVS3	33
	FDATE=FLOAT(IDATE)	AVS3	34
	FDATE1=FDATE+10000.*F1YR	AVS3	35
35	DATE1=IFIX(FDATE1)	AVS3	36
	IDATE=DATE2/100	AVS3	37
	IYR=MOD(CATE2,100)	AVS3	38
	F1YR=FLOAT(IYR)	AVS3	39
	FDATE=FLOAT(IDATE)	AVS3	40
40	FDATE1=FDATE+10000.*F1YR	AVS3	41
	DATE2=IFIX(FDATE1)	AVS3	42
	IF(DATE2.EQ.0)DATE2=DATE1	AVS3	43
	GOTO(40,40,40,50),FUNC	AVS3	44
40	CALL MRGIT(DATE1,DATE2)	AVS3	45
45	SHIFT=0.	AVS3	46
	CALL RPRT(DATE1,DATE2,SHIFT)	AVS3	47
	STOP	AVS3	48
50	DEV1=8	AVS3	49
	CALL RPRT(DATE1,DATE2,SHIFT)	AVS3	50
50	STOP	AVS3	51
	60 IF(.NOT.(DATE1.EQ.999999.AND.DATE2.EQ.999999))STOP	AVS3	52
	CALL ENDF(11)	AVS3	53
	PRINT 1200	AVS3	54
1200	FORMAT(" FILE DELETED")	AVS3	55
55	END	AVS3	56







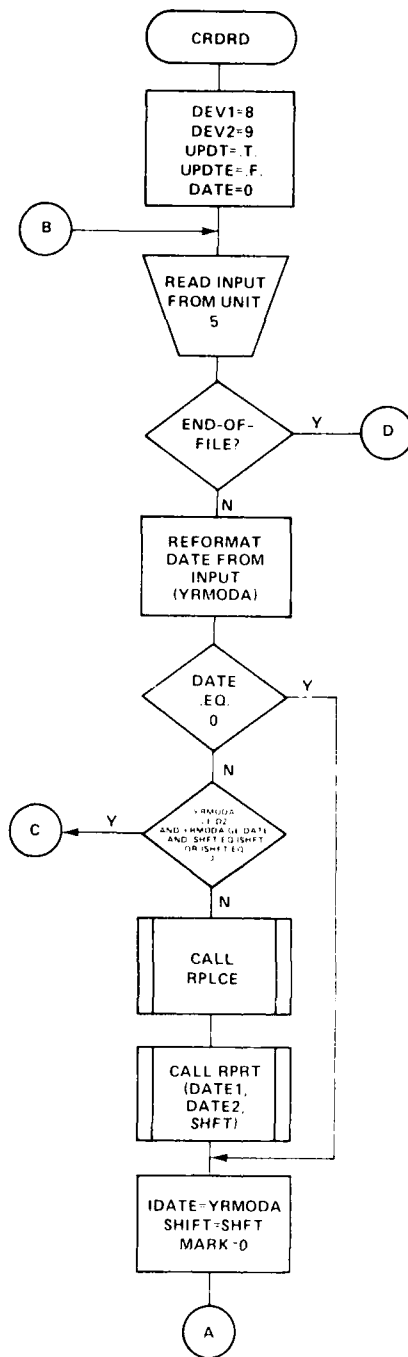


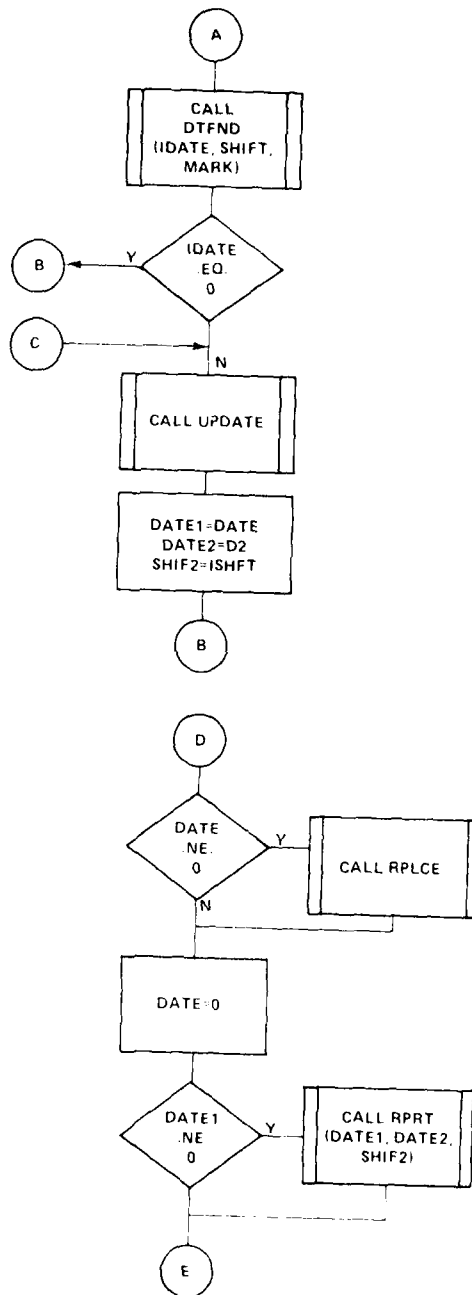


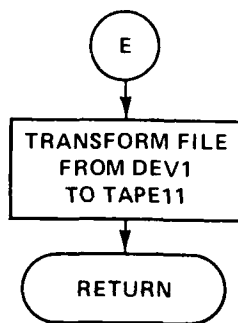
1			AVS3	57
		SUBROUTINE AVSRD(NEWOR, IDATE, SHIFT)	AVS3	58
	C	ALPHANUMERIC	AVS3	59
		ALPHA JUNK1	AVS3	60
5		ALPHA ORIGN(200), TERMN(200), TNAME(50)	AVS3	61
		* , WHNAM(92)	AVS3	62
	C	BINARY	AVS3	63
		INTEGER CNUMB(200), PTSOR(50), LFRWD(200), LSIZE(200), PSPUL(3)	AVS3	64
	C	ALPHANUMERIC	AVS3	65
10		ALPHA NAME, TRK, ORG, DEST, TINME(50), TORIG(50), TOSTN(50), INAME(50),	AVS3	66
		1 ORIG(50), OSTN(50)	AVS3	67
	C	BINARY	AVS3	68
		INTEGER YRMOVA, PAL, EMERG, TOTE, TREGP(50), TE PAL(50), DATE,	AVS3	69
		IRLGP(50), EPAL(50)	AVS3	70
15		LOGICAL UPOTE	AVS3	71
		INTEGER I MARK, T MARK, MARK, J2, T02	AVS3	72
		REAL SHFT, ISHFT, TSHFT	AVS3	73
		INTEGER DARRY(10), DEV1, DEV2	AVS3	74
		LOGICAL NEWOR, UPDT	AVS3	75
20		EXTERNAL DT FNC, MERGE	AVS3	76
		COMMON/AVSRC1/YRMCCA, PAL, EMERG, IFTN	AVS3	77
		COMMON/AVSML2/SHFT	AVS3	78
		COMMON/AVSRC3/NAME, TRK, ORG, DEST	AVS3	79
		COMMON/TMPRC1/TOTE, TD2, T MARK, TREGP, TE PAL	AVS3	80
25		COMMON/TMPRC2/ISHFT	AVS3	81
		COMMON/TMPRC3/TINME, TOFIG, TOSTN	AVS3	82
		COMMON/HISRC1/DATE, D2, I MARK, REGP, EPAL	AVS3	83
		COMMON/HISRC2/ISHFT	AVS3	84
		COMMON/HISRC3/INAME, ORIG, OSTN	AVS3	85
30		COMMON/CNTRL/UPDT, DARRY, DEV1, DEV2, UPOTE	AVS3	85
		DATA TNAME / 4HST1, 4HST2, 4HST3, 4HST4, 4HST5,	AVS3	87
		* 4HST6, 4HST7, 4HST8, 4HST9, 4HST10,	AVS3	88
		* 4HST11, 4HST12, 4HST13, 4HST14, 4HST15,	AVS3	89
		* 4HST16, 4HST17, 4HST18, 4HST19, 4HST20,	AVS3	90
35		* 4HTR1, 4HTR2, 4HTR3, 4HTR4, 4HTR5,	AVS3	91
		* 4HTR6, 4HTR7, 4HTR8, 4HTR9, 4HTR10,	AVS3	92
		* 4HTT1, 4HTT2, 4HTT3, 4HTT4, 4HTT5,	AVS3	93
		* 4HTT6, 4HTT7, 4HTT8, 4HTT9, 4HTT10,	AVS3	94
		* 4HIT1, 4HIT2, 4HIT3, 4HIT4, 4HIT5,	AVS3	95
40		* 4HIT6, 4HIT7, 4HIT8, 4HIT9, 4HIT10 /	AVS3	96
		DEV1=8	AVS3	97
		UPDT=.F.	AVS3	98
	C	READ DATA FROM AVS SHE01	AVS3	99
		READ(7, 1600) NHARE, (JUNK, I=1, 16), (WHNAM(I), I=1, NHARE)	AVS3	100
45		1600 FORMAT(10I8/7I8/(1X, 10A6))	AVS3	101
		DO 1001 I=1, 200	AVS3	102
		READ(7, 1000) CNUMB(I), INFO, IDESTP, IAREA, LFRWD(I)	AVS3	103
		1000 FORMAT(5I8, 36X)	AVS3	104
		IF (CNUMB(I).EQ.0) GO TO 1001	AVS3	105
50		ICLK=1	AVS3	106
		IF (INFO.GT.0) ICLK=-1	AVS3	107
		INFO=IABS(INFO)	AVS3	108
		IPTR=MOD(INFO/100, 100)	AVS3	109
		TERMN(I)=WHNAM(IPTR)	AVS3	110
55		IPTR=INFO/10000	AVS3	111
		ORIGN(I)=WHNAM(IPTR)	AVS3	112
		LSI7(I)=MOD(INFO, 100)*ICLK	AVS3	113

	1001 CONTINUE	AVS3	114
	READ(7,1100) (PTSOR(I),I=1,50)	AVS3	115
60	1100 FORMAT(I8,40X)	AVS3	116
	READ(7,1750) (FJUNK,I=1,141)	AVS3	117
	1750 FORMAT(3F6,1)	AVS3	118
	READ(7,1800) (JUNK,I=1,10)	AVS3	119
	1800 FORMAT(4I8/4I8/2I8)	AVS3	120
65	READ(7,1300) SHFT,IDATE	AVS3	121
	1300 FORMAT(F6.1,I8)	AVS3	122
	READ(7,1400) MARK	AVS3	123
	1400 FORMAT(I6)	AVS3	124
	C ORGANIZE DATA IN FORMAT DATE,TRUCKNAME,ORIGIN,DESTINATION,LOGSI7	AVS3	125
70	C EMERGENCY (=1 EMERGENCY,=0 REGULAR)	AVS3	126
	IYR=MOD(IDATE,100)	AVS3	127
	IDATE2=IDATE/100	AVS3	128
	FIYR=FLOAT(IYR)	AVS3	129
	FDATE=FLGAT(IDATE2)	AVS3	130
75	FDTE2=FDATE+1000.*FIYR	AVS3	131
	IDATE=IFIX(FDTE2)	AVS3	132
	YRMODA=IDATE	AVS3	133
	CALL DTFND(IDATE,SHFT,MARK)	AVS3	134
	DO 200 I=1,50	AVS3	135
80	NAME=TNAME(I)	AVS3	136
	IF(PTSOR(I).EQ.0) GO TO 200	AVS3	137
	C GET FIRST ELEMENT IN CHAIN	AVS3	138
	IPTR=PTSOR(I)	AVS3	139
85	20 ORG=ORIGIN(IPTR)	AVS3	140
	DEST=TERMN(IPTR)	AVS3	141
	PAL=LSIZE(IPTR)	AVS3	142
	EMERG=0	AVS3	143
	IF(ONUMB(IPTR).EQ.999)GOTO 185	AVS3	144
	IF(ONUMB(IPTR).LE.0) EMERG=1	AVS3	145
90	C CALL UPDATING ROUTINE-VARIABLES PASSED THROUGH AVSR0(IN COMMON)	AVS3	146
	CALL UPDATE	AVS3	147
	185 IPTR=LFIND(IPTR)	AVS3	148
	IF(IPTR.NE.0) GOTO 20	AVS3	149
	200 CONTINUE	AVS3	150
95	IF(.NOT.NEWR) GOTO 30	AVS3	151
	WRITE(6,3010)	AVS3	152
	3010 FORMAT(1P1," THE FOLLOWING UNDELIVERED ORDERS WILL BE INCLUDED",	AVS3	153
	1," IN THE INPUT FOR THE NEXT AVS RUN")	AVS3	154
	WRITE(6,3015)	AVS3	155
100	3015 FORMAT(//,1H,"ORDER NO",2X,"LOT SIZE",4X,"TO")	AVS3	156
	ICMK=0	AVS3	157
	DO 250 IPTR=1,200	AVS3	158
	IF(LSIZE(IPTR).GE.0) GO TO 250	AVS3	159
	ICMK=1	AVS3	160
105	LSIZE(IPTR)=IABS(LSIZE(IPTR))	AVS3	161
	IF(ISHFT.EQ.0.0) GO TO 25	AVS3	162
	IF(ONUMB(IPTR).EQ.999) GO TO 25	AVS3	163
	EMERG=0	AVS3	164
	IF(ONUMB(IPTR).LT.0) EMERG=1	AVS3	165
110	NAME=4HZZZZ	AVS3	166
	ORG=ORIGN(IPTR)	AVS3	167
	DEST=TERMN(IPTR)	AVS3	168
	PAL=LSIZE(IPTR)	AVS3	169
	CALL UPDATE	AVS3	170

