US Army Corps of Engineers
Hydrologic Engineering Center

GENERALIZED COMPUTER PROGRAM



# SIDEDT

# Structure Inventory for Damage Analysis Edit Program

## User's Manual

**December 1983**



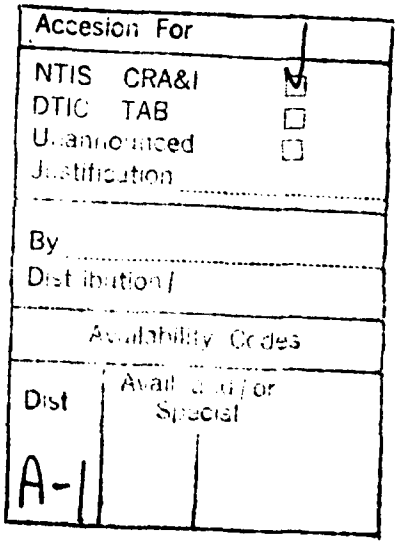


CPD-44

# SIDEDT

# Structure Inventory for Damage Analysis Edit Program

## User's Manual




**December 1983**

Original: June 1977
Revised: August 1979, February 1984

Hydrologic Engineering Center
US Army Corps of Engineers
609 Second Street
Davis, CA 95616

(916) 756-1104



CPD-44

STRUCTURE INVENTORY FOR DAMAGE ANALYSIS EDIT PROGRAM
USERS MANUAL

## Table Of Contents

## Table of Contents (Continued)



## Appendices



## Tables

# STRUCTURE INVENTORY FOR DAMAGE ANALYSIS EDIT PROGRAM
# USERS MANUAL

## I. INTRODUCTION

### 1. Background and Overview

The Structure Inventory for Damage Analysis Edit computer program (SIDEDT) is a companion program to the Structure Inventory for Damage Analysis computer program (SID) (Hydrologic Engineering Center, 1982). It is designed to assist in the management and maintenance of stage-damage function files and structure inventory files which are both used as input to the SID program.

An overview of the edit capablities of the SIDEDT program are presented herein. In addition, the input necessary for program execution and related output are described in detail. Sample input and output have also been included to demonstrate the program capabilities and to assist the user in preparing input data for the program.

The SIDEDT program was originally designed and written by the Environmental Systems Research Institute (ESRI) under contract to the New York District, U.S. Army Corps of Engineers. After significant modification and extension, the program is maintained and distributed by the Hydrologic Engineering Center (HEC), U.S. Army Corps of Engineers, 609 Second Street, Davis, California 95616. HEC should be contacted for any questions regarding the program's use or availability.

### 2. Job-size Limitations

There is virtually no limit to the number of stage-damage functions or structures that this program can process. When using random access files, the file-size limitations are the same as for the SID program. For example, random access files cannot contain more than 2000 stage-damage functions or when processing structure inventory files, the random files cannot contain more than 300 damage reaches, with a maximum of 5000 structures per damage reach.

### 3. Hardware and Software Requirements

The SIDEDT program was developed on a HARRIS 500 minicomputer located at the HEC. The program is also maintained at the Lawrence Berkeley Laboratory (LBL), University of California, Berkeley, California, and on the national Corps of Engineers computer vendor equipment, currently the CDC Cybernet Computer Services. Both LBL and CDC use Control Data Corporation (CDC) computer equipment. In general, SIDEDT is compatible with other major computer systems. Difficulties in installation should be reported to the Hydrologic Engineering Center.

| | |
|---|---|
| Program Language: | FORTRAN IV (ANSI Standard) |
| Memory Requirement: | 124,000 Words (octal) of core at CDC (word size: 60 bits) |

Special Library Functions: Random access file read/write routines
Character manipulation routines

Printer Positions: 132

Tape/Disk Assignments:

TAPE5: The local disk file name which represents the card reader; i.e., the primary input device.

TAPE6: The local disk file name which represents the line printer, i.e., the primary output device.

TAPE8: The local disk or tape file name used when the input structure data or damage function data resides in card image on a sequential disk or tape device.

TAPE9: The local disk or tape file name used when merging structure or damage function data with TAPE8.

TAPE10: The local disk or tape file name used when updating the structure or damage function data on TAPE8.

TAPE11: The local disk file name used by the SIDEDT program as a formatted temporary work file or alternative output file (See TAPE12.)

TAPE12: The local disk file name used when the output structure or damage function data is to reside in card image on a sequential disk file.

TAPE92: The local disk file name used when the damage function selection data (DF Cards) are output from the SIDEDT program.

TAPE98: The local disk file name used when the damage function data (DF, DP, PC, or (DD) cards) reside on a random access disk file. It may be used as both an input and output file.

TAPE99: The local disk file name used when the structural information (SL, SD, SO, SS, SA cards) reside on a random access disk file. It may be used as both an input and output file.

## II. PROGRAM CAPABILITIES

### 1. Overview

The SIDEDT program has been developed to assist in locating and correcting errors in a structure inventory file or a damage function file. In addition, it provides a wide range of options to aid in file merging, data selection, data manipulation, data modification and output selection.

Data are input and output as character files containing attributes descriptive of individual structures, such as structure identification, reference flood elevation, etc., or a specific stage-damage function, such as damage function identification, beginning stage, etc.. SIDEDT provides the capability of selectively reading all or some of these attributes from the input file.

### 2. Commands

The capabilities of the SIDEDT program are reflected in the nine available commands. The commands are briefly described below:

| Command | Description |
|---|---|
| READ | Defines the type of input file to be read, either structure or damage function data. |
| LIST | Generates a listing of specific attributes. |
| MERGE | Merges two like-type files. |
| UPDATE | Replaces or adds attribute fields in the files. |
| NEWFIELD | Creates a temporary new attribute field. |
| WINDOW | Selects structures based on geographic location. |
| PULL | Selects a subset of the input file based on attribute values. |
| MODIFY | Performs arithmetic operations on attributes and saves the results. |
| END | Ends the SIDEDT run. |

Each command is described in its own sub-section in Chapter III of this manual. Simple examples of each command are given in each command description. Detailed examples are provided in APPENDIX A.

## 3. Command Syntax

Execution of the SIDEDT program options is selected by a series of commands and keywords. Each command requires a specific English-like syntax that must be followed. The program is unforgiving and will terminate if it encounters a syntax error. To allow some flexibility, all command entries are free-format. In free-format, the command entries are made on one or more cards in correct sequence, with one or more blank spaces separating each entry. Entries for a single command may be continued on additional cards by entering a space and a slash (/) as the last entries on a card. The end of a command's entry is signified by not entering the slash on the last card (or line image) in the command string.

## 4. File Description

The input and output files to the SIDEDT program consist of a collection of records that correspond to the structure cards or damage function cards defined in the SID Users Manual (Hydrologic Engineering Center, 1982). Each record is composed of a group of attribute fields defined by the variable name for that field in the SID Users Manual. Associated with each attribute is a data type (character, integer, or real) and a beginning and ending column number. These column numbers are relative to the entire record, not the specific card in the record. For example, in a structure inventory record, the damage category (IDCAT) resides on the SD card in card columns 17-24. Because the SD card is the second card in the structure record, IDCAT is in columns 97-104 in the record as a whole. See APPENDICES B and C.

Table 1 lists the predefined structure inventory record attribute field names, beginning and ending columns, and data types.

Table 2 lists the same information for the damage function file.

## 5. File Units

All of the valid commands except READ and END require specification of at least one file unit number and perhaps two or three unit numbers. These numbers identify which file is used as input and which file is to be used as output. These unit numbers also implicity define whether the file is a sequential file or a random file. For example, the command "LIST FROM 8 ..." identifies TAPE8 as the input file and TAPE8 is a sequential file; however, if the command is "LIST FROM 98 ...", the program would expect a random file because TAPE98 defines a random access damage function file.