115	25	WRITE(12,3000)ONUMB(IPTR),LSIZE(IPTR),ORIGN(IPTR),TERMN(IPTR)	AVS3	171
	3000	FORMAT(2I5,2A6)	AVS3	172
		WRITE(16,3005)ONUMB(IPTR),LSIZE(IPTR),ORIGN(IPTR),TERMN(IPTR)	AVS3	173
	3005	FORMAT(1H,15,5X,15,6X,A6,2X,A6)	AVS3	174
	250	CONTINUE	AVS3	175
120		IF(ICMK.EQ.0) GO TO 28	AVS3	176
		GOTO 30	AVS3	177
	28	WRITE(16,3020)	AVS3	178
	3020	FORMAT(" ALL ORDERS SCHEDULED FOR DELIVERY")	AVS3	179
	C END	AVS INPUT	AVS3	180
125	30	REWIND DEV1	AVS3	181
		SHIFT=ISHFT	AVS3	182
	35	READ(DEV1,1500,END=100)TOTE,TD2,TSHFT,TPARK,TINME,TORIG,TDSTN,	AVS3	183
	1	TREGP,TEPAL	AVS3	184
	1500	FORMAT(2I6,F6.1,I6,50A4,100A6,100I6)	AVS3	185
130		IF(TOTE.EQ.0) GOTO 100	AVS3	186
		IF(TOTE.GT.DATE)GOTO 80	AVS3	187
		IF(TOTE.EQ.DATE) GOTO 40	AVS3	188
	38	WRITE(11,1500)DATE,02,ISHFT,TPARK,INAME,ORIG,TDSTN,REGP,EPAL	AVS3	189
		WRITE(11,1500)TOTE,TD2,TSHFT,TPARK,TINME,TORIG,TDSTN,TREGP,TEPAL	AVS3	190
135		DATE=0	AVS3	191
		GOTO 35	AVS3	192
	40	IF(TOTE.EQ.TD2)GOTO 50	AVS3	193
		WRITE(11,1500)DATE,02,ISHFT,TPARK,INAME,ORIG,TDSTN,REGP,EPAL	AVS3	194
		DATE=0	AVS3	195
140		GOTO 35	AVS3	196
	50	IF(TSHFT.GT.ISHFT)GOTO 38	AVS3	197
		IF(TSHFT.EQ.ISHFT) GOTO 55	AVS3	198
		GOTO 80	AVS3	199
	55	WRITE(11,1500)DATE,02,ISHFT,TPARK,INAME,ORIG,TDSTN,REGP,EPAL	AVS3	200
145		DATE=0	AVS3	201
		GOTO 35	AVS3	202
	80	WRITE(11,1500)TOTE,TD2,TSHFT,TPARK,TINME,TORIG,TDSTN,TREGP,TEPAL	AVS3	203
		GOTO 35	AVS3	204
	100	IF(DATE.NE.0)WRITE(11,1500)DATE,02,ISHFT,TPARK,INAME,ORIG,	AVS3	205
150	1	TDSTN,REGP,EPAL	AVS3	206
		CALL ENDF(11)	AVS3	207
		DEV1=11	AVS3	208
		RETURN	AVS3	209
	5000	CALL ERROR(200)	AVS3	210
155		STOP	AVS3	211
		END	AVS3	212



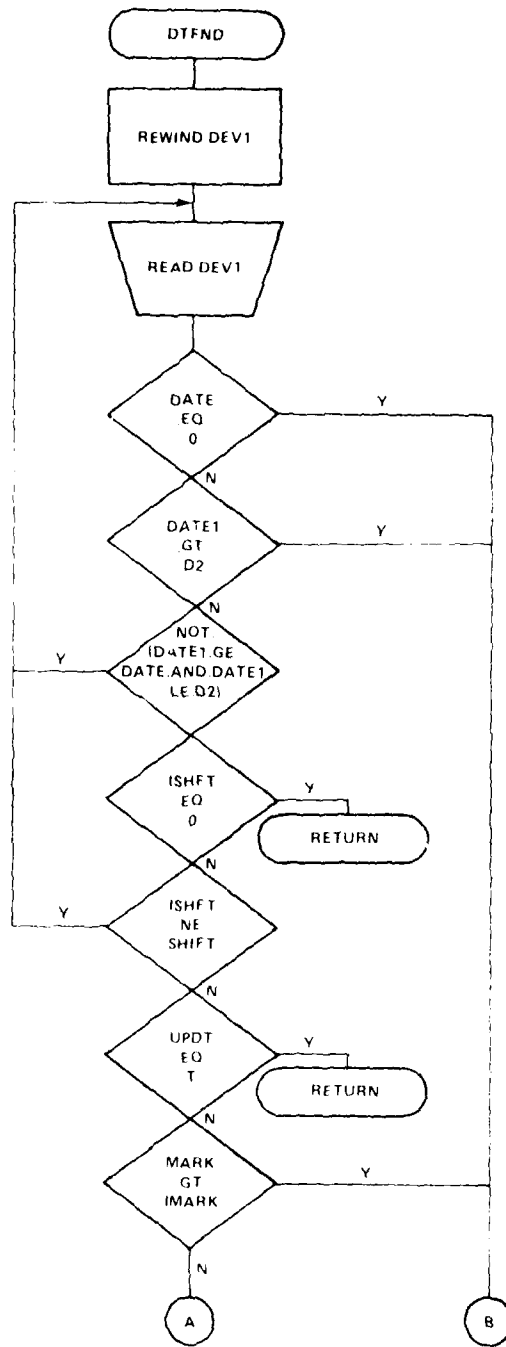


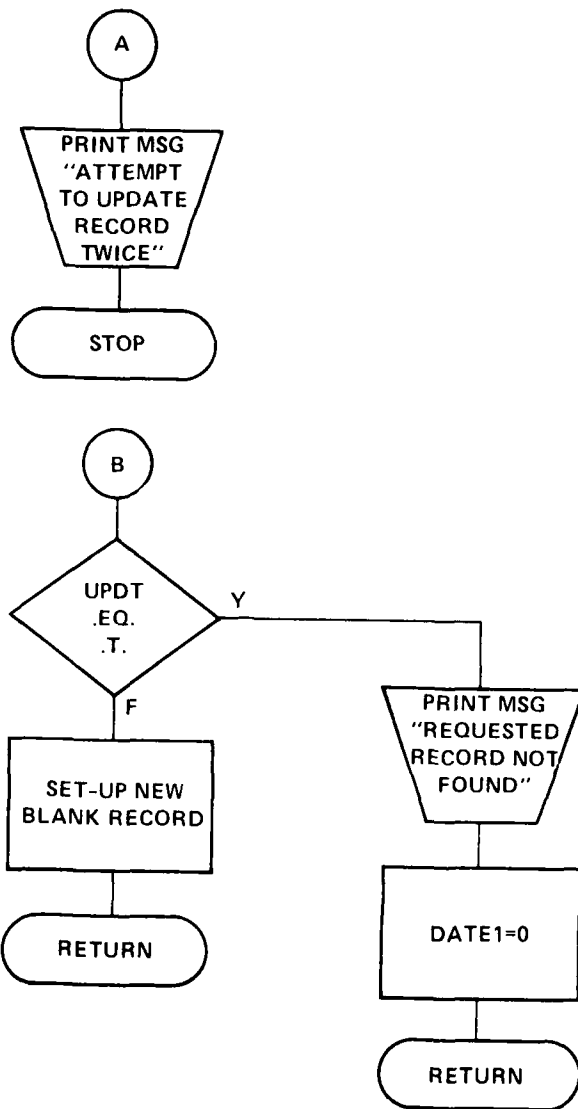


1	SUBROUTINE CRDRD	CRDRD	2
	C ALPHANUMERIC	CRDRD	3
	ALPHA NAME,TRK,ORG,DEST,TINME(50),TORIG(50),TOSTN(50),INAME(50),	CRDRD	4
	1 ORIG(50),DSTN(50)	CRDRD	5
5	C BINARY	CRDRD	6
	INTEGER YRMOA,PAL,EMERG,TOTE,TREGP(50),TEPAL(50),DATE,	CRDRD	7
	1REGP(50),EPAL(50)	CRDRD	8
	INTEGER IMARK,TMARK,MARK,D2,TC2,DEV1,DEV2,DATE1,DATE2,DARRY(10)	CRDRD	9
	REAL SHFT,ISHFT,TSHFT,SHIF2	CRDRD	10
10	LOGICAL UPDTE	CRDRD	11
	LOGICAL UPDT	CRDRD	12
	COMMON/4VSRG1/YRMOA,PAL,EMERG,IFTN	CRDRD	13
	COMMON/4VSRG2/SHFT	CRDRD	14
	COMMON/4VSRG3/NAME,TRK,ORG,DEST	CRDRD	15
15	COMMON/IMPRG1/TOTE,TO2,TMARK,TREGP,TEPAL	CRDRD	16
	COMMON/IMPRG2/ISHFT	CRDRD	17
	COMMON/IMPRG3/TINME,TORIG,TOSTN	CRDRD	18
	COMMON/HISRC1/DATE,D2,IMARK,REGP,EPAL	CRDRD	19
	COMMON/HISRC2/ISHFT	CRDRD	20
20	COMMON/HISRC3/INAME,ORIG,DSTN	CRDRD	21
	COMMON/DNTAL/UPDT,DARRY,DEV1,DEV2,UPDTE	CRDRD	22
	DEV1=0	CRDRD	23
	DEV2=9	CRDRD	24
	UPDT=.T.	CRDRD	25
25	UPDTE=.F.	CRDRD	26
	DATE=0	CRDRD	27
3	READ(5,1000,END=70)YRMOA,SHFT,NAME,ORG,DEST,PAL,EMERG,IFTN	CRDRD	28
1000	FORMAT(I6,F0.1,A4,2A6,I5,2I1)	CRDRD	29
	IYR=MOD(YRMOA,100)	CRDRD	30
30	IJATE=YRMOA/100	CRDRD	31
	FIYR=FLOAT(IYR)	CRDRD	32
	FJATE=FLOAT(IJATE)	CRDRD	33
	FJATE2=FJATE*1000.*FIYR	CRDRD	34
	YRMOA=FIX(FJATE2)	CRDRD	35
35	IF(DATE.EQ.0) GOTO 45	CRDRD	36
	IF(YRMOA.LE.D2.AND.YRMOA.GE.DATE.AND.	CRDRD	37
	1(SHFT.EQ.ISHFT.OR.ISHFT.EQ.0)) GOTO 60	CRDRD	38
	CALL RPLCE	CRDRD	39
	CALL RPRT(1,DATE1,DATE2,SHIF2)	CRDRD	40
40	45 IJATE=YRMOA	CRDRD	41
	SHIF=SHFT	CRDRD	42
	MARK=0	CRDRD	43
	CALL DTFN(IGATE,SHIF,MARK)	CRDRD	44
	IF(IJATE.EQ.0) GOTO 3	CRDRD	45
45	DATE1=DATE	CRDRD	46
	DATE2=D2	CRDRD	47
	SHIF2=ISHFT	CRDRD	48
50	CALL UPJATE	CRDRD	49
	DATE1=DATE	CRDRD	50
50	DATE2=D2	CRDRD	51
	SHIF2=ISHFT	CRDRD	52
	GOTO 3	CRDRD	53
70	IF(DATE.NE.0) CALL RPLCE	CRDRD	54
	DATE=0	CRDRD	55
55	IF(DATE1.NE.0) CALL RPRT(1,DATE1,DATE2,SHIF2)	CRDRD	56
	REWIND DEVI	CRDRD	57
75	READ(DEV1,1500,END=5000)TOTE,TO2,ISHFT,TMARK,TINME,TORIG,TOSTN,	CRDRD	58
	1TREGP,TEPAL	CRDRD	59
	IF(TOTE.EQ.0) GOTO 5000	CRDRD	60
1500	FORMAT(I6,F0.1,I5,50A4,10A6,100I1)	CRDRD	61
	WRITE(11,1500)TOTE,TO2,ISHFT,TMARK,TINME,TORIG,TOSTN,TREGP,TEPAL	CRDRD	62
	GOTO 75	CRDRD	63
5000	CALL ENDF(11)	CRDRD	64
	RETURN	CRDRD	65
60	END	CRDRD	66

1	SUBROUTINE DMPIT	FRR	8
	C ALPHA	FRR	9
	ALPHA INAME(50),ORIG(50),DSTN(50)	FRR	10
	C BINARY	FRR	11
5	INTEGER DZ	FRR	12
	INTEGER DATE,REGP(50),EPAL(50)	FRR	13
	REAL ISHFT	FRR	14
	COMMON/HISRC1/DATE,DZ,IMARK,REGP,EPAL	FRR	15
	COMMON/HISRC2/ISHFT	FRR	16
10	COMMON/HISRC3/INAME,ORIG,DSTN	FRR	17
	DATE=0	FRR	18
	REWIND 11	FRR	19
	10 READ(11,1000,END=20)DATE,DZ,ISHFT,IMARK,INAME,ORIG,DSTN,REGP,EPAL	FRR	20
	1000 FORMAT(2I6,F6.1,I6,50A4,100A6,100I6)	FRR	21
15	IF(DATE.EQ.0) GOTC 20	FRR	22
	PRINT 1200,DATE,ISHFT	FRR	23
	1200 FORMAT('1',I6,F6.1)	FRR	24
	DO 15 I=1,50	FRR	25
	PRINT 1500,INAME(I),ORIG(I),DSTN(I),REGP(I),EPAL(I)	FRR	26
20	1500 FORMAT(2M ,A4,2M ,A6,2M ,A6,2M ,I6,2M ,I6)	FRR	27
	15 CONTINUE	FRR	28
	GOTO 10	FRR	29
	20 PRINT 2900	FRR	30
	2900 FORMAT(' END OF FILE')	FRR	31
25	RETURN	FRR	32
	END	FRR	33

1	SUBROUTINE DATRAN(DATEIN,IDA,IYR,MONTH)	DTRN	2
	C ALPHA	DTRN	3
	ALPHA MOTBLE(12),MONTH	DTRN	4
	C BINARY	DTRN	5
5	INTEGER DATEIN,IDA,IYR	DTRN	6
	DATA MOTBLE/31JAN,31FEB,31MAR,31APR,31MAY,31JUN,31JUL,31AUG,	DTRN	7
	1 31SEP,31OCT,31NOV,31DEC/	DTRN	8
	IYR=DATEIN/10000	DTRN	9
	IMO=(DATEIN-10000*IYR)/100	DTRN	10
10	IDA=DATEIN-10000*IYR-100*IMO	DTRN	11
	MONTH=MOTBLE(IMO)	DTRN	12
	RETURN	DTRN	13
	END	DTRN	14





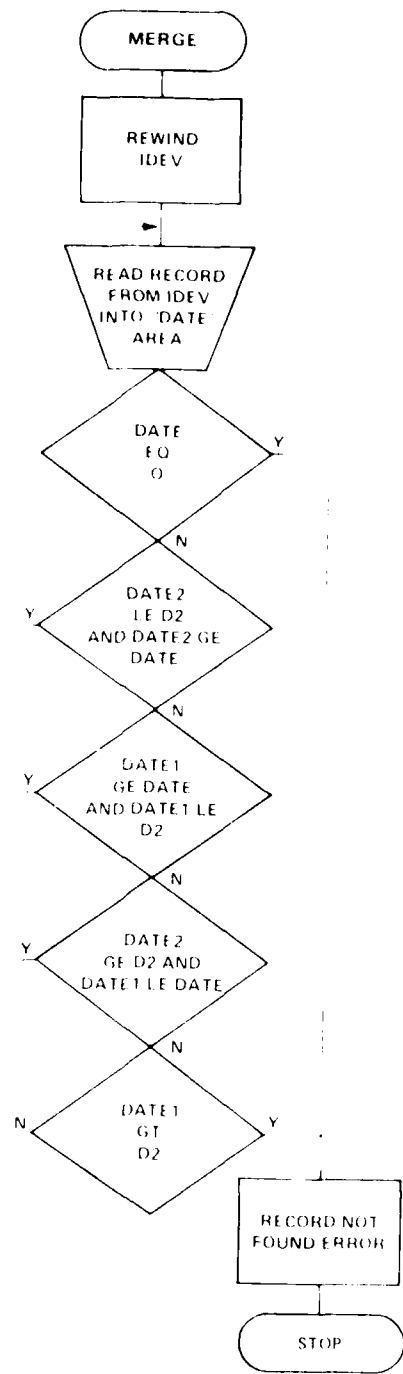
1	SUBROUTINE DTFND(DATE1, SHIFT, MARK)	DTFND	2
	C ALPHANUMERIC	DTFND	3
	ALPHA INAME(50), ORIG(50), DSTN(50)	DTFND	4
	C BINARY	DTFND	5
5	INTEGER DATE, CARRY(10), JSAVE, REGP(50), EPAL(50),	DTFND	6
	LOGV1, DEV2, DATE1, D2, IMARK	DTFND	7
	REAL SHIFT, ISHFT	DTFND	8
	LOGICAL UPOTE, UPDT	DTFND	9
	COMMON/HISRC1/CATE, D2, IMARK, REGP, EPAL	DTFND	10
10	COMMON/HISRC2/ISHFT	DTFND	11
	COMMON/HISRC3/INAME, ORIG, DSTN	DTFND	12
	COMMON/CNTRL/LFDT, CARRY, DEV1, DEV2, UPOTE	DTFND	13
	REWIND DEV1	DTFND	14
10	READ(DEV1, 1000, END=20) DATE, D2, ISHFT, IMARK, INAME, ORIG, DSTN, REGP,	DTFND	15
15	EPAL	DTFND	16
	1000 FORMAT(2I6, F6.1, I6, 50A4, 100A6, 100I6)	DTFND	17
	IF(DATE.EQ.0) GOTO 20	DTFND	18
	IF(DATE1.GT.02) GOTO 20	DTFND	19
20	IF(.NOT. (DATE1.GE.DATE.AND. DATE1.LE.02)) GOTO 10	DTFND	20
	IF(ISHFT.EQ.0) RETURN	DTFND	21
	IF(ISHFT.NE.SHIFT) GOTO 10	DTFND	22
	IF(UPDT) RETURN	DTFND	23
	IF(MARK.GT.IMARK) GOTO 20	DTFND	24
	PRINT 2000	DTFND	25
25	2000 FORMAT(" ATTEMPT TO UPDATE RECORD TWICE WITH SAME DATA",	DTFND	26
	1 /, " PROGRAM ABORTED")	DTFND	27
	STOP	DTFND	28
20	IF(UPDT) GOTO 40	DTFND	29
	DATE=DATE1	DTFND	30
30	D2=DATE1	DTFND	31
	ISHFT=SHIFT	DTFND	32
	IMARK=MARK	DTFND	33
	DO 30 J=1, 50	DTFND	34
	INAME(J)=10H	DTFND	35
35	ORIG(J)=10H	DTFND	36
	DSTN(J)=10H	DTFND	37
	REGP(J)=0	DTFND	38
30	EPAL(J)=0	DTFND	39
	RETURN	DTFND	40
40	40 PRINT 3000, DATE1, SHIFT	DTFND	41
	3000 FORMAT(" NO RECORD FOUND IN HISTORY FILE FOR ENTRY WITH DATE "	DTFND	42
	1 /, " AND SHIFT ", F6.1)	DTFND	43
	DATE1=J	DTFND	44
	RETURN	DTFND	45
45	END	DTFND	46

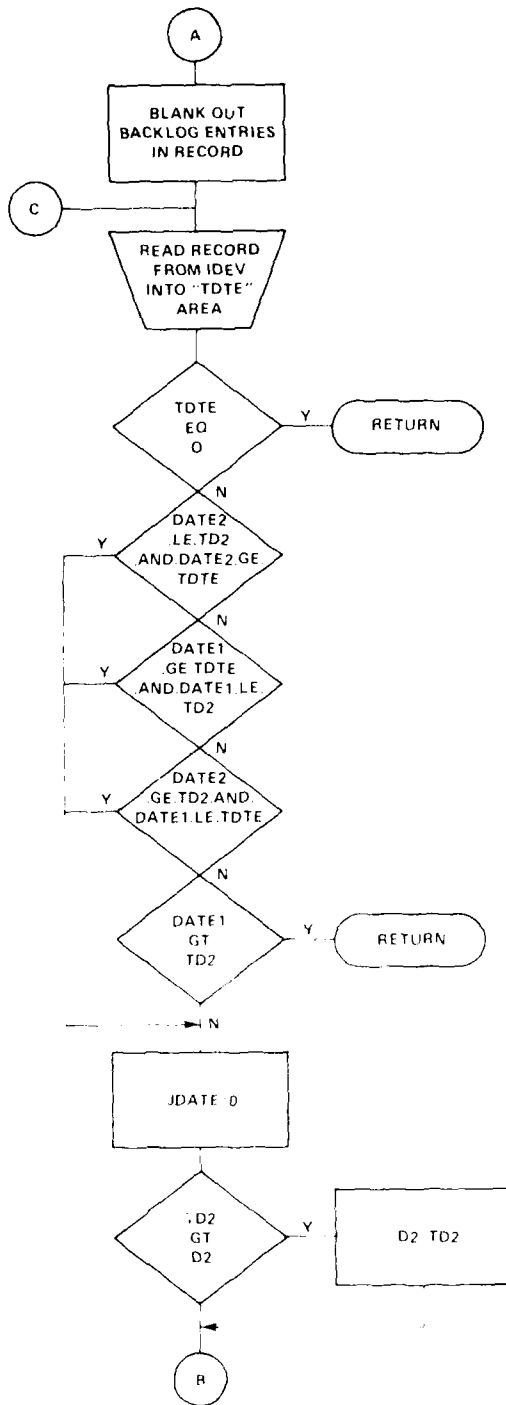
SUBROUTINE ENROR 7474 CPT=0 ROUND=0/ TRACE

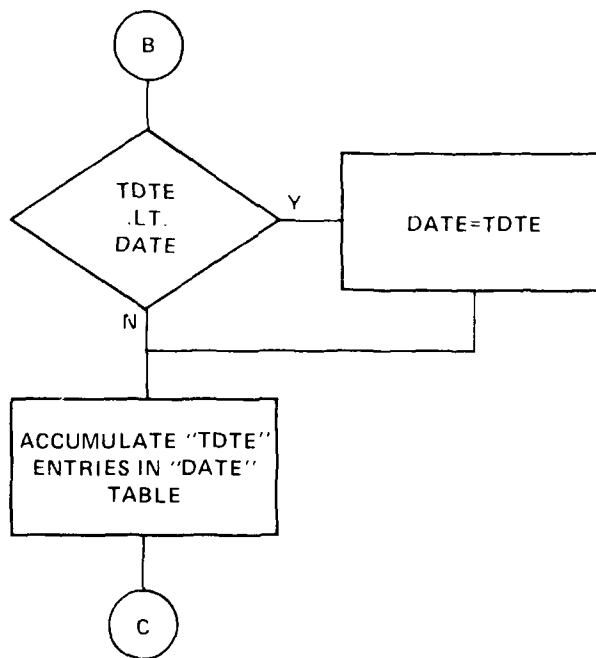
FTN 4.6*460

10/15/80 12.23.05

1	SUBROUTINE ENROR(MSGNO)	ERR	2
	INTEGER MSGNO	ERR	3
	PRINT 1000,MSGNO	ERR	4
5	1000 FORMAT(" ERROR NUMBER ",I4)	ERR	5
	STOP	ERR	6
	END	ERR	7
1	SUBROUTINE ENDF(ICEV)	ENDF	2
	INTEGER DATE,02,IPARK,REGP(50),EPAL(50)	ENDF	3
	REAL ISHFT	ENDF	4
5	ALPHA INAME(50),ORIG(50),DSTN(50)	ENDF	5
	COMMON/HISRC1/DATE,02,IPARK,REGP,EPAL	ENDF	6
	COMMON/HISRC2/ISHFT	ENDF	7
	COMMON/HISRC3/INAME,ORIG,DSTN	ENDF	8
	DATE=0	ENDF	9
10	WRITE(11,1500)DATE,02,ISHFT,IPARK,INAME,ORIG,DSTN,REGP,EPAL	ENDF	10
	1500 FORMAT(2I6,F6.1,I6,50A4,100A6,100I6)	ENDF	11
	RETURN	ENDF	12
	END	ENDF	13