TABLE 1

STRUCTURE INVENTORY RECORD
(SL, SD and Optional SO, SS and SA cards)

| SL Card | | | | SD Card | | | | SO Card (Optional) | | | | SS Card (Optional) | | | | SS Card (Optional) | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Variable Name | 1/ B | 2/ E | 3/ Data Type | Variable Name | B | E | Data Type | Variable Name | B | E | Data Type | Variable Name | B | E | Data Type | Variable Name | B | E | Data Type |
| KODE1 | 1 | 2 | C | KODE2 | 81 | 82 | C | KODE3 | 161 | 162 | C | KODE4 | 241 | 242 | C | KODE5 | 321 | 322 | C |
| IDRCH1 | 3 | 8 | C | IDRCH2 | 83 | 88 | C | IDRCH3 | 163 | 168 | C | IDRCH4 | 243 | 248 | C | IDRCH5 | 323 | 328 | C |
| IBLDG1 | 9 | 16 | C | IBLDG2 | 89 | 96 | C | IBLDG3 | 169 | 176 | C | IBLDG4 | 249 | 256 | C | IBLDG5 | 329 | 336 | C |
| ROWN | 17 | 24 | R | IDCAT | 97 | 104 | C | IDBS | 177 | 179 | C | YC | 257 | 260 | R | RESID1 | 337 | 344 | C |
| COLE | 25 | 32 | R | ID1FS | 105 | 107 | C | VBS | 180 | 184 | R | SF | 261 | 262 | C | RESID2 | 345 | 352 | C |
| ADJ | 33 | 40 | R | V1FS | 108 | 112 | R | IDBC | 185 | 187 | C | TG | 263 | 264 | C | ADDR1 | 353 | 360 | C |
| STOPO | 41 | 48 | R | ID1FC | 113 | 115 | C | VBC | 188 | 192 | R | CG | 265 | 266 | C | ADDR2 | 361 | 368 | C |
| DELTZ | 49 | 56 | R | V1FC | 116 | 120 | R | IDBO | 193 | 195 | C | NG | 267 | 268 | I | ADDR3 | 369 | 376 | C |
| DELTB | 57 | 64 | R | ID1FO | 121 | 123 | C | VBO | 196 | 200 | R | BG | 269 | 270 | R | CITY1 | 377 | 384 | C |
| DELTG | 65 | 72 | R | V1FO | 124 | 128 | R | IDAS | 201 | 203 | C | BT | 273 | 274 | C | CITY2 | 385 | 392 | C |
| IFUNC | 73 | 74 | I | IADDR1 | 129 | 136 | C | VAS | 204 | 208 | R | BC | 275 | 276 | C | IZIP | 393 | 400 | I |
| NEWSTR | 75 | 75 | I | IADDR2 | 137 | 144 | C | IDAC | 209 | 211 | C | BSIZE | 277 | 280 | R | | | | |
| | | | | IADDR3 | 145 | 152 | C | VAC | 212 | 216 | R | MWB | 281 | 282 | I | | | | |
| | | | | IADDR4 | 153 | 160 | C | IDAD | 217 | 219 | C | WAB | 283 | 285 | R | | | | |
| | | | | | | | | VAO | 220 | 224 | R | WBF | 286 | 288 | R | | | | |
| | | | | | | | | | | | | NOB | 289 | 290 | I | | | | |
| | | | | | | | | | | | | DAD | 291 | 293 | R | | | | |
| | | | | | | | | | | | | OBF | 294 | 296 | R | | | | |
| | | | | | | | | | | | | FC | 299 | 300 | C | | | | |
| | | | | | | | | | | | | FSIZE | 301 | 304 | R | | | | |
| | | | | | | | | | | | | NWF | 305 | 306 | I | | | | |
| | | | | | | | | | | | | WAF | 307 | 309 | R | | | | |
| | | | | | | | | | | | | WOF | 310 | 312 | R | | | | |
| | | | | | | | | | | | | NOF | 313 | 314 | I | | | | |
| | | | | | | | | | | | | OAF | 315 | 317 | R | | | | |
| | | | | | | | | | | | | ODF | 318 | 320 | R | | | | |

NOTE: Footnotes apply for all cards

1/ Beginning Column
2/ Ending Column
3/ Data Type; C = alpha-numeric characters,
R = real (decimal) numbers, I = integer numbers

TABLE 2

DAMAGE FUNCTION RECORD
(DF, DP, PC/DD Cards)

| DF Card | | | | DP Card | | | | DP Card (Optional) | | | | PC/DD Card | | | | PC/DD (Optional) | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Variable Name | 1/ B | 2/ E | Data3/ Type | Variable Name | B | E | Data Type | Variable Name | B | E | Data Type | Variable Name | B | E | Data Type | Variable Name | B | E | Data Type |
| KODE1 | 1 | 2 | C | KODE2 | 81 | 82 | C | KODE3 | 161 | 162 | C | KODE4 | 241 | 242 | C | KODE5 | 321 | 322 | C |
| I1 | 6 | 8 | C | SAGE1 | 83 | 88 | R | SAGE11 | 163 | 168 | R | PERCNT1 | 243 | 248 | R | PERCNT11 | 323 | 328 | R |
| NSTAG | 9 | 16 | I | SAGE2 | 89 | 96 | R | SAGE12 | 169 | 176 | R | PERCNT2 | 249 | 256 | R | PERCNT12 | 329 | 336 | R |
| IDF | 17 | 24 | I | SAGE3 | 97 | 104 | R | SAGE13 | 177 | 184 | R | PERCNT3 | 257 | 264 | R | PERCNT13 | 337 | 344 | R |
| IDFILE | 25 | 32 | I | SAGE4 | 105 | 112 | R | SAGE14 | 185 | 192 | R | PERCNT4 | 265 | 272 | R | PERCNT14 | 345 | 352 | R |
| | | | | SAGE5 | 113 | 120 | R | SAGE15 | 193 | 200 | R | PERCNT5 | 273 | 280 | R | PERCNT15 | 353 | 360 | R |
| | | | | SAGE6 | 121 | 128 | R | SAGE16 | 201 | 208 | R | PERCNT6 | 281 | 288 | R | PERCNT16 | 361 | 368 | R |
| | | | | SAGE7 | 129 | 136 | R | SAGE17 | 209 | 216 | R | PERCNT7 | 289 | 296 | R | PERCNT17 | 369 | 376 | R |
| | | | | SAGE8 | 137 | 144 | R | SAGE18 | 217 | 224 | R | PERCNT8 | 297 | 304 | R | PERCNT18 | 377 | 384 | R |
| | | | | SAGE9 | 149 | 152 | R | SAGE19 | 225 | 232 | R | PERCNT9 | 305 | 312 | R | PERCNT19 | 385 | 392 | R |
| | | | | SAGE10 | 153 | 160 | R | SAGE20 | 233 | 240 | R | PERCNT10 | 313 | 320 | R | PERCNT20 | 393 | 400 | R |

NOTE: Footnotes apply for all cards.

1/ Beginning Column
2/ Ending Column
3/ Data Type: C = alpha-numeric characters;
R = real (decimal) numbers, I = integer numbers

# III. USER INPUT DESCRIPTION

## 1. Overview

This chapter provides a detailed description of the SIDEDT data input requirements.

Certain notations are used throughout this chapter with consistent meaning. UPPERCASE indicates a statement keyword that is to be written as shown. Lowercase enclosed in brackets, [lowercase], indicates a name or number that is to be supplied by the user.