1	SUBROUTINE MERGE(DATE1, DATE2, IDEV)	MRGE	2
	C DATE1-START DATE DATE2-TERMINAL DATE	MRGE	3
	C BINARY	MRGE	4
	INTEGER DATE1, DATE2, IPTR2(50), DATE, D2, IMARK, SEARCH,	MRGE	5
5	IREGP(50), EPAL(50), TOTE, T02, TMARK, TREGP(50), TEPAL(50)	MRGE	6
	LOGICAL RESET	MRGE	7
	REAL ISHFT, TSHFT, SHIFT	MRGE	8
	C ALPHA	MRGE	9
	ALPHA TMPOR, TPFUS, ADATE1, ADATE2, INAME(50), ORIG(50), OSTN(50),	MRGE	10
10	TINME(50), TOSTN(50), TORIG(50), JNAME, JORG, JCEST	MRGE	11
	COMMON/TMPRL1/TOTE, T02, TMARK, TREGP, TEPAL	MRGE	12
	COMMON/TMPRC2/TSHFT	MRGE	13
	COMMON/TMPRC3/TINME, TORIG, TOSTN	MRGE	14
	COMMON/HISRC1/DATE, D2, IMARK, REGP, EPAL	MRGE	15
15	COMMON/HISRC2/ISHFT	MRGE	16
	COMMON/HISRC3/INAME, JORG, OSTN	MRGE	17
	REWIND IDEV	MRGE	18
	READ(IDEV, 1000, END=5000) DATE, D2, ISHFT, IMARK, INAME, ORIG, OSTN, REGP,	MRGE	19
	1 EPAL	MRGE	20
20	1000 FORMAT(2I6, F0.1, I6, 5J4, 10J6, 10I6)	MRGE	21
	IF(DATE, EQ, 0) GOTO 5000	MRGE	22
	10 IF(DATE2, LE, D2, AND, DATE2, GE, DATE) GOTO 20	MRGE	23
	IF(DATE1, GE, DATE, AND, DATE1, LE, D2) GOTO 20	MRGE	24
	IF(DATE2, GE, D2, AND, DATE1, LE, DATE) GOTO 20	MRGE	25
25	IF(DATE1, GT, D2) GOTO 16	MRGE	26
	15 READ(IDEV, 1000, END=16) DATE, D2, ISHFT, IMARK, INAME, ORIG, OSTN, REGP,	MRGE	27
	1 EPAL	MRGE	28
	IF(DATE, EQ, 0) GOTO 16	MRGE	29
	GOTO 10	MRGE	30
30	16 PRINT 4000	MRGE	31
	4000 FORMAT(1H, "RECORD NOT FOUND FOR REQUESTED DATES AND/OR SHIFT")	MRGE	32
	STOP	MRGE	33
	20 DD 25 I=1.50	MRGE	34
	IF(INAME(I), NE, 4HZZZZ) GOTO 25	MRGE	35
35	ORIG(I)=10H	MRGE	36
	OSTN(I)=10H	MRGE	37
	INAME(I)=10H	MRGE	38
	REGP(I)=0	MRGE	39
	EPAL(I)=0	MRGE	40
40	25 CONTINUE	MRGE	41
	27 READ(IDEV, 1000, END=5010) TOTE, T02, TSHFT, TMARK, TINME, TORIG, TOSTN,	MRGE	42
	1 TREGP, TEPAL	MRGE	43
	IF(TOTE, EQ, 0) RETURN	MRGE	44
	IF(DATE2, LE, T02, AND, DATE2, GE, TOTE) GOTO 30	MRGE	45
45	IF(DATE1, GE, TOTE, AND, DATE1, LE, T02) GOTO 30	MRGE	46
	IF(DATE2, GE, T02, AND, DATE1, LE, TOTE) GOTO 30	MRGE	47
	IF(DATE1, GT, T02) RETURN	MRGE	48
50	30 JDATE=0	MRGE	49
	IF(T02, GT, D2) D2=T02	MRGE	50
	IF(TOTE, LT, DATE) DATE=TOTE	MRGE	51
55	DD 42 I=1.50	MRGE	52
	IF(TINME(I), EQ, 10H, .AND, TORIG(I), EQ, 10H, .AND,	MRGE	53
	1 TOSTN(I), EQ, 10H,) GOTO 42	MRGE	54
	IF(TINME(I), EQ, 4HZZZZ) GOTO 42	MRGE	55
	JNAME=TINME(I)	MRGE	56
	JORG=TORIG(I)	MRGE	57
	JCEST=TOSTN(I)	MRGE	58

SUBROUTINE MERGE

74/74

CPT=0 ROLND=97 TRACE

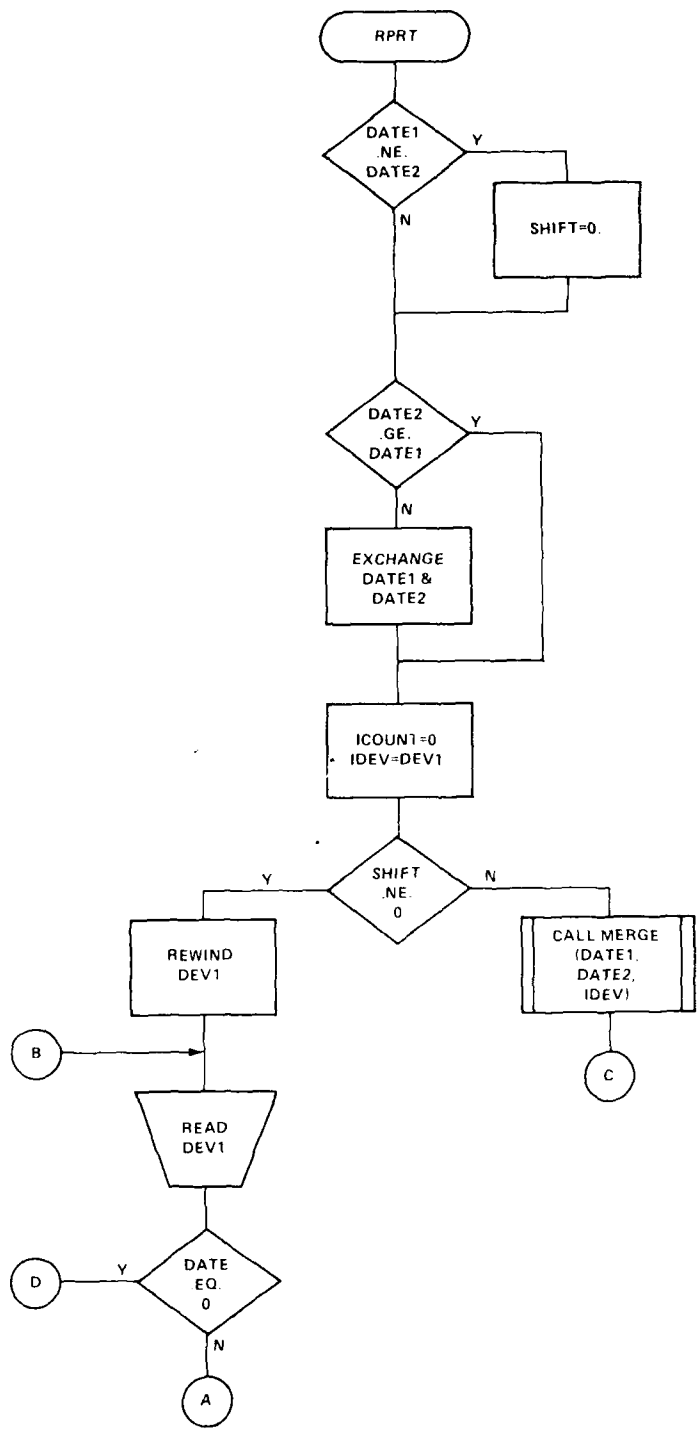
FTN 4,6+460

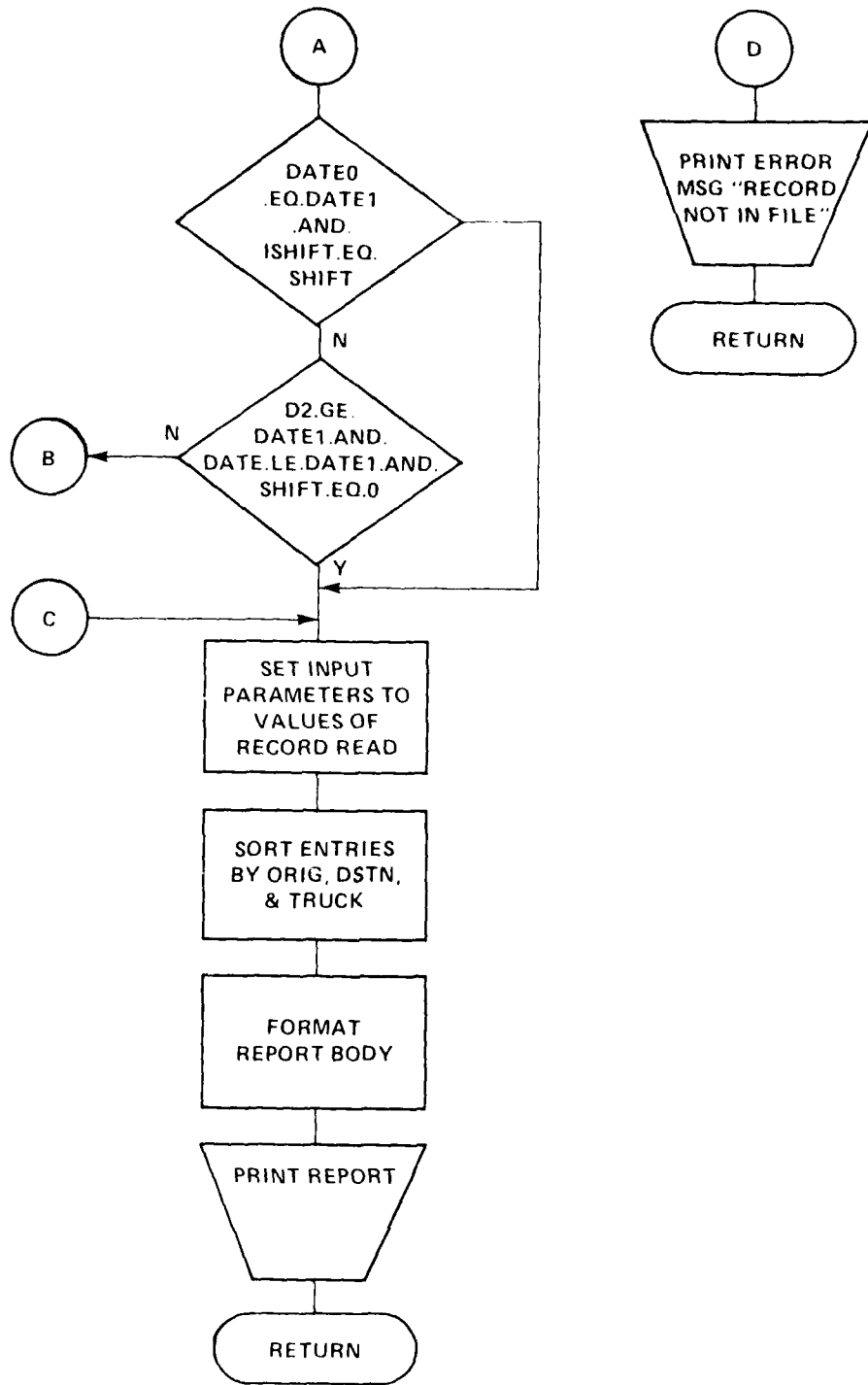
10/15/80 12.27.05

	RESET=.T.	MPGE	59
	INDEX=SEARCH(JDATE,JNAME,JORG,JOEST,RESET)	MPGF	60
60	IF(INDEX.NE.0) GOTO 35	MPGF	61
	JNAME=10H	MPGF	62
	JORG=10H	MPGF	63
	JOEST=1)H	MPGF	64
	INDEX=SEARCH(JDATE,JNAME,JORG,JOEST,RESET)	MPGF	65
65	IF(INDEX.EQ.0)GOTO 40	MPGE	66
	INAME(INDEX)=TINME(I)	MPGF	67
	ORIG(INDEX)=TORIG(I)	MPGE	68
	OSTN(INDEX)=TOSTN(I)	MPGF	69
70	35 REGP(INDEX)=RECP(INDEX)+TREGP(I)	MPGF	70
	EPAL(INDEX)=EPAL(INDEX)+TEPAL(I)	MPGF	71
	GOTO 42	MPGF	72
	40 PRINT 1500,JDATE,IC2,TORIG(I),TOSTN(I),TINME(I),TREGP(I),TEPAL(I)	MPGF	73
	1500 FORMAT(" FOLLOWING ENTRY NOT INCLUDED ",2I6,2A6,44,2I6)	MPGF	74
	42 CONTINUE	MPGF	75
75	GOTO 27	MPGF	76
	5000 CALL ERROR(300)	MPGE	77
	5010 RETURN	MPGF	78
	END	MPGE	79

1	SUBROUTINE MRGIT(DATE1, DATE2)	ENDIF	14
	ALPHA TINME(5), TORIG(50), TOSTN(50), INAME(50), ORIG(50), OSTN(50)	ENDIF	15
	INTEGER DEV1, DEV2, CARRY(10), TREGP(50), TEPAL(50), REGP(50), EPAL(50),	ENDIF	16
	1 DATE, TOTE, D2, T02, IMARK, T MARK, DATE1, DATE2	ENDIF	17
5	REAL TSHFT, ISHFT	ENDIF	18
	LOGICAL UPDT, UPOTE	ENDIF	19
	COMMON/TMPRC1/TOTE, T02, T MARK, TREGP, TEPAL	ENDIF	20
	COMMON/TMPRC2/TSHFT	ENDIF	21
	COMMON/TMPRC3/TINME, TORIG, TOSTN	ENDIF	22
10	COMMON/HISRC1/DATE, D2, IMARK, REGP, EPAL	ENDIF	23
	COMMON/HISRC2/ISHFT	ENDIF	24
	COMMON/HISRC3/INAME, ORIG, OSTN	ENDIF	25
	COMMON/CNTRL/UPDT, DARRY, DEV1, DEV2, UPOTE	ENDIF	26
	DEV1=8	ENDIF	27
15	REWIND DEV1	ENDIF	28
	IDEV=DEV1	ENDIF	29
	IF(DATE2, GE, DATE1) GOTO 10	ENDIF	30
	ITEMP=DATE1	ENDIF	31
	DATE1=DATE2	ENDIF	32
20	DATE2=ITEMP	ENDIF	33
10	CALL MERGE(DATE1, DATE2, IDEV)	ENDIF	34
	DATE2=D2	ENDIF	35
	DATE1=DATE	ENDIF	36
25	REWIND DEV1	ENDIF	37
	ISHFT=0.	ENDIF	38
	IMARK=0	ENDIF	39
	READ(DEV1, 1000, END=5000) TOTE, T02, TSHFT, T MARK, TINME, TORIG, TOSTN,	ENDIF	40
	1 TREGP, TEPAL	ENDIF	41
1300	FORMAT(2I6, F6.1, I6, 50A4, 100A6, 100I6)	ENDIF	42
30	15 IF(DATE2, LE, T02, AND, DATE2, GE, TOTE) GOTO 23	ENDIF	43
	IF(DATE1, LE, T02, AND, DATE1, GE, TOTE) GOTO 20	ENDIF	44
	IF(DATE2, GE, T02, AND, DATE1, LE, TOTE) GOTO 20	ENDIF	45
	WRITE(11, 1000) TOTE, T02, TSHFT, T MARK, TINME, TORIG, TOSTN, TREGP, TEPAL	ENDIF	46
	GOTO 30	ENDIF	47
35	20 IF(DATE, NE, 0) WRITE(11, 1000) DATE, D2, ISHFT, IMARK, INAME, ORIG, OSTN,	ENDIF	48
	1 REGP, EPAL	ENDIF	49
	DATE=0	ENDIF	50
30	READ(DEV1, 1000, END=40) TOTE, T02, TSHFT, T MARK, TINME, TORIG, TOSTN, TREGP	ENDIF	51
	1, TEPAL	ENDIF	52
40	IF(TOTE, EQ, 0) GOTO 40	ENDIF	53
	GOTO 15	ENDIF	54
40	IF(DATE, NE, 0) WRITE(11, 1000) DATE, D2, ISHFT, IMARK, INAME, ORIG, OSTN,	ENDIF	55
	1 REGP, EPAL	ENDIF	56
	CALL ENDF(11)	ENDIF	57
45	DEV1=11	ENDIF	58
	RETURN	ENDIF	59
5000	CALL ERROR(300)	ENDIF	60
	END	ENDIF	61

1	SUBROUTINE RPLCE	RPLCE	2
	C ALPHANUMERIC	RPLCE	3
	ALPHA NAME,TRK,ORG,DEST,TINME(50),TORIG(50),TOSTN(50),INAME(50),	RPLCE	4
	1 ORIG(50),DSTN(50)	RPLCE	5
5	C BINARY	RPLCE	6
	INTEGER YRMODE,PAL,EMERG,TOTE,TREGP(50),TEPAL(50),DATE,CARRY(10),	RPLCE	7
	REGP(50),EPAL(50)	RPLCE	8
	INTEGER IMARK,TMARK,MARK,C2,TC2,DEV1,DEV2	RPLCE	9
	REAL SHFT,ISHFT,TSHFT	RPLCE	10
10	LOGICAL UPDTE	RPLCE	11
	LOGICAL UPDT	RPLCE	12
	COMMON/AVSRC1/YRMODE,PAL,EMERG,IFTN	RPLCE	13
	COMMON/AVSRC2/SHFT	RPLCE	14
	COMMON/AVSRC3/NAME,TRK,ORG,DEST	RPLCE	15
15	COMMON/TMPC1/TOTE,TD2,TMARK,TREGP,TEPAL	RPLCE	16
	COMMON/TMPC2/TSHFT	RPLCE	17
	COMMON/TMPC3/TINME,TORIG,TOSTN	RPLCE	18
	COMMON/HISRC1/DATE,D2,IMARK,REGP,EPAL	RPLCE	19
	COMMON/HISRC2/ISHFT	RPLCE	20
20	COMMON/HISRC3/INAME,ORIG,DSTN	RPLCE	21
	COMMON/CNTRL/UPDT,CARRY,DEV1,DEV2,UPDTE	RPLCE	22
	REWIND DEV1	RPLCE	23
	REWIND DEV2	RPLCE	24
5	READ(DEV1,1500,END=50)TOTE,TD2,TSHFT,TMARK,TINME,TORIG,TOSTN,TREGP	RPLCE	25
	1,TEPAL	RPLCE	26
25	1,TEPAL	RPLCE	27
1500	FORMAT(2I6,F6.1,I6,50A4,100A6,100I6)	RPLCE	28
	IF(TOTE.EQ.0)GOTO 50	RPLCE	29
	IF(TOTE.GT.DATE) GOTO 40	RPLCE	30
	IF(TOTE.EQ.DATE) GOTO 20	RPLCE	31
30	10 WRITE(DEV2,1500)DATE,D2,ISHFT,IMARK,INAME,ORIG,DSTN,REGP,EPAL	RPLCE	32
	WRITE(DEV2,1500)TOTE,TD2,TSHFT,TMARK,TINME,TORIG,TOSTN,TREGP,	RPLCE	33
	1,TEPAL	RPLCE	34
	DATE=0	RPLCE	35
	GOTO 5	RPLCE	36
35	20 IF(TOTE.EQ.TD2) GOTO 25	RPLCE	37
	WRITE(DEV2,1500)DATE,D2,ISHFT,IMARK,INAME,ORIG,DSTN,REGP,	RPLCE	38
	1,EPAL	RPLCE	39
	DATE=0	RPLCE	40
	GOTO 5	RPLCE	41
40	25 IF(TSHFT.GT.ISHFT)GOTO 10	RPLCE	42
	IF(TSHFT.EQ.ISHFT)GOTO 30	RPLCE	43
	GOTO 40	RPLCE	44
30	30 WRITE(DEV2,1500)DATE,D2,ISHFT,IMARK,INAME,ORIG,DSTN,REGP,	RPLCE	45
	1,EPAL	RPLCE	46
	DATE=0	RPLCE	47
45	GOTO 5	RPLCE	48
40	40 WRITE(DEV2,1500)TOTE,TD2,TSHFT,TMARK,TINME,TORIG,TOSTN,TREGP,	RPLCE	49
	1,TEPAL	RPLCE	50
	GOTO 5	RPLCE	51
50	50 IF(DATE.NE.0)	RPLCE	52
	WRITE(DEV2,1500)DATE,D2,ISHFT,IMARK,INAME,ORIG,DSTN,REGP,EPAL	RPLCE	53
	IDEV = DEV2	RPLCE	54
	CALL ENJF(IDEV)	RPLCE	55
	IF(.NOT.UPDTE)CALL CHANGE(8,5,TEMP2)	RPLCE	56
55	UPDTE=.T.	RPLCE	57
	TEMP=DEV1	RPLCE	58
	DEV1=DEV2	RPLCE	58





1	SUBROUTINE RPRT(0,DATE1,DATE2,SHIFT)	RPRT	2
	C DATE1=START DATE DATE2=TERMINAL DATE SHIFT=VALID WHEN DATE1=DATE2	RPRT	3
	C BINARY	RPRT	4
	INTEGER DEV2	RPRT	5
5	INTEGER DATE1,DATE2,IPTR2(50),DATE,02,IMARK,	RPRT	6
	IREGP(50),EPAL(50),IDTE,TD2,TMARK,TREGP(50),TEPAL(50),DEV1,	RPRT	7
	1 DARRY(10)	RPRT	8
	LOGICAL UPDT,UPDTE	RPRT	9
	LOGICAL RESCT	RPRT	10
10	REAL ISHFT,TSHFT,SHIFT	RPRT	11
	C ALP4A	RPRT	12
	ALPHA TMPOR,TFPDS,ADATE1,ADATE2,INAME(50),ORIG(50),OSTN(50),	RPRT	13
	1 TINME(50),TOSTN(50),TORIG(50),MNTM1,MNTM2	RPRT	14
	COMMON/TMPRC1/IDTE,TD2,TMARK,TREGP,TEPAL	RPRT	15
15	COMMON/TMPRC2/ TSHFT	RPRT	16
	COMMON/TMPRC3/TINME,TORIG,TOSTN	RPRT	17
	COMMON/HISRC1/DATE,02,IMARK,REGP,EPAL	RPRT	18
	COMMON/HISRC2/ISHFT	RPRT	19
	COMMON/HISRC3/INAME,ORIG,OSTN	RPRT	20
20	COMMON/CNTRL/UPDT,DARRY,DEV1,DEV2,UPDTE	RPRT	21
	IF(0=TE1.NE.DATE2) SHIFT=0.	RPRT	22
	IF(0=TE2.GE.DATE1) GOTO 5	RPRT	23
	ITEMP=DATE2	RPRT	24
	DATE2=DATE1	RPRT	25
25	DATE1=ITEMP	RPRT	25
	ICOUNT=J	RPRT	27
	IUEV=DEV1	RPRT	28
	IF(ISHFT.NE.0.) GOTO 10	RPRT	29
	CALL MERGE(0,DATE1,DATE2,IUEV)	RPRT	30
30	GOTO 50	RPRT	31
	10 REMIND DEV1	RPRT	32
	READ(DEV1,1000,END=5000)DATE,02,ISHFT,IMARK,INAME,ORIG,OSTN,REGP,	RPRT	33
	1 EPAL	RPRT	34
	1000 FORMAT(2I6,F6.1,I6,50A4,100A6,100I6)	RPRT	35
35	IF(0=DATE.EQ.0) GOTO 5000	RPRT	35
	15 IF(0=DATE.EQ.0) GOTO 50	RPRT	37
	IF(02.GE.DATE1.AND.DATE.LE.0) GOTO 50	RPRT	38
	READ(DEV1,1000,END=20)DATE,02,ISHFT,IMARK,INAME,ORIG,OSTN,REGP,	RPRT	39
	1 EPAL	RPRT	40
40	IF(0=DATE.EQ.0) GOTO 20	RPRT	41
	GOTO 15	RPRT	42
	20 PRINT 4000	RPRT	43
	4000 FORMAT(1H,"RECORD NOT FOUND FOR REQUESTED DATE AND SHIFT")	RPRT	44
	RETJFN	RPRT	45
45	50 DATE2=02	RPRT	46
	DATE1=DATE	RPRT	47
	SHIFT=ISHFT	RPRT	48
	DO 52 I=1,50	RPRT	49
	IF(0=IG(I).EQ.10H .AND.OSTN(I).EQ.10H .AND.	RPRT	50
50	1 INAME(I).EQ.10H) GOTO 52	RPRT	51
	ICOUNT=ICOUNT+1	RPRT	52
	IPTR2(ICOUNT)=I	RPRT	53
	52 CONTINUE	RPRT	54
	53 DO 70 I1=1,ICOUNT	RPRT	55
55	I=ICOUNT+1-I1	RPRT	56
	I2=I-1	RPRT	57
	DO 70 J=1,I2	RPRT	58

	INDEXA=IPTR2(J)	RPT	59
	INDEXB=IPTR2(J+1)	RPT	60
60	IF (OR IG(INDEXA).LT.ORIG(INDEXB))GOTO 70	RPT	61
	IF (OR IG(INDEXA).NE.ORI((INDEXB))GOTO 55	RPT	62
	IF (OSTN(INDEXA).L1.OSTN(INDEXB))GOTO 70	RPT	63
	IF (OSTN(INDEXA).NE.OSTN(INDEXB))GOTO 55	RPT	64
	IF (INAME(INDEXA).LE.INAME(INDEXB))GOTO 70	RPT	65
65	55 IPTR2(J)=INDEXB	RPT	66
	IPTR2(J+1)=INDEXA	RPT	67
	70 CONTINUE	RPT	68
	JJ=JJ+1	RPT	69
	INDEXA=IPTR2(J-1)	RPT	70
70	INDEXB=IPTR2(J)	RPT	71
	IF (OR IG(INDEXA).NE.10H)TMPOR=ORIG(INDEXA)	RPT	72
	IF (OSTN(INDEXA).NE.10H)TMPOS=OSTN(INDEXA)	RPT	73
	IF (OR IG(INDEXA).EQ.TMPOR)ORIG(INDEXB)=10H	RPT	74
	IF (OSTN(INDEXA).EQ.TMPOS.AND.ORIG(INDEXB).EQ.10H)	RPT	75
75	1 OSTN(INDEXA)=10H	RPT	76
	80 CONTINUE	RPT	77
	ILINE=10	RPT	78
	PRINT 2000	RPT	79
	2000 FORMAT(1H,25X,"AVS HISTORY FILE REPORT")	RPT	80
80	CALL DATRAN(DATE1,IDA1,IYR1,MNTH1)	RPT	81
	IF (DATE1.EQ.DATE2) GOTO 100	RPT	82
	CALL DATRAN(DATE2,IDA2,IYR2,MNTH2)	RPT	83
	PRINT 2050,IDA1,MNTH1,IYR1,IDA2,MNTH2,IYR2	RPT	84
2050	FORMAT(1H,20X,I2,A3,I2," - ",I2,A3,I2)	RPT	85
85	GOTO 110	RPT	86
	100 PRINT 2100,IDA1,MNTH1,IYR1	RPT	87
	2100 FORMAT(1H,34X,I2,A3,I2)	RPT	88
	IF (SHIFT.EQ.0.) GOTO 110	RPT	89
	PRINT 2110,SHIFT	RPT	90
90	2110 FORMAT(1H,32X,"SHIFT=",F6.1)	RPT	91
	PRINT 2125	RPT	92
	PRINT 2150	RPT	93
	PRINT 2275	RPT	94
95	2275 FORMAT(1H,116,"FROM",T27,"TO",T35,"NAME",T41,"REGULAR",T51,	RPT	95
	1"EMERGENCY",T63,"BACMLCS")	RPT	96
	PRINT 2295	RPT	97
	2295 FORMAT(1H,116,55(1H-))	RPT	98
	ILINE=11	RPT	99
	GOTO 130	RPT	100
100	110 PRINT 2125	RPT	101
	2125 FORMAT(1H,116,"DELIVERIES")	RPT	102
	PRINT 2150	RPT	103
	2150 FORMAT(1H,117,"WAREHOUSE NAME",T34,"TRUCK",T47,"(PALLET(S))")	RPT	104
	PRINT 2175	RPT	105
105	2175 FORMAT(1H,116,"FROM",T27,"TO",T35,"NAME",T41,"REGULAR",T51,	RPT	106
	1"EMERGENCY")	RPT	107
	PRINT 2285	RPT	108
	2285 FORMAT(1H,116,45(1H-))	RPT	109
	130 JJ=JJ+1	RPT	110
110	ILINE=ILINE+1	RPT	111
	IF (ILINE.LE.40)GOTO 105	RPT	112
	ILINE=9	RPT	113
	PRINT 3000	RPT	114
	3000 FORMAT(1H1)	RPT	115