## 2. READ Command. Required.

The READ command specifies what type of data is to be processed, i.e., structure inventory data or damage function data. This command must be the first card in the users input stream. The format of the READ command is as follows:

| | |
|---|---|
| READ TYPE | [data type] |
| | Where data type may be one of two keywords: |
| | STRUCT<br>for structure inventory data |
| | or |
| | DAMAGE<br>for damage function data. |

Example:

READ TYPE STRUCT is the first card required when processing a structure inventory file.

## 3. LIST Command. Optional.

The LIST command writes specified attributes to the printer to create a listing of attribute values. LIST can list any of the fields from the following data files: TAPE8, TAPE9, TAPE11, TAPE12, TAPE98, TAPE99. The LIST command is optional and may be used anywhere after the READ command, as many times as desired.

The listed attributes will appear in the listing as columns headed by the field names. If all of the listed attributes are wider than a printer page, the attributes will wrap around onto a new line in the listing and be unreadable.

The LIST command is used by specifying the file that contains the attributes to be listed and the names of the attribute fields that are to appear in the listing. If no attribute field names are specified, LIST will list all of the attributes in the file. The format of the LIST command is as follows:

LIST FROM [unit # ] FIELDS [field name] [field name] . . .

Example:

LIST FROM 8 FIELDS IDRCH1 IBLDG1 STOPO IDCAT

will list the damage reach id, building id, first floor elevation and damage category for each structure record in file TAPE8.

LIST FROM 98 FIELDS

will list the entire random damage function file, TAPE98.

4. MERGE Command. Optional.

The MERGE command merges two identically formatted sequential structure or damage function files. The two input data files are TAPE8 and TAPE9. The output file may be any one of the following data files: TAPE12, TAPE98, or TAPE99.

MERGE automatically inserts records from the second (subordinate) file into its correct position in the first (master) file. If there is a duplicate record, the program will replace the record in the master file with the record from the subordinate file.

MERGE requires that the files being merged have been sorted into the same order based on the choice of keys.

The MERGE command is used by specifying the two input files, the output file and the key sort field. The format of the MERGE command is as follows:

MERGE FROM 8 9 TO [output unit# ] KEYS [keysort field name]

Example:

MERGE FROM 8 9 TO 12 KEYS IT

will merge two sequential damage function files (TAPE8 and TAPE9) using the damage function identifier as the sort key to output file, TAPE12, producing a sequential file containing all the damage function data from TAPE8 and TAPE9 sorted by damage function identifier.

5. NEWFIELD Command. Optional.

The NEWFIELD command temporarily defines a new field, within the structure or damage function file's records. The NEWFIELD command may be used to define a new field to contain an additional attribute generated by

the UPDATE or MODIFY commands or to redefine part of an existing field as a new field to allow access to part of an attribute code. This command must be input before the command that utilizes the new attribute field. Each NEWFIELD command defines one new field but the command may be used as many times as needed.

A new field may be defined anywhere within the records defined in Tables 1 and 2. There are two cautionary notes. First, NEWFIELD cannot extend the record length of 400 columns. Second, it is possible to define a new field in columns used by existing attribute fields. Therefore, if data is written to the new field, the existing data will be written over.

The NEWFIELD command is used by specifying the field name, the first and last columns of the field, and the data type (INTEGER, REAL, CHAR) of the field. One new field is defined on each NEWFIELD command. The format is as follows:

NEWFIELD [field name] [first col.] [last col.] [date type]

Example:

NEWFIELD allows new fields to be defined over existing fields so that parts of fields may be read by the program commands. For example, if the first two characters of the damage reach identifier are a code for the tributary and the last four characters are the damage reach number, NEWFIELD may be used to access all damage reaches within a particular tributary. The command would be as follows:

NEWFIELD TRIB 3 4 CHAR

Note that either field, IDRCH1 or TRIB, may now be used.

6. UPDATE Command. Optional.

The UPDATE command replaces existing attribute values with specified values or adds attribute values to blank fields. UPDATE updates one field at a time, either adding or replacing its values. Update data are input in a special update file which must be present if the UPDATE command is used. The update file structure is detailed after this command. If a new attribute is to be added, NEWFIELD must precede UPDATE to define the new attribute field. UPDATE is optional and may be used as many times as needed.

Updates are made by locating the record(s) that have the match value entered in the update file's match field and replacing the value in the specified file's attribute field with the value in the corresponding update file's attribute field. This is nothing more than matching, for example, structure identifications between the two files so that an update intended for a specific structure is not made to another. Note - Both the structure or damage function file and the update file must be sorted by the match field before UPDATE is used. If the files are not sorted in the same order, the results are unpredictable.

A single update may be applied to many records by entering a match value in the update file that applies to multiple records. For example, a single entry in the update file may 'match' against damage reach number, so that every structure within the specified damage reach would have the same update value written to the specified attribute field in the structure file.

The UPDATE command is used by specifying the input file, the update file, the output file, the location of the match field in the update file, the name of the match field in the structure or damage function file, the location of the field containing the update value in the update file, and the name of the attribute field in the structure or damage function file to receive the value. The locations of the match and update fields must be specified as beginning and ending columns. The format of the UPDATE command is as follows:

UPDATE FROM 8 10 TO [output unit #]

MATCH [first col. of update file match field] [last col. of update file match field] WITH [input file match field name]

MOVE [first col. of update file attribute field] [last col. of update file attribute field] INTO [input field field name]

Example:

```
UPDATE FROM 8 10 TO 12 /
MATCH 1 8 WITH IBLDG1 /
MOVE 11 15 INTO V1FC
```

NOTE: The slash (/) is used to indicate that the command is continued on another line.

This command will locate specific structures and replace their contents value (V1FC) with the value found in the update file. The match field is the structure identification, which is IBLDG1 in the structure file and columns 1-8 in the update file. The attribute field is contents value which is V1FC in the structure file and columns 11-15 in the update file.

UPDATE FILE

The update file contains one 80-column card image for each record that will be updated. Each card image must have at least two fields, a 'match' field and one or more attribute fields. The match field contains a value that must match the value in a specified attribute field within the structure or damage function file before the update takes place. Cards in the update file must be sorted by the match field in the same order as the structure or damage function file.

The attribute field(s) contains the new or replacement value that will be written to the records identified by the match field. Although the update file may contain as many attribute fields as can fit on a card, the UPDATE command can only apply one attribute update at a time.

The format of the update file is as follows:

[match field] [first attribute field] [second attribute field] . . .

For example, if the value entered for the 'contents value' attribute, VIFC, in the structure file for structure AB123456 is to replaced with a correct value of 1300, the card might appear as follows:

AB123456 1300.

7. WINDOW Command. Optional.

The WINDOW command selects structures to be written to the output file based on whether they fall within a specified geographic window. The window is defined by minimum and maximum northing and easting UTM coordinates. WINDOWed structures are output in the order in which they are found. Of course, the structure inventory file must contain geographic coordinates for this command to work. The WINDOW command is optional and may be used anywhere after the READ command.

The WINDOW command is used by specifying the file that contains the structures to be windowed, the output file unit number, the names of the fields that contain the X-coordinates (easting values) and Y-coordinates (northing values), and the minimum and maximum coordinates that define the window. The format of the WINDOW command is as follows:

```
WINDOW FROM [input file unit#] TO [output file unit #]
XCOOR COLE  YCOOR ROWN
XMIN [coordinate]  XMAX [coordinate]  XMIN [coordinate]  YMAX [coordinate]
```

Example:

The following command would be used to window from a sequential input file to a random output file:

```
WINDOW FROM 8 TO 99  XCOOR   COLE   YCOOR   ROWN   /

    XMIN          587335.2      /
    XMAX          637229.8      /
    YMIN          526304.2      /
    YMAX          546297.8
```

8. PULL Command. Optional.

The PULL command selects records to be written to the output file based on their attribute values. Each record's attributes, structure or damage function data, are subjected to tests which, if passed, that record is selected for output. Tests are stated as logical expressions that incorporate an 'IF' test, logical operators (e.g. AND, OR, NOT) and relational operators (e.g. equal to (EQ), not equal to (NE), greater than

(GT), less than (LT), etc.). PULLed structures are output in the order in which they are found. PULL is optional and may be used anywhere after the READ command, as many times as desired.