SUBROUTINE RPRT

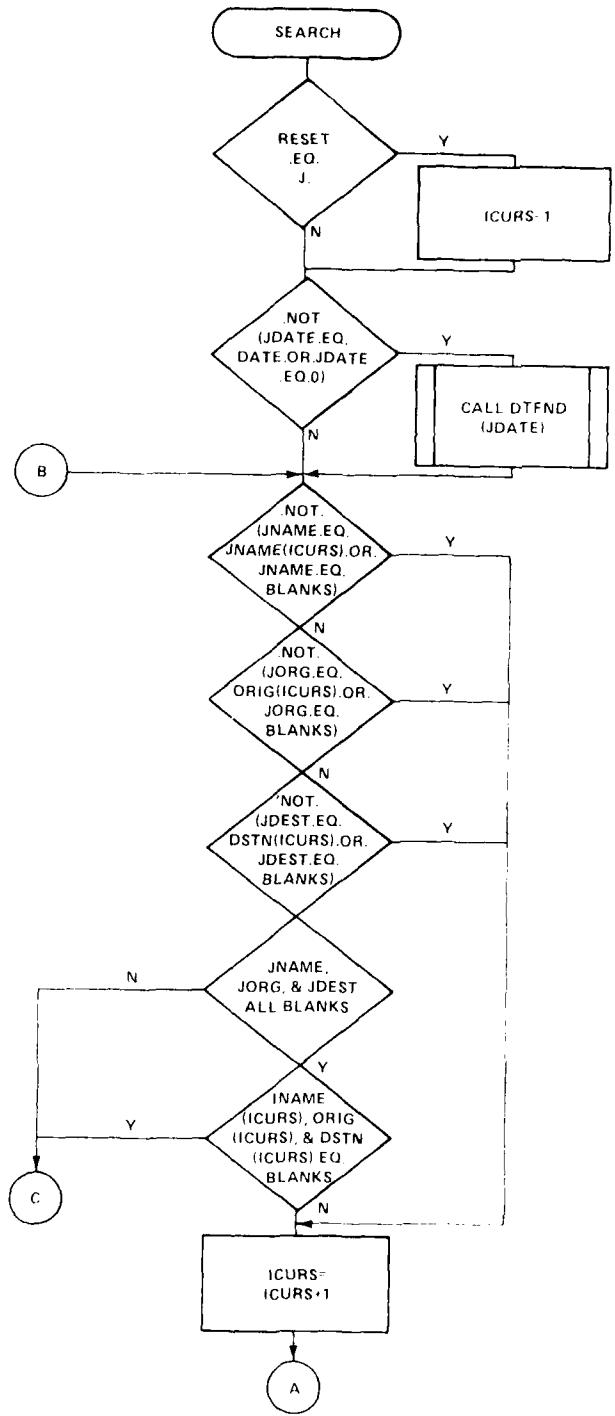
7+774

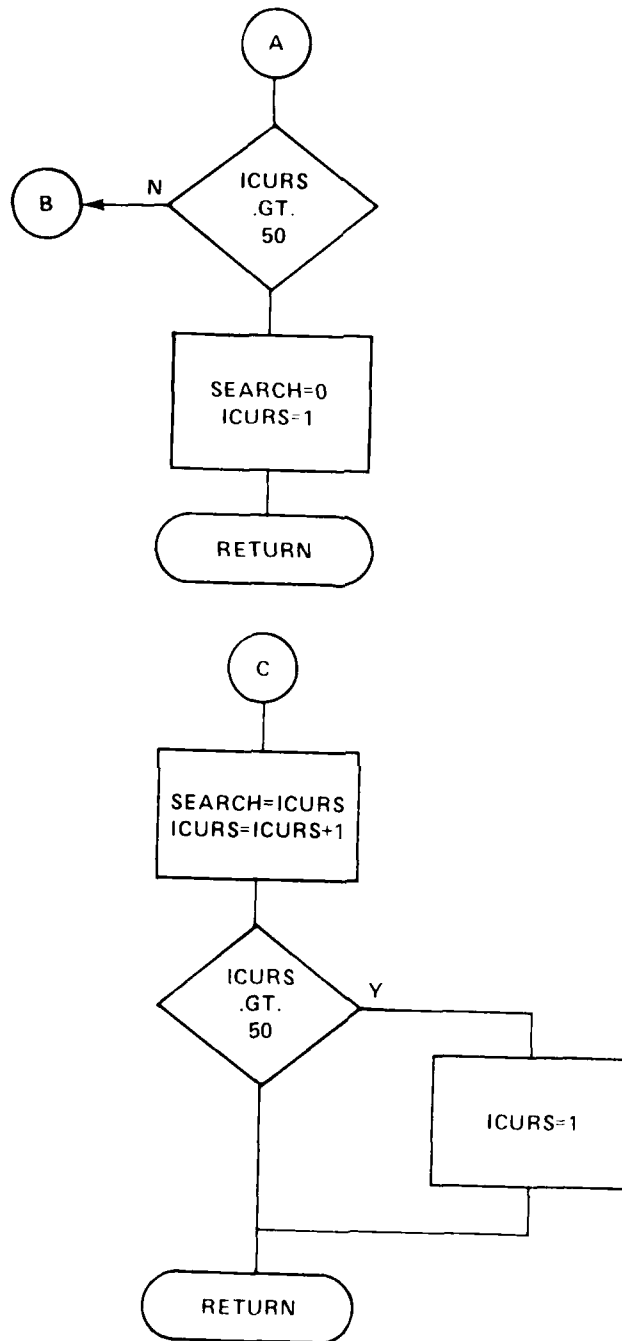
LPT=0 ROUND=* / TRACE

FTN 4,6+460

10/15/80 12.23.05

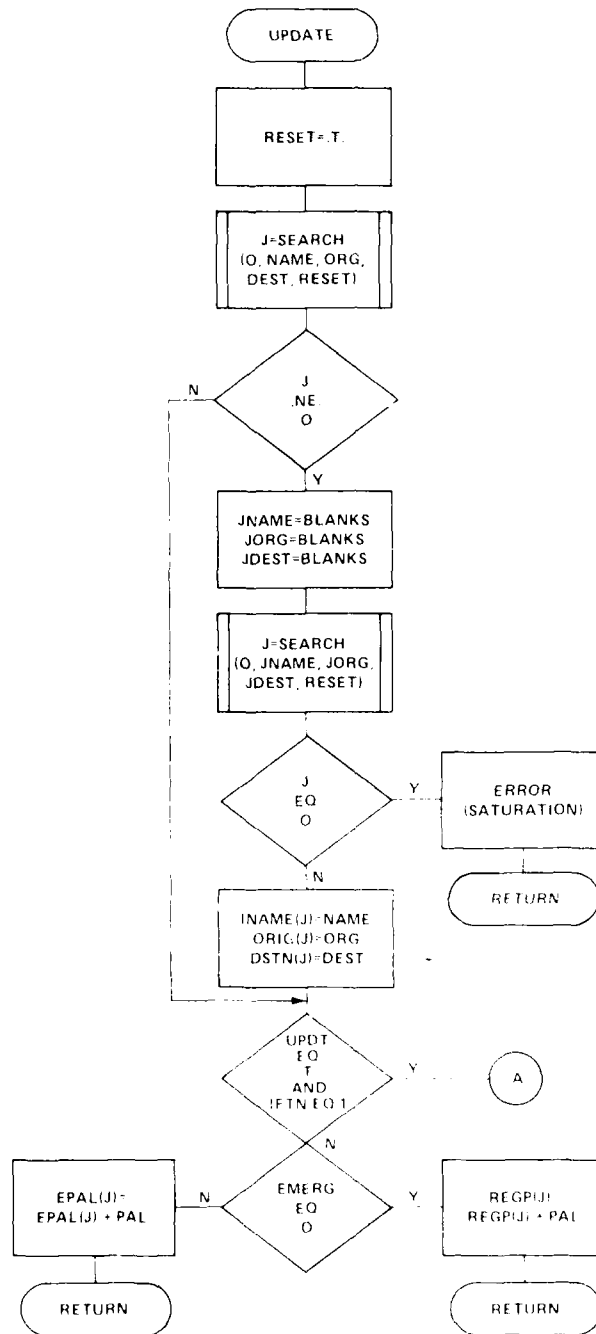
115	IF (SHIFT.EQ.0.) GOTO 140	RPRT	116
	PRINT 2125	RPRT	117
	PRINT 2150	RPRT	118
	PRINT 2175	RPRT	119
	PRINT 2205	RPRT	120
120	GOTO 125	RPRT	121
	140 PRINT 2125	RPRT	122
	PRINT 2150	RPRT	123
	PRINT 2275	RPRT	124
	PRINT 2295	RPRT	125
125	105 I=IPTR2(I1)	RPRT	126
	IF (ORIG(I).EQ.10H) GOTO 115	RPRT	127
	PRINT 2225	RPRT	128
	2225 FJRMAT (1M)	RPRT	129
	ILINE=ILINE+1	RPRT	130
130	115 IF (INAME (I).EQ.4MZZZ) GOTO 125	RPRT	131
	PRINT 2200,ORIG(I),OSTN(I),INAME(I),REGP(I),EPAL(I)	RPRT	132
	2200 FORMAT (1M ,T17,A6,T25,A6,T35,A4,T43,I6,T53,I6)	RPRT	133
	GOTO 120	RPRT	134
	125 PRINT 2500,ORIG(I),OSTN(I),REGP(I)	RPRT	135
135	2500 FJRMAT (1M ,T17,A6,T25,A6,T65,I6)	RPRT	136
	120 CONTINUE	RPRT	137
	RETURN	RPRT	138
	5000 CALL ERROR (JQC)	RPRT	139
	END	RPRT	140

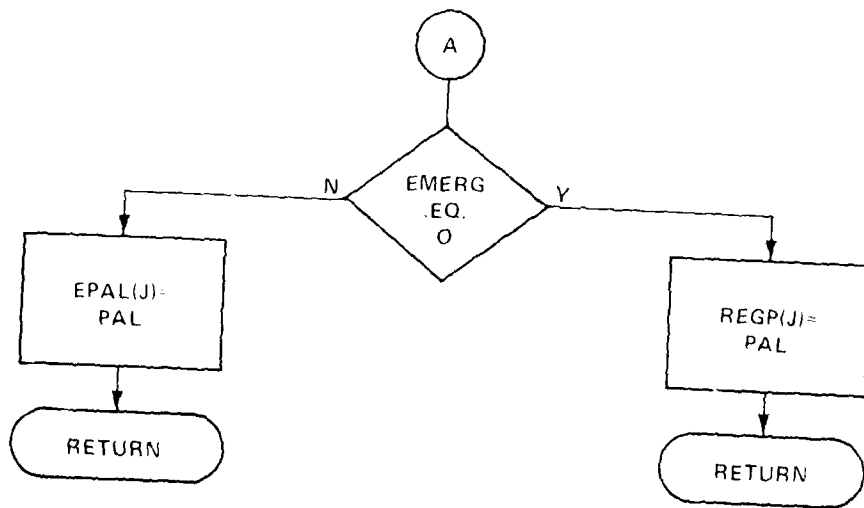




```

1      INTEGER FUNCTION SEARCH(JDATE,JNAME,JORG,JDEST,RFSET)
C ALPHANUM RIDS
      ALPHA JNAME,JORG,JDEST,NAME,ORG,DEST,INAME(50),ORIG(50),TRK,
      1)STN(50)
5      C BINARY
      INTEGER D2
      INTEGER DATE,YRMOCA,JDATE,TOTE,PAL,EMERG,REGP(50),
      1)PAL(50),ICURS
      REAL ISHFT
10     LOGICAL RESET
      COMMON/AVSRC1/YRMOCA,PAL,EMERG,IFTN
      COMMON/AVSRC2/ISHFT
      COMMON/AVSRC3/NAME,TRK,ORG,DEST
      COMMON/HISRC1/DATE,D2,IMARK,REGP,EPAL
15     COMMON/HISRC2/ISHFT
      COMMON/HISRC3/INAME,ORIG,OSTN
      IF(.NOT.(JDATE.EQ.DATE.OR.JDATE.EQ.0)) CALL DTEND(JDATE)
5      IF(.NOT.(JNAME.EQ.INAME(ICURS).OR.JNAME.EQ.10H
20     IF(.NOT.(JORG.EQ.ORIG(ICURS).OR.JORG.EQ.10H
      IF(.NOT.(JDEST.EQ,OSTN(ICURS).OR,JDEST.EQ.10H
C THIS IS EITHER A HIT OR ALL BLANKS -CHECK FOR LATTER
      IF(JNAME.EQ.10H .AND.JORG.EQ.10H .AND.
1 JDEST.EQ.10H 1)GOTO 7
      GOTO 23
25     C IF INPUT IS NOT BLANKS, THIS IS A HIT-X-FER TO 20
7      IF(INAME(ICURS).EQ.10H .AND.ORIG(ICURS).EQ.10H
      1) .AND,OSTN(ICURS).EQ.10H 1)GOTO 20
      GOTO 10
30     C IF THE TABLE LINE IS ALL BLANKS YOO, THIS IS A HIT- X-FER TO 20
C OTHERWISE READ THE NEXT LINE IN THE TABLE
10     ICURS =ICURS+1
      IF(ICURS.GT.50) GOTO 15
      GOTO 5
35     15 SEARCH=J
      ICURS=1
      R=J-N
40     20 SEARCH=ICURS
      ICURS=ICURS+1
      IF(ICURS.GT.50) ICURS=1
      R=J-N
      END
      SRCH 2
      SRCH 3
      SRCH 4
      SRCH 5
      SRCH 5
      SRCH 7
      SRCH 8
      SRCH 9
      SRCH 10
      SRCH 11
      SRCH 12
      SRCH 13
      SRCH 14
      SRCH 15
      SRCH 16
      SRCH 17
      SRCH 18
      SRCH 19
      SRCH 20
      SRCH 21
      SRCH 22
      SRCH 23
      SRCH 24
      SRCH 25
      SRCH 26
      SRCH 27
      SRCH 28
      SRCH 29
      SRCH 30
      SRCH 31
      SRCH 32
      SRCH 33
      SRCH 34
      SRCH 35
      SRCH 36
      SRCH 37
      SRCH 38
      SRCH 39
      SRCH 40
      SRCH 41
      SRCH 42
      SRCH 43
  
```





SUBROUTINE UPDTE

74/74

CPT=0 ROUND=97 TRACE

FTN 4.6+460

10/15/80 12.23.45

		UPDTE	2
1	SUBROUTINE UPDTE	UPDTE	3
	C ALPHANUMERIC	UPDTE	4
	ALPHA JORG,JDEST,JNAME,NAME,ORG,DEST,INAME(50),ORIG(50),DSTN(50),	UPDTE	5
	1 TRK	UPDTE	6
5	C BINARY	UPDTE	7
	INTEGER DATE,C2	UPDTE	8
	INTEGER JDATE,EMERG,REGP(50),PAL,YRMOCA,EPAL(50),CARRY(10),	UPDTE	9
	IDEV1,DEV2,SEARCH	UPDTE	10
	REAL ISHFT	UPDTE	11
10	LOGICAL RESET,UPDTE,UPDT	UPDTE	12
	COMMON/VSRC1/YRMOCA,PAL,EMERG,IFTN	UPDTE	13
	COMMON/VSRC2/ISHFT	UPDTE	14
	COMMON/VSRC3/NAME,TRK,ORG,DEST	UPDTE	15
15	COMMON/HISRC1/DATE,C2,INMARK,REGP,EPAL	UPDTE	16
	COMMON/HISRC2/ISHFT	UPDTE	17
	COMMON/HISRC3/INAME,ORIG,DSTN	UPDTE	18
	COMMON/CTRL/UPDT,CARRY,DEV1,DEV2,UPDTE	UPDTE	19
	RESET=.T.	UPDTE	20
	JDATE=0	UPDTE	21
20	JNAME=NAME	UPDTE	22
	JORG=ORG	UPDTE	23
	JDEST=DEST	UPDTE	24
	J=SEARCH(JDATE,JNAME,JORG,JDEST,RESET)	UPDTE	25
	IF(J.NE.0) GOTO 10	UPDTE	26
25	C FIN) NEXT BLANK AREA IN TABLE	UPDTE	27
	JNAME=10H	UPDTE	28
	JORG=10H	UPDTE	29
	JDEST=10H	UPDTE	30
	J=SEARCH(JDATE,JNAME,JORG,JDEST,RESET)	UPDTE	31
30	IF(J.NE.0) GOTO 5	UPDTE	32
	PRINT 1000,PAL,JORG,JDEST,JNAME	UPDTE	33
	1000 FORMAT(1H,"TABLE SATURATION-FOLLOWING NOT INCLUDED",/	UPDTE	34
	1 1F.16," PALLETS FROM ",A6," TO ",A6," VIA ",A4)	UPDTE	35
	RETURN	UPDTE	36
35	5 INAME(J)=NAME	UPDTE	37
	ORIG(J)=ORG	UPDTE	38
	DSTN(J)=DEST	UPDTE	39
40	IF(UPDT.AND.(IFIN.EQ.1)) GOTO 40	UPDTE	40
	IF(EMERG.EQ.0) GOTO 20	UPDTE	41
40	EPAL(J)=EPAL(J)+PAL	UPDTE	42
	UPDTE=.T.	UPDTE	43
	RETURN	UPDTE	44
20	REGP(J)=REGP(J)+PAL	UPDTE	45
	UPDTE=.T.	UPDTE	46
	RETURN	UPDTE	47
40	IF(EMERG.EQ.0) GOTO 50	UPDTE	48
	EPAL(J)=PAL	UPDTE	49
	UPDTE=.T.	UPDTE	50
	RETURN	UPDTE	51
50	REGP(J)=PAL	UPDTE	52
	UPDTE=.T.	UPDTE	53
	RETURN	UPDTE	54
	END		

APPENDIX E - SAMPLE RUNS

SAMPLE - AVS1 RUN, REGULAR ORDERS

 AVS REGULAR ORDER PROGRAM

10/20/80

800.0

(OPT=0)

 ORDERS

1	10	PALLETS FROM SM	TU NWS
2	0	PALLETS FROM 1605	TO NWS
3	5	PALLETS FROM 23	TO NWS
4	8	PALLETS FROM 1	TO NWS
5	12	PALLETS FROM 1	TO NWS
6	6	PALLETS FROM 1172	TO 198
7	0	PALLETS FROM 198	TO 1172
8	12	PALLETS FROM 6-W	TO 198
9	10	PALLETS FROM 6-W	TO 224
10	11	PALLETS FROM 6-W	TO X10
11	12	PALLETS FROM 6-W	TO 1606
12	3	PALLETS FROM 67E	TO 224
13	11	PALLETS FROM 67E	TO SM
14	10	PALLETS FROM 67E	TO X10
15	1	PALLETS FROM 67E	TO 16
16	1	PALLETS FROM 67E	TO 23
17	1	PALLETS FROM 67E	TO 61
18	1	PALLETS FROM 67E	TO 647
19	1	PALLETS FROM 67E	TO 1
20	8	PALLETS FROM 67E	TO 1621
21	1	PALLETS FROM 67E	TO 49
22	1	PALLETS FROM 67E	TO 64
23	11	PALLETS FROM 1604	TO 224
24	11	PALLETS FROM 1604	TO 647
25	11	PALLETS FROM 1604	TO 23
26	1	PALLETS FROM 1604	TO 1
27	11	PALLETS FROM 1603	TO 1172
28	11	PALLETS FROM 1603	TO 1605
29	11	PALLETS FROM 1603	TO 1503
30	11	PALLETS FROM 1603	TO 1138
31	5	PALLETS FROM 1603	TO 1606
32	1	PALLETS FROM 191	TO 224
33	1	PALLETS FROM 191	TO X10
34	2	PALLETS FROM 191	TO 23
35	2	PALLETS FROM 1502	TO 1621
36	7	PALLETS FROM 1502	TO 224
37	3	PALLETS FROM 1605	TO 224
38	11	PALLETS FROM 1605	TO SM
39	3	PALLETS FROM 1605	TO 1606
40	11	PALLETS FROM 1605	TO X10
41	1	PALLETS FROM 1605	TO 23
42	1	PALLETS FROM 1605	TO 647
43	1	PALLETS FROM 1605	TO 1621

43	1	PALLETS FROM 1002	TO 1021
44	2	PALLETS FROM 1002	TO 224
45	7	PALLETS FROM 1002	TO SM
46	1	PALLETS FROM 1002	TO 23
47	1	PALLETS FROM 1002	TO 64b
48	2	PALLETS FROM 1002	TO 647
49	30	PALLETS FROM 1002	TO 1507
50	1	PALLETS FROM 60E	TO 1503
51	10	PALLETS FROM 60E	TO 191
52	1	PALLETS FROM 60E	TO 224
53	1	PALLETS FROM 00E	TO SM
54	1	PALLETS FROM 60E	TO X10
55	10	PALLETS FROM 1004	TO 647
56	2	PALLETS FROM 1004	TO 224
57	1	PALLETS FROM SM	TO 1172
58	1	PALLETS FROM 67H	TO SM

VEHICLES SELECTED

1	VEHICLE ST 1	CAPACITY =	7	PALLETS,	ROUTE DURATION =	240.0	MINS.
2	VEHICLE ST 2	CAPACITY =	7	PALLETS,	ROUTE DURATION =	240.0	MINS.
3	VEHICLE ST 3	CAPACITY =	7	PALLETS,	ROUTE DURATION =	240.0	MINS.
4	VEHICLE TR 1	CAPACITY =	12	PALLETS,	ROUTE DURATION =	240.0	MINS.
5	VEHICLE TT 1	CAPACITY =	14	PALLETS,	ROUTE DURATION =	240.0	MINS.
6	VEHICLE IT 1	CAPACITY =	10	PALLETS,	ROUTE DURATION =	240.0	MINS.
7	VEHICLE IT 2	CAPACITY =	10	PALLETS,	ROUTE DURATION =	240.0	MINS.

VEHICLE - ST 1
 START TIME - 800.
 DATE 102080

STOP	SITE	TIME	DELIVER	PICK UP	ORDER	STAY TIME
1	1602	809		7 PALLETS	45	3
2	SM	814	7 PALLETS		45	3
3	1502	823		7 PALLETS	36	3
4	224	838	7 PALLETS		36	3
5	00E	853		1 PALLETS	54	2
				1 PALLETS	52	1
6	07c	057		3 PALLETS	12	3
7	X10	911	1 PALLETS		54	2
8	224	915	1 PALLETS		52	2
			3 PALLETS		12	1
9	1003	941		5 PALLETS	31	3
10	1006	1006	5 PALLETS		31	3
11	1004	1011		2 PALLETS	56	2
12	1002	1015		2 PALLETS	44	2
13	191	1019		1 PALLETS	32	2
14	224	1044	2 PALLETS		56	2
			2 PALLETS		44	1
			1 PALLETS		32	1
15	191	1110		2 PALLETS	34	2
16	1602	1114		1 PALLETS	46	2
17	23	1143	2 PALLETS		34	2
			1 PALLETS		40	1

ROUTE ENDED
 LOCATION =1073
 TIME = 1211
 NO OF PALLETS MOVED = 32

VEHICLE - ST 2
 START TIME - 800.
 DATE 102080

STOP	SITE	TIME	DELIVER	PICK UP	ORDER	STAY TIME
1	1602	809		7 PALLETS	49	3 *SPLIT
2	1607	818	7 PALLETS		49	3 *SPLIT
3	1602	834		7 PALLETS	49	3 *SPLIT
4	1607	839	7 PALLETS		49	3 *SPLIT
5	1602	855		7 PALLETS	49	3 *SPLIT
6	1607	900	7 PALLETS		49	3 *SPLIT
7	1602	916		7 PALLETS	49	3 *SPLIT
8	1607	921	7 PALLETS		49	3 *SPLIT
9	1604	937		7 PALLETS	23	3 *SPLIT
10	224	1000	7 PALLETS		23	3 *SPLIT
11	1604	1026		4 PALLETS	23	3 *SPLIT
12	224	1031	4 PALLETS		23	3 *SPLIT

ROUTE ENDED
 LOCATION = 1078
 TIME = 1042
 NO OF PALLETS MOVED = 39

VEHICLE - ST 3
START TIME - 800.
DATE 102080

```
*****  
STOP SITE TIME DELIVER PICK UP ORDER STAY TIME  
*****  
1 1004 809 7 PALLETS 25 3 *SPLIT  
2 23 830 7 PALLETS 25 3 *SPLIT  
3 1004 908 4 PALLETS 25 3 *SPLIT  
4 23 912 4 PALLETS 25 3 *SPLIT  
5 1004 942 7 PALLETS 24 3 *SPLIT  
6 647 953 7 PALLETS 24 3 *SPLIT  
7 1004 1026 4 PALLETS 24 3 *SPLIT  
8 647 1030 4 PALLETS 24 3 *SPLIT
```

ROUTE ENDED
LOCATION =1078
TIME = 1053
NO OF PALLETS MOVED = 22

VEHICLE - TR 1
 START TIME - 800.
 DATE 102080

STOP	SITE	TIME	DELIVER	PICK UP	ORDER	STAY TIME
1	04W	802		12 PALLETS	8	12
2	19A	815	12 PALLETS		8	12
3	04W	828		12 PALLETS	11	12
4	1006	846	12 PALLETS		11	12
5	1003	900		11 PALLETS	28	11
6	1005	912	11 PALLETS		28	11
				11 PALLETS	30	9
7	SM	930	11 PALLETS		38	11
8	07E	951		11 PALLETS	13	11
9	00E	1003		1 PALLETS	53	3
10	SM	1008	1 PALLETS		53	3
			11 PALLETS		13	9
11	04W	1021		11 PALLETS	10	11
12	X10	1042	11 PALLETS		10	11
13	1003	1113		11 PALLETS	30	11
14	1138	1134	11 PALLETS		30	11
15	1003	1151		11 PALLETS	29	11
16	1003	1213	11 PALLETS		29	11