The PULL command is used by specifying the file that contains the input structure or damage function file, the output file unit number and up to twenty (20) logical expressions. At least one logical expression must be entered. In the PULL command, if any of the logical expressions are true, the record will be selected and written to the output file.

Logical expressions have strict format and logic rules that must be followed. The format of a logical expression is:

IF [variable] [relational operator] [variable] AND
[variable] [relational operator] [variable] AND . . .

The PULL command has one test; 'IF'. This type of test is 'IF something is true, then action', where the action is to either write the record to the output file or subject it to another test. An 'AND' extends the test begun by an 'IF'. A logical expression that has an 'AND' in it is of the form: 'IF something is true, AND something else is true, then action'. Up to 20 'AND's may be included in a logical expression. Note that it is not required to use 'AND' in a logical expression. A logical expression must always start with and 'IF', and 'ANDs' (if used) must always follow 'IFs'. For a test extended with 'ANDs', all of the 'IF' and 'ANDs' must be true for the test as a whole to be true. If one of the conditions is false, the entire string of 'IF' and 'ANDs' is false.

Variables in a logical expression may be an attribute's field name (the attribute value is tested) or a constant value specified by the user. These may be integers, real numbers or character strings. If a character string is used in a logical expression, it must be enclosed in single quotes (') on the Harris computers and enclosed in double quotes (") on CDC computer equipment.

Relational operators test one variable against another to determine if they are equal, unequal, etc.. The PULL command has six relational operators. Each is listed below as it is entered (uppercase letters) and what is means (lowercase letters).

| | | |
|---|---|---|
| EQ - equal | LT - less than | LE - less than or equal |
| NE - not equal | GT - greater than | GE - greater than or equal |

The format of the PULL command is as follows:

PULL FROM [input unit#] TO [output unit#] BY

[logical expression] [logical expression] . . .

Example:

To create a subfile of only those structures in damage reach # 1 whose first-floor elevation is greater than 500.0, the command would be as follows:

PULL FROM 8 TO 12 BY IF IDRCH1 EQ ' DR1' AND STOPO GT 500.0

9. MODIFY Command. Optional.

The MODIFY command selects records from a structure or damage function file using the same logic as the PULL command and then performs arithmetic operations on the contents of selected attribute fields and/or moves the contents of an attribute field. An attribute field may be moved to another attribute field, replacing whatever was previously there, or to a temporary attribute for storage. The temporary attribute may then be used for additional operations. For example, a constant can be added to an attribute value storing the result in a temporary variable and then the temporary variable can be divided by a constant, storing the result in the original attribute or a different one. All of the selected attributes will be subjected to the specified operation. Note that the PULL or WINDOW commands may be used to select specific records before the MODIFY command is used. MODIFYed records will be output in the order they are found.

The MODIFY command is used by specifying the input file unit number, the output file unit number, a logical expression to select records and up to twenty (20) arithmetic statements to operate on the attributes. At least one logical expression and one arithmetic expression must be entered.

The format of an arithmetic expression is:

[operation] [variable] XX [variable] GIVING [attribute]

There are five types of operations. Each is shown below as it is used in a statement.

ADD [variable] TO [variable] GIVING [attribute]
SUBTRACT [variable] FROM [variable] GIVING [attribute]
MULTIPLY [variable] BY [variable] GIVING [attribute]
DIVIDE [variable] INTO [variable] GIVING [attribute]
MOVE [variable] TO [attribute or temporary attribute]

Variables are the same as in the PULL command; they may be an attribute field name, an integer or real number, or a character string. If a character string is used, it must be enclosed in single quotes (') on Harris computers and in double quotes (") on CDC computer equipment.

Each arithmetic statement will write its results to the specified attribute field. Either an existing attribute field or a special temporary attribute may be specified. The special temporary attribute is specified by an asterisk (*). Note that the results written to an existing attribute field will replace whatever is already there.

Each MODIFY command must start with at least one logical expression to select records, although the user may use an expression that will select all records, if desired. Logical expressions are linked to arithmetic statements by entering "THEN" between them, as shown below:

[logical expression] THEN [arithmetic statement]

The format of the MODIFY command is as follows:

```
MODIFY FROM [input file unit #] TO [output file unit #] BY
[logical expression] [logical expression] . . . THEN [arithmetic
statement] [arithmetic statement] . . .
```

Example.

To subtract 1 foot from the reference flood elevation for all structures in damage reach # 1, use the following command:

```
MODIFY FROM 99 TO 12 BY /
IF IDRCH1 EQ ' DR1' /
THEN SUBTRACT 1.0 FROM ADJ GIVING ADJ
```

10. END Command. Required.

The END command is required to signify the end of the SIDEDT run. It is always the last command entered.

# IV. TEST PROBLEMS

## 1. Purpose and Overview

The test problems in Appendix A are included to illustrate detailed examples of the input requirements of the Structure Inventory for Damage Analysis Edit (SIDEDT) program. The problems are also intended for use in verification of distributed program code. Six problems are presented in the appendix; to the extent possible, subsequent problems expand on the previous problem. The purposes of the problems are summarized below.

Test problem 1. This problem lists selected structure attributes from a sequential structure file. It merges two sequential structure files, and lists selected structure attributes.

Test problem 2. This problem updates the damage function attribute for specific structures in a sequential structure file. It also changes the reference flood elevation for the same structures. Then, all structures values are increased by 15% to reflect a price level change. Finally, the relevant structure attributes will be listed.

Test problem 3. A sequential structure file will be windowed by geographic coordinates. In this particular file, the damage reach attribute contains a code for the tributary and a code for the damage reach number. All structures located along a selected tributary will be listed.

Test problem 4. This problem uses a sequential damage function file to illustrate the replacing of an entire damage function with a new one. A bad percent damage value will be corrected and selected damage function identifiers and stage-percent damage attributes will be listed.

Test problem 5. This problem demonstrates how to create a random damage function file from a sequential damage function file. It will also create the TAPE92 file required to accompany the random file when executing the SID program. Selected damage function attributes will be listed from the random damage function file.

Test problem 6. This problem creates a random access structure file which is a subset of the sequential structure file used in Test problem 1. The structure attributes in the random structure file will then be modified and listed.

## 2. Structure Inventory File For Test Problems (SFILE1)

```
SL080902     EMR                      272     263     -.01        0        0 00
SD080902     EMR     EMRL72
SL080909     EMR                      280     273     -.01        0        0 00
SD080909     EMR     EMRL79
SL080909     TRN                    280.0     285        0        0        0 00
SD080909     TRN     TRN7T7
SL080909   AC001 4523100  495725      280     286      -10                  10
SD080909   AC001     COMX05   29GAA    35          12-1 52-24   16   99GAS STATION
SS080909   AC001      11 4 2 1   1 4  16 1 16 -7 2 21-10   4  1625 15 +4 8 70  0
SL080909   AC002 4523100  495845      280     291       -3                  10
SD080909   AC002     COMX04   52RAF    30           12 55-11   15   77 RESTAURANT
SS080909   AC002      11 1 2 1   1 4  15 4  3 -3           1  1516 20 +3 5 21  0
SL080909   AR001 4522974  495628      280     274       -2                  10
SD080909   AR001     RESZ05   40Z06 22.1        TAXMAP12-1 52-31              15
SS080909   AR001      10 1 1 1   1 4   7 6  2 -2 1 20 -2   1   712 12 +3 2 24  0
SL080909   AR002 4522952  495619      280     274       -2                  10
SD080909   AR002     RESZ05   40Z06 22.1        TAXMAP12-1 52-32              15
SS080909   AR002      10 1 1 1   1 4   7 6  2 -2 1 20 -2   1   712 12 +3 2 24  0
SL080909   AR005 4522886  495592      280     275       -3                  10
SD080909   AR005     RESZ07   40Z08 22.1        TAXMAP12-1 52-34              19
SS080909   AR005       1 7 2 1   1 4   6 5  6 -3 1 20 -2   7   610 15 +3 2 21  0
SL080909   AR022 4522710  495520      280     275     -.01                  10
SD080909   AR022     RESZ13 12.5Z14    9        TAXMAP12 52-16                9
SS080909   AR022       6 1 1 0   4                          1   914 12 +4 2 21  0
SL080909   AR023 4522994  495637      280     275       -3                  10
SD080909   AR023     RESZ07   40Z08 22.1        TAXMAP12-1 52-30              20
SS080909   AR023       1 1 2 1   1 4   6 5  6 -3 1 20 -2   1   610 15 +3 2 21  0
SL081008   BB001                      258     258       -4                  00
SD081008   BB001     TRNB04  336                        4LA HWY
SL081008   BB002                      258   265.0       -4                  00
SD081008   BB002     TRNB04  216                        2LA HWY
SL081008   BC001 4519817  493389      257     258       -6                  10
SD081008   BC001     COMX04    5FAN     9             6  6-22    4   25 PLANT SHOP
SS081008   BC001      11 1 2 1   1 4   4 2  2 -2           1   410 12 -2 2 24  0
SL081008   BC002 4519718  493380      257     257     -.01                  10
SD081008   BC002     COMX06    9AAP     8             6  14-6   10   36     VACANT
SS081008   BC002      11 1 1 0   4                          1  10 8  8 +2 1 24  0
SL081008   BC003 4519675  493375      257     258       -3                  10
SD081008   BC003     COMX04   64HAA    60             6  6-14   27   29   HARDWARE
SS081008   BC003      11 1 2 1   1 7  2710  3 -2 1 12 -2   1  2712 12 +2 5 50  0
SL081008   BC016 4519650  493575      257     258     -.01                  10
SD081008   BC016     COMX03   61AAN    9             6 10-13   17  279 CAR DEALER
SS081008   BC016      11 4 2 0   4                          4  17 8 64 +3 8100  0
SL081008   BC017 4519625  493550      257     259       -1                  10
SD081008   BC017     COMX04   24PAH 11.9             6 20-11    6   39     PRINTER
SS081008   BC017      11 1 2 1   1 4   6 2  1 -1           1   618  6 +3 3 21  0
SL081008   BR001 4519875  493400      257     262       -2                  10
SD081008   BR001     RESZ07   35Z08 19.7          TAXMAP7 6-24                32
SS081008   BR001       1 1 2 1   1 2   6 4  2 -3 1 20 -3   1   6 4  8 +3 2 21  0
SL081008   BR002 4519789  493397      257     260       -2                  10
SD081008   BR002     RESZ07   40Z08 22.1          TAXMAP7 6-21                14
SS081008   BR002       1 1 2 1   1 4   6 5  6 -3 1 20 -2   1   610 15 +2 2 21  0
SL081008   BR003 4519760  493385      257     259       -2                  10
```

```
SD081008  BR003     RESZ07   40Z08 22.1        TAXMAP7 6-20                  42
SS081008  BR003       1 1 2 1   1 4   6 5  6 -3 1 20 -2   1   610 15 +2 2 21  0
SL081008  BR034 4519650  493433     257     260    -.01                  10
SD081008  BR034     RESZ05   30Z06 17.4        TAXMAP6 11-6                  10
SS081008  BR034       2 1 1 0                          1  1011 12 +4 2 21  0
SL081008  BS001 4519846  493393     257     259    -.01                  10
SD081008  BS001     SERX05   88S04   35             6  23-6   30  250     CHURCH
SS081008  BS001      11 7 2 1   1 7  30                    7  3012 12 +3 3 40  0
SL081008  BS002 4519675  493455     257     258      -2                  10
SD081008  BS002     SERX05311.7S05264.9             6  6-12   85 1600 FIRE HOUSE
SS081008  BS002      11 2 2 1   1 2  85 2  2 -2           2  85 5 15 +3 2 18  0
SL081009  CC008 4519506  493418     257     272      -8                  10
SD081009  CC008     COMX04   15HAA   19             6 33-11    8   28REGISTER CO
SS081009  CC008      11 1 2 1   1 4   8          1 21 -8   1   8 2  8 +3 1 21  0
SL081009  CC010 4519425  493350     257     260    -.01                  10
SD081009  CC010     COMX01   10GAB   14             6  1-12   11   64GAS STATION
SS081009  CC010      11 4 1 0   4                        4  1110 12 +3 2 21  0
SL081009  CI098 4519525  493550     257     276     -12                  10
SD081009  CI098     INDX07   57I75  199             6 21 22  100      SCREW MACHS
SS081009  CI098      12 2 1 1   1 4 100 4 12 -7 1 21-12   2 10028 24 +2 1 21  0
SL081009  CI099 4519525  493550     257     276     -12                  10
SD081009  CI099     INDX05  180I74  900             6 21-22  200      SCREW MACHS
SS081009  CI099      13 2 2 1   1 4 200 6 12 -7 1 21-12   2 20040 30 +2 1 48  0
SL081009  CR001 4519543  493381     257     266      -8                  10
SD081009  CR001     RESZ07   40Z08 22.1        TAXMAP6 11-36                 613G
SS081009  CR001       9 1 3 1   1 5   5 2  2 -3 1 20 -2   1   5 5 15 +3 2 24  0
SL081009  CR002 4519537  493387     257     266      -8                  10
SD081009  CR002     RESZ07   40Z08 22.1        TAXMAP6 11-36                 612G
SS081009  CR002       9 1 3 1   1 5   5 2  2 -3 1 20 -2   1   5 5 15 +3 2 24  0
```

3. Structure Inventory Merge File For Test Problems (SFILE2)

```
SL080909   AC003 4522750  495550     280     276    -.01                  10
SD080909   AC003     COMX06    5BAJ                 12 52-15    5   89BAR ' GRILL
SS080909   AC003      11 1 1 0   4                          1   540  8 +4 4 21  0
SL080909   AR013 4522933  495627     280     271      -3                  10
SD080909   AR013     RESZ07   40Z08 22.1        TAXMAP12-1 52-34            13
SS080909   AR013       1 1 2 1   1 4   6 5  6 -3 1 20 -2   1   610 15 +3 2 21  0
SL080909   AR014 4522915  495650     280     271    -.01                  10
SD080909   AR014     RESZ13 12.5Z14    9        TAXMAP12-1 52-34            10
SS080909   AR014       6 1 1 0   4                          1   914 12 +4 2 21  0
SL081008   BC010 4519675  493415     257     258    -.01                  10
SD081008   BC010     COMX01   15DAH   10             6 6-14A    8   50      DINER
SS081008   BC010      11 1 1 0   4                          1   821  7 +3 2 24  0
SL081008   BC011 4519650  493477     257     258      -1                  10
SD081008   BC011     COMX04   33AAF 14.7             6 10-11    7  184 APPLIANCES
SS081008   BC011      11 1 2 1   1 4   7          1 24 -1   1   7 5  8 +1 3 21  0
SL081008   BR029 4519712  493538     257     262      -2                  10
SD081008   BR029     RESZ07   40Z08 22.1        TAXMAP6 6-4                 120
SS081008   BR029       1 1 2 1   1 4   6 5  6 -3 1 20 -2   1   610 15 +3 2 21  0
SL081008   BR031 4519100  493562     257     262      -2                  10
SD081008   BR031     RESZ07   40Z08 22.1        TAXMAP6 7-10                 11
SS081008   BR031       9 1 3 1   1 5   5 2  2 -3 1 20 -2   1   5 5 15 +3 2 24  0
SL081009   CC001 4519475  493350     257     258      -1                  10
SD081009   CC001     COMX05   26MAC   80             6 38-11   14   24CHILD STORE
SS081009   CC001      11 2 2 1   1 4  14          1 12 -2   2  14 5  8 +3 3 24  0
SL081009   CR004 4519500  493425     257     272    -.01                  10
SD081009   CR004     RESZ07   34Z16 19.2        TAXMAP6 11-33                40
SS081009   CR004       7 1 2 1   1 5   8                          1   8 6 16 +4 2 24  0
```

4. Damage Function File For Test Problems (DFILE1)

```
DF    C1      10      1       0
DP    -3      -2      -1      0       1       2       3       4       5       6
DD    0       0       0       0       1       2.4     7.2     12      18      24
DF    C2      10      1       0
DP    -3      -2      -1      0       1       2       3       4       5       6
DD    0       0       1.6     3.2     16      20      22.4    24      25.3    26
DF    C3      7       1       0
DP    0       1       2       3       4       5       6
DD    0       3.6     8.3     14.7    22.2    25      27.7
DF    C4      12      1       0
DP    -1      0       1       2       3       4       5       6       7       8
DP    9       10
DD    0       5       5.1     5.7     5.9     7.9     8       8       8.1     8.1
DD    8.4     8.7
DF    C5      17      1       0
DP    -2      -1      0       1       2       3       4       5       6       7
DP    8       9       10      11      12      13      14
DD    0       .2      .2      2.2     4.5     6.7     7.6     7.6     8.1     8.1
DD    8.4     8.4     8.8     9       9.3     9.3     9.6
DF    C6      8       1       0
DP    0       1       2       3       4       5       6       7
DD    0       2.8     5.5     9.4     11.6    15      16      16.6
DF    C7      12      1       0
DP    0       1       2       3       4       5       6       7       8       9
DP    10      11
DD    0       1.3     1.7     2.2     3.1     3.2     3.3     3.3     3.5     3.5
DD    3.7     8.2
DF    C8      15      1       0
DP    -2      -1      0       1       2       3       4       5       6       7
DP    8       9       10      11      12
DD    0       .1      .2      .5      .7      .9      1.3     1.3     1.7     1.7
DD    2.1     2.3     2.5     2.6     2.8     2.8     3.0
DF    R1      20      0       0
DP    -8      -7      -6      -5      -4      -3      -2      -1      0       1
DP    2       3       4       5       6       7       8       9       10      11
PC            2       5       5       6       6       8       9       11      17
PC    22      28      33      35      38      40      440000  46      48      50
DF    R2      20      0       0
DP    -8      -7      -6      -5      -4      -3      -2      -1      0       1
DP    2       3       4       5       6       7       8       9       10      11
PC    0       2       7       7       10      12      12      14      16      17
PC    22      28      33      39      44      49      55      61      64      71
```

5. Structure Update File For Test Problems (UPDATE)

```
   EMR      L77   275.5
 AC003      XXX     257
 AR022      ZZZ     300
 BC016      XXX     280
 BR034      ZZZ     259
 C1098      X00     272
```

6. Damage Function Replacement File For Test Problems (DFILE2)

```
DF    C1      10       1       0
DF    -2      -1      -0       1       2       3       4       5       6       7
DD     0       0       0       1     3.5     6.4     8.5      11      17      30
DF    R2      20       0       0
DP    -9      -8      -7      -6      -5      -4      -3      -2      -1       0
DP     1       2       3       4       5       6       7       8       9      10
PC     0       1       4       8      12      14      15      16      17      18
PC    23      30      31      38      45      52      60      75      80     100
```

7. Harris 500 Job Control Language (JCL) For Test Problems

```
$JOB,SIDEDT,HLIB,HEC,PRI=4
$
$ TEST PROBLEM 1
$ LIST AND MERGE A SEQUENTIAL STRUCTURE FILE
$
SIDEDTX,INPUT=TPROB1,TAPE8=SFILE1,TAPE9=SFILE2,TAPE12=SFILE3
$
$ TEST PROBLEM 2
$ UPDATE AND MODIFY A SEQUENTIAL STRUCTURE FILE
$
SIDEDTX,INPUT=TPROB2,TAPE8=SFILE3,TAPE10=UPDATE
$
$ TEST PROBLEM 3
$ WINDOW, NEWFIELD AND PULL A SEQUENTIAL STRUCTURE FILE
$
SIDEDTX,INPUT=TPROB3,TAPE8=SFILE3
$
$ TEST PROBLEM 4
$ MERGE AND MODIFY SEQUENTIAL DAMAGE FUNCTION FILES
$
SIDEDTX,INPUT=TPROB4,TAPE8=DFILE1,TAPE9=DFILE2,TAPE12=DFILE3
$
$ TEST PROBLEM 5
$ CREATE A RANDOM DAMAGE FUNCTION FILE
$
GE TAPE98 R G=100
SIDEDTX,INPUT=TPROB5,TAPE8=DFILE3,TAPE92=TAPE92,TAPE98=TAPE98
$
$ TEST PROBLEM 6
$ CREATE A RANDOM STRUCTURE INVENTORY FILE
$
GE TAPE99 R G=200
SIDEDTX,INPUT=TPROB6,TAPE8=SFILE3,TAPE99=TAPE99
```

# APPENDIX A

# TEST PROBLEMS

# Appendix A

## TEST PROBLEM 1

## LIST AND MERGE A SEQUENTIAL STRUCTURE FILE

### 1. Problem Purpose

This example represents the basic program option of listing selected attributes from a sequential structure inventory file. It also demonstrates how to merge two sequential files that have been created by other means and are sorted by increasing structure identification code.

### 2. List of Input Cards for the Run

```
READ TYPE STRUCT
LIST FROM 8 FIELDS IDRCH1 IBLDG1 ADJ STOPO IADDR1 IADDR2 IADDR3 IADDR4
LIST FROM 9 FIELDS IDRCH1 IBLDG1 ADJ STOPO IADDR1 IADDR2 IADDR3 IADDR4
MERGE FROM 8 9 TO 12 KEYS IBLDG1
LIST FROM 12 FIELDS IDRCH1 IBLDG1 ADJ STOPO IADDR1 IADDR2 IADDR3 IADDR4
END
```

### 3. Output Description

The program echoes back every input card read (item 1). To the right of each input card is printed a program assigned sequence number (item 2). When the list option is selected, the SIDEDT program will always go to the top of the next page before printing the list (item 3). The column headings are chosen from the requested attribute names (item 4). After each listing, a report of how many records were processed is printed (item 5). Following the MERGE command, the SIDEDT programs reports how many records were read from both files and written to the output file (item 6).

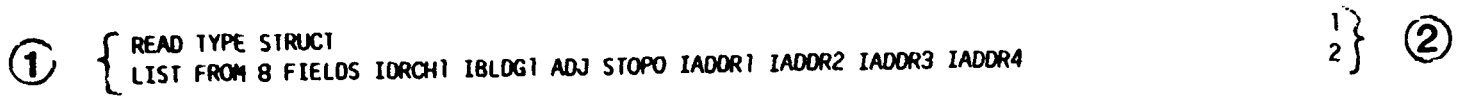
### 4. Program Output

The following pages are reproductions of the program execution.

```
*********************************************************
*       SID EDIT PROGRAM                                *
*       USERS MANUAL -DRAFT IN PROGRESS                 *
*       UPDATED JULY 1983                               *
*                                                       *
*  RUN DATE 23 SEP 83  TIME  12:53:55                   *
*********************************************************
```

```
****************************************
*  U.S. ARMY CORPS OF ENGINEERS        *
*  THE HYDROLOGIC ENGINEERING CENTER   *
*  609 SECOND STREET, SUITE B          *
*  DAVIS, CALIFORNIA  95616            *
*  (916) 440-2105 (FTS) 448-2105       *
****************************************
```

```
 SSSSS   IIIIIII  DDDDDD   EEEEEEE  DDDDDD   TTTTTTT
S     S     I     D     D  E        D     D     T
S           I     D     D  E        D     D     T
 SSSSS      I     D     D  EEEEE    D     D     T
      S     I     D     D  E        D     D     T
S     S     I     D     D  E        D     D     T
 SSSSS   IIIIIII  DDDDDD   EEEEEEE  DDDDDD     T
```


①
READ TYPE STRUCT
LIST FROM 8 FIELDS IORCH1 IBLOG1 ADJ STOPO IADDR1 IADDR2 IADDR3 IADDR4
1
2
②

③ SIDEDT PROGRAM

| ④ IDRCH1 | IBLDG1 | ADJ | STOPO | IADDR1 | IADDR2 | IADDR3 | IADDR4 |
|---|---|---|---|---|---|---|---|
| 080902 | EMR | 272 | 263 | | | | |
| 080909 | EMR | 280 | 273 | | | | |
| 080909 | TRN | 280.0 | 285 | | | | |
| 080909 | AC001 | 280 | 286 | 12-1 52 | -24 16 | 99GAS | STATION |
| 080909 | AC002 | 280 | 291 | 12 55 | -11 15 | 77 RE | STAURANT |
| 080909 | AR001 | 280 | 274 | TAXMAP12 -1 | 52-31 | | 15 |
| 080909 | AR002 | 280 | 274 | TAXMAP12 -1 | 52-32 | | 15 |
| 080909 | AR005 | 280 | 275 | TAXMAP12 -1 | 52-34 | | 19 |
| 080909 | AR022 | 280 | 275 | TAXMAP12 | 52-16 | | 9 |
| 080909 | AR023 | 280 | 275 | TAXMAP12 -1 | 52-30 | | 20 |
| 081008 | BB001 | 258 | 258 | 4LA | HWY | | |
| 081008 | BB002 | 258 | 265.0 | 2LA | HWY | | |
| 081008 | BC001 | 257 | 258 | 6 6 | -22 4 | 25 PL | ANT SHOP |
| 081008 | BC002 | 257 | 257 | 6 1 | 4-6 . 10 | 36 | VACANT |
| 081008 | BC003 | 257 | 258 | 6 6 | -14 27 | 29 | HARDWARE |
| 081008 | BC016 | 257 | 258 | 6 10 | -13 17 | 279 CA | R DEALER |
| 081008 | BC017 | 257 | 259 | 6 20 | -11 6 | 39 | PRINTER |
| 081008 | BR001 | 257 | 262 | TAXMAP7 | 6-24 | | 32 |
| 081008 | BR002 | 257 | 260 | TAXMAP7 | 6-21 | | 14 |
| 081008 | BR003 | 257 | 259 | TAXMAP7 | 6-20 | | 42 |
| 081008 | BR034 | 257 | 260 | TAXMAP6 | 11-6 | | 10 |
| 081008 | BS001 | 257 | 259 | 6 2 | 3-6 30 | 250 | CHURCH |
| 081008 | BS002 | 257 | 258 | 6 6 | -12 85 | 1600 FI | RE HOUSE |
| 081009 | CC008 | 257 | 272 | 6 33 | -11 8 | 28REG | ISTER CO |
| 081009 | CC010 | 257 | 260 | 6 1 | -12 11 | 64GAS | STATION |
| 081009 | CI098 | 257 | 276 | 6 21 | 22 100 | SCR | EW MACHS |
| 081009 | CI099 | 257 | 276 | 6 21 | -22 200 | SCR | EW MACHS |
| 081009 | CR001 | 257 | 266 | TAXMAP6 | 11-36 | | 613G |
| 081009 | CR002 | 257 | 266 | TAXMAP6 | 11-36 | | 612G |

⑤ RECORDS READ ..... 29
LIST FROM 9 FIELDS IDRCH1 IBLDG1 ADJ STOPO IADDR1 IADDR2 IADDR3 IADDR4 3

| IDRCH1 | IBLDG1 | ADJ | STOPO | IADDR1 | IADDR2 | IADDR3 | IADDR4 |
|---|---|---|---|---|---|---|---|
| 080909 | AC003 | 280 | 276 | 12 52 | -15 5 | 89BAR | ' GRILL |
| 080909 | AR013 | 280 | 271 | TAXMAP12 | -1 52-34 | | 13 |
| 080909 | AR014 | 280 | 271 | TAXMAP12 | -1 52-34 | | 10 |
| 081008 | BC010 | 257 | 258 | 6 6- | 14A 8 | 50 | DINER |
| 081008 | BC011 | 257 | 258 | 6 10 | -11 7 | 184 AP | PLIANCES |
| 081008 | BR029 | 257 | 262 | TAXMAP6 | 6-4 | | 120 |
| 081008 | BR031 | 257 | 262 | TAXMAP6 | 7-10 | | 11 |
| 081009 | CC001 | 257 | 258 | 6 38 | -11 14 | 24CHI | LD STORE |
| 081009 | CR004 | 257 | 272 | TAXMAP6 | 11-33 | | 40 |

```
RECORDS READ ......    9                                                    4
        MERGE FROM 8 9 TO 12 KEYS IBLDG1

RECORDS READ FROM FIRST INPUT FILE.....    29
RECORDS READ FROM SECOND INPUT FILE.....    9      ⑥
RECORDS WRITTEN TO OUTPUT FILE.........    38

        LIST FROM 12 FIELDS IDRCH1 IBLDG1 ADJ STOPO IADDR1 IADDR2 IADDR3 IADDR4   5
```

SIDEDT PROGRAM

| IDRCH1 | IBLDG1 | ADJ | STOPO | IADDR1 | IADDR2 | IADDR3 | IADDR4 |
|---|---|---|---|---|---|---|---|
| 080902 | EMR | 272 | 263 | | | | |
| 080909 | EMR | 280 | 273 | | | | |
| 080909 | TRN | 280.0 | 285 | | | | |
| 080909 | AC001 | 280 | 286 | 12-1 52 | -24 16 | 99GAS | STATION |
| 080909 | AC002 | 280 | 291 | 12 55 | -11 15 | 77 RE | STAURANT |
| 080909 | AC003 | 280 | 276 | 12 52 | -15 5 | 89BAR | ' GRILL |
| 080909 | AR001 | 280 | 274 | TAXMAP12 | -1 52-31 | | 15 |
| 080909 | AR002 | 280 | 274 | TAXMAP12 | -1 52-32 | | 15 |
| 080909 | AR005 | 280 | 275 | TAXMAP12 | -1 52-34 | | 19 |
| 080909 | AR013 | 280 | 271 | TAXMAP12 | -1 52-34 | | 13 |
| 080909 | AR014 | 280 | 271 | TAXMAP12 | -1 52-34 | | 10 |
| 080909 | AR022 | 280 | 275 | TAXMAP12 | 52-16 | | 9 |
| 080909 | AR023 | 280 | 275 | TAXMAP12 | -1 52-30 | | 20 |
| 081008 | BB001 | 258 | 258 | 4LA | HWY | | |
| 081008 | BB002 | 258 | 265.0 | 2LA | HWY | | |
| 081008 | BC001 | 257 | 258 | 6 6 | -22 4 | 25 PL | ANT SHOP |
| 081008 | BC002 | 257 | 257 | 6 1 | 4-6 10 | 36 | VACANT |
| 081008 | BC003 | 257 | 258 | 6 6 | -14 27 | 29 | HARDWARE |
| 081008 | BC010 | 257 | 258 | 6 6- | 14A 8 | 50 | DINER |
| 081008 | BC011 | 257 | 258 | 6 10 | -11 7 | 184 AP | PLIANCES |
| 081008 | BC016 | 257 | 258 | 6 10 | -13 17 | 279 CA | R DEALER |
| 081008 | BC017 | 257 | 259 | 6 20 | -11 6 | 39 | PRINTER |
| 081008 | BR001 | 257 | 262 | TAXMAP7 | 6-24 | | 32 |
| 081008 | BR002 | 257 | 260 | TAXMAP7 | 6-21 | | 14 |
| 081008 | BR003 | 257 | 259 | TAXMAP7 | 6-20 | | 42 |
| 081008 | BR029 | 257 | 262 | TAXMAP6 | 6-4 | | 120 |
| 081008 | BR031 | 257 | 262 | TAXMAP6 | 7-10 | | 11 |
| 081008 | BR034 | 257 | 260 | TAXMAP6 | 11-6 | | 10 |
| 081008 | BS001 | 257 | 259 | 6 2 | 3-6 30 | 250 | CHURCH |
| 081008 | BS002 | 257 | 258 | 6 6 | -12 85 | 1600 FI | RE HOUSE |
| 081009 | CC001 | 257 | 258 | 6 38 | -11 14 | 24CHI | LD STORE |
| 081009 | CC008 | 257 | 272 | 6 33 | -11 8 | 28REG | ISTER CO |
| 081009 | CC010 | 257 | 260 | 6 1 | -12 11 | 64GAS | STATION |
| 081009 | CI098 | 257 | 276 | 6 21 | 22 100 | SCR | EW MACHS |
| 081009 | CI099 | 257 | 276 | 6 21 | -22 200 | SCR | EW MACHS |
| 081009 | CR001 | 257 | 266 | TAXMAP6 | 11-36 | | 613G |
| 081009 | CR002 | 257 | 266 | TAXMAP6 | 11-36 | | 612G |
| 081009 | CR004 | 257 | 272 | TAXMAP6 | 11-33 | | 40 |

```
RECORDS READ ......    38
          END
```

## Appendix A

## TEST PROBLEM 2

## UPDATE AND MODIFY A SEQUENTIAL STRUCTURE FILE

### 1. Problem Purpose

This example will illustrate the use of the UPDATE and MODIFY commands. The UPDATE file is listed on page 20; it contains a structure identifier, a new structure damage function identifier for the structure, and a new reference flood elevation for the structure. Since the UPDATE command can only change one attribute at a time, the command must be used twice; the first time to change the damage function identifiers and the second time to change the reference flood elevation. In addition, all structures will be modified to increase their total value by fifteeen percent.

### 2. List of Input Cards for the Run

```
READ TYPE STRUCT
LIST FROM 8 FIELDS IBLDG1 IDIFS ADJ V1FS
UPDATE FROM 8 10 TO 11 /
     MATCH 1 8 WITH IBLDG1 /
     MOVE 14 16 INTO ID1FS
UPDATE FROM 11 10 TO 12 /
     MATCH 1 8 WITH IBLDG1/
     MOVE 17 24 INTO ADJ
MODIFY FROM 12 TO 11 BY /
     IF KODE1 EQ 'SL' THEN /
     MULTIPLY V1FS BY 1.15 GIVING V1FS
LIST FROM 11 FIELDS IBLDG1 ID1FS ADJ V1FS
END
```

### 3. Output Description

The sequential structure file for this example was created in Test Problem 2 by merging the two files listed on pages 16 and 17. Items 1 and 2 show the listing of the structure file before any modification. The UPDATE commands are listed as items 3 and 4. It tells the program to UPDATE file TAPE8 using file TAPE10 as the UPDATE FILE. The resulting modified file is the scratch file, TAPE11. The match key field is the structure identifier (IBLDG1) and is located in card columns 1 thru 8 in the UPDATE FILE. The field in the structure file to be updated is the damage function identifier (ID1FS); card columns 14 thru 16 of the UPDATE FILE will be moved into that field. The second UPDATE command uses the file just created (TAPE11) as the file to be modified. The UPDATE FILE (TAPE10) remains the same and so does the match key. The resultant file this time is TAPE12 and the field modified is ADJ. TAPE12 now contains the entire sequential structure file with the selected structures propertly updated.

The command to modify all the structures total value is item 5. The file to be modified is TAPE12; the output file is called TAPE11. Reusing files this way is perfectly all right. In order to modify all the structures, a

condition must be selected that applies to every structure. One possibility is the one shown in item 6, i.e., "IF KODE1 EQ 'SL'" since all structure records must have an SL card. Item 7 shows how to increase the structure's total value by 15 percent. The listing on page 36 illustrates the new structure file.

```
**********************************************************
*        SID EDIT PROGRAM                                *
*        USERS MANUAL -DRAFT IN PROGRESS                 *
*        UPDATED JULY 1983                               *
*                                                        *
*  RUN DATE 23 SEP 83  TIME  12:55:34                    *
**********************************************************
```

```
****************************************
*  U.S. ARMY CORPS OF ENGINEERS        *
*  THE HYDROLOGIC ENGINEERING CENTER   *
*  609 SECOND STREET, SUITE B          *
*  DAVIS, CALIFORNIA  95616            *
*  (916) 440-2105 (FTS) 448-2105       *
****************************************
```

```
 SSSSS   IIIIIII  DDDDDD   EEEEEEE  DDDDDD   TTTTTTT
S     S     I     D     D  E        D     D     T
S           I     D     D  E        D     D     T
 SSSSS      I     D     D  EEEEE    D     D     T
      S     I     D     D  E        D     D     T
S     S     I     D     D  E        D     D     T
 SSSSS   IIIIIII  DDDDDD   EEEEEEE  DDDDDD     T
```

```
READ TYPE STRUCT                                                    1
① LIST FROM 8 FIELDS IBLDG1 ID1FS ADJ V1FS                          2
```

②

| IBLDG1 | IDIFS | ADJ | VIFS |
|---|---|---|---|
| EMR | L77 | 275.5 | 0.00 |
| EMR | L77 | 275.5 | 0.00 |
| TRN | 777 | 280.0 | 0.00 |
| AC001 | X05 | 280 | 33.35 |
| AC002 | X04 | 280 | 59.80 |
| AC003 | XXX | 257 | 5.75 |
| AR001 | Z05 | 280 | 45.00 |
| AR002 | Z05 | 280 | 45.00 |
| AR005 | Z07 | 280 | 45.00 |
| AR013 | Z07 | 280 | 45.00 |
| AR014 | Z13 | 280 | 14.37 |
| AR022 | ZZZ | 300 | 14.37 |
| AR023 | Z07 | 280 | 45.00 |
| BB001 | B04 | 258 | 386.4 |
| BB002 | B04 | 258 | 248.4 |
| BC001 | X04 | 257 | 5.75 |
| BC002 | X06 | 257 | 10.35 |
| BC003 | X04 | 257 | 73.60 |
| BC010 | X