ROUTE ENDED
 LOCATION = 1074
 TIME = 1227
 NO OF PALLETS MOVED = 91

VEHICLE - TT 1
START TIME - 800.
DATE 102080

STOP	SITE	TIME	DELIVER	PICK UP	ORDER	STAY TIME
1	SM	802		10 PALLETS	1	22
2	NWS	909	10 PALLETS		1	22
3	1005	1016		6 PALLETS	2	16
4	1	1053		8 PALLETS	4	19
5	NWS	1156	8 PALLETS 6 PALLETS		4 2	19 11

ROUTE ENDED
LOCATION = 1078
TIME = 1310
NO OF PALLETS MOVED = 24

VEHICLE - IT 1
START TIME - 800.
DATE 102080

```
*****  
STOP SITE TIME DELIVER PICK UP ORDER STAY TIME  
*****  
1 66E 802 10 PALLETS 10 PALLETS 51 22  
2 191 833 10 PALLETS 10 PALLETS 51 22  
3 1604 857 10 PALLETS 10 PALLETS 55 22  
4 647 949 10 PALLETS 10 PALLETS 55 22  
5 644 1031 10 PALLETS 10 PALLETS 9 22  
6 224 1102 10 PALLETS 10 PALLETS 9 22  
7 67E 1136 10 PALLETS 10 PALLETS 14 22  
8 X10 1210 10 PALLETS 10 PALLETS 14 22
```

ROUTE ENDED
LOCATION = 1074
TIME = 1244
NO OF PALLETS MOVED = 40

VEHICLE - IT 2
 START TIME - 800.
 DATE 102080

STOP	SITE	TIME	DELIVER	PICK UP	ORDER	STAY TIME
1	67E	802		8 PALLETS	20	19
2	1621	829	8 PALLETS		20	19
3	1605	850		3 PALLETS	37	10
4	191	902		1 PALLETS	33	7
5	1602	910		1 PALLETS	47	7
				2 PALLETS	48	4
6	1605	923		1 PALLETS	42	7
7	647	959	1 PALLETS		42	7
			2 PALLETS		48	4
8	646	1011	1 PALLETS		47	7
9	X10	1027	1 PALLETS		33	7
10	224	1030	3 PALLETS		37	10
11	1605	1109		10 PALLETS	40	22 *SPLIT
12	X10	1154	10 PALLETS		40	22 *SPLIT

ROUTE ENDED
 LOCATION = 1078
 TIME = 1228
 NO OF PALLETS MOVED = 26

.....
 ORDERS NOT MOVED

ORDER	-49.	2FROM	.1602	TO	1507
ORDER	27.	11FROM	.1603	TO	1172
ORDER	26.	1FROM	.1604	TO	1
ORDER	39.	3FROM	.1605	TO	1606
ORDER	43.	1FROM	.1606	TO	1621
ORDER	-40.	1FROM	.1608	TO	X10
ORDER	41.	1FROM	.1609	TO	23
ORDER	50.	1FROM	.66E	TO	1503
ORDER	21.	1FROM	.67E	TO	49
ORDER	17.	1FROM	.67E	TO	61
ORDER	22.	1FROM	.67E	TO	64
ORDER	18.	1FROM	.67E	TO	647
ORDER	19.	1FROM	.67E	TO	1
ORDER	15.	1FROM	.67E	TO	16
ORDER	16.	1FROM	.67E	TO	23
ORDER	50.	1FROM	.67W	TO	SM
ORDER	7.	6FROM	.198	TO	1172
ORDER	57.	1FROM	.SM	TO	1172
ORDER	35.	2FROM	.1502	TO	1621
ORDER	6.	6FROM	.1172	TO	193
ORDER	5.	12FROM	.1	TO	NWS
ORDER	3.	5FROM	.23	TO	NWS

SAMPLE - AVS2 RUN, SPECIAL ORDERS

AD-A095 729

DAVID W TAYLOR NAVAL SHIP RESEARCH AND DEVELOPMENT CE--ETC F/G 12/2
AUTOMATIC VEHICLE SCHEDULING (AVS) PROGRAMMER'S INSTRUCTION MAN--ETC(U)
FEB 81 R WINCHELL, R MELTON, M NATRELLA
DTNSRDC-81/017

UNCLASSIFIED

NL

3 of 3
86-124



END
DATE
FILMED
3-23-81
DTIC

AVS SPECIAL ORDER PROGRAM

10/20/80

800.0

ORDERS

1	14 PALLETS FROM 67E	TO NWS
2	10 PALLETS FROM 67E	TO SM
3	14 PALLETS FROM 67E	TO 1601A

SPECIAL ORDER TIME = 900.

BUMP OPTION = NO

VEHICLES SELECTED

1	VEHICLE ST 4	CAPACITY = 7 PALLETS.	ROUTE DURATION = 240.0 MINS.
2	VEHICLE ST 5	CAPACITY = 7 PALLETS.	ROUTE DURATION = 240.0 MINS.
3	VEHICLE TT 2	CAPACITY = 14 PALLETS.	ROUTE DURATION = 240.0 MINS.

VEHICLE - ST 4
START TIME - 800.
DATE 102080

```
*****  
STOP SITE TIME DELIVER PICK UP ORDER STAY TIME  
*****  
1 67E 902 7 PALLETS 7 PALLETS 2 3 *SPCL*  
2 SM 907 7 PALLETS 3 PALLETS 2 3 *SPCL*  
3 67E 912 3 PALLETS 7 PALLETS 2 3 *SPCL*  
4 SM 916 3 PALLETS 7 PALLETS 2 3 *SPCL*  
5 67E 920 7 PALLETS 7 PALLETS 3 3 *SPCL*  
6 1601A 932 7 PALLETS 7 PALLETS 3 3 *SPCL*  
7 67E 944 7 PALLETS 7 PALLETS 3 3 *SPCL*  
8 1601A 950 7 PALLETS 7 PALLETS 3 3 *SPCL*
```

ROUTE ENDED
LOCATION =107H
TIME = 1002
NO OF PALLETS MOVED = 24

VEHICLE - TT 2
START TIME - 800.
DATE 102080

```
*****  
STOP SITE TIME DELIVER PICK UP ORDER STAY TIME  
*****  
1 67E 902 14 PALLETS 1 29 *SPCL*  
2 NWS 1015 14 PALLETS 1 29 *SPCL*  
3 1 1129 12 PALLETS 5 26  
4 NWS 1239 12 PALLETS 5 26
```

ROUTE ENDED
LOCATION =107A
TIME = 1350
NO OF PALLETS MOVED = 26

 ORDERS NOT MOVED

ORDER	-49.	2FROM	,1602	TO	1507
ORDER	27.	11FROM	,1603	TO	1172
ORDER	20.	1FROM	,1604	TO	1
ORDER	39.	3FROM	,1605	TO	1606
ORDER	43.	1FROM	,1605	TO	1621
ORDER	-40.	1FROM	,1605	TO	X10
ORDER	41.	1FROM	,1605	TO	23
ORDER	50.	1FROM	,66E	TO	1503
ORDER	21.	1FROM	,67E	TO	49
ORDER	17.	1FROM	,57E	TO	61
ORDER	22.	1FROM	,67E	TO	84
ORDER	18.	1FROM	,57E	TO	647
ORDER	19.	1FROM	,67E	TO	1
ORDER	15.	1FROM	,67E	TO	16
ORDER	16.	1FROM	,67E	TO	23
ORDER	58.	1FROM	,67W	TO	SM
ORDER	7.	6FROM	,198	TO	1172
ORDER	57.	1FROM	,SM	TO	1172
ORDER	35.	2FROM	,1502	TO	1621
ORDER	6.	6FROM	,1172	TO	198
ORDER	3.	5FROM	,23	TO	NWS

SAMPLE - HISTORY FILE UPDATE RUN

AVS HISTORY FILE REPORT
 26JUN79
 SHIFT- 745.0

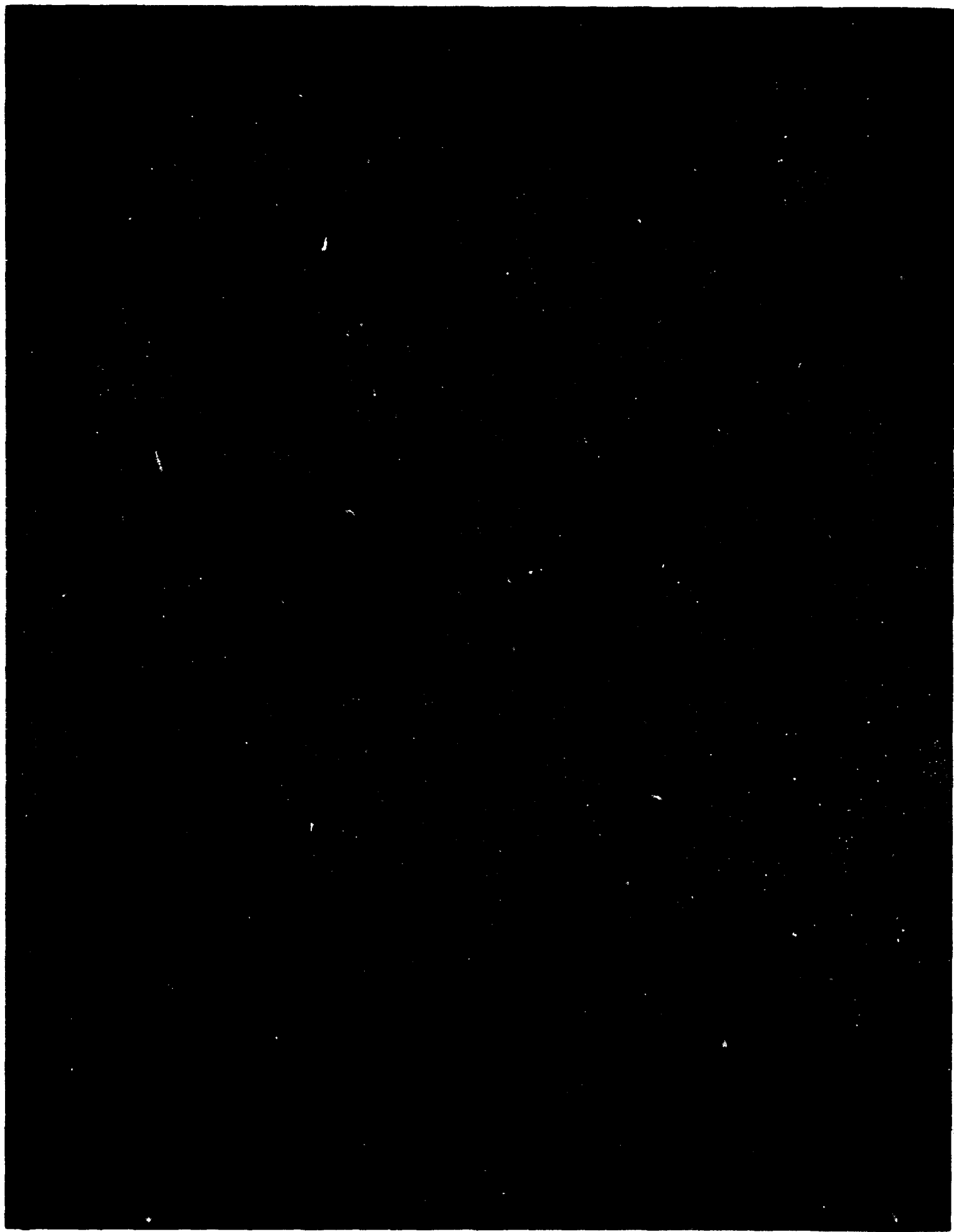
WAREHOUSE	NAME	TRUCK	DELIVERIES		BACKLOG
			REGULAR	EMERGENCY	
1138	ARASF				
	DEYTN	TT2	14	0	12
		TT3	14	0	
	64F	ST1	12	0	12
193	1601R	TT1	23	0	
19A	1138	ST1	12	0	
	1601A	ST1	5	0	
		ST2	7	0	
	1605	TR1	30	0	
45	DEYTN				23
	1138	ST2	24	0	
	193	TT1	14	0	
		TT2	9	0	
	64W	ST1	88	1	
		ST2	5	0	

INITIAL DISTRIBUTION

Copies		Copies	Code	Name
1	NAVMAT 08T246 (LCDR J. Bland)	10	5211.1	Reports Distribution
1	NAVMATO (Code 11)	1	522.1	Unclassified Lib (C)
2	NPGS (Library)	1	522.2	Unclassified Lib (A)
3	FMSO 1 Code 953 (J. Blaylock) 1 Code 953 (Dave Polm) 1 Code 953 (Bill McDevitt)			
1	NAVSUPSCOL (Athens, GA Lib)			
10	NAVSUP 1 SUP 04112B (Sandra Jones) 1 SUP 04141A (Carol Steward) 6 SUP 043 2 SUP 053 (Bill McClure) (CDR Palmatier)			
8	NSC CHARLESTON 4 Code 400 (CDR Dell) 2 Code 61 (Bob Owens) 2 Code 61 (Bob Lee)			
1	NSC San Diego (Mr. Osborne)			
1	NSC Puget Sound (Code 43)			
1	NSC Norfolk			
1	NSC Oakland			
1	NSC Pearl Harbor			
12	DTIC			

CENTER DISTRIBUTION

Copies	Code	Name
1	18	G.H. Gleissner
2	1809.3	D. Harris
1	185	T. Corin
20	187	R. Melton



DATE
FILMED
8-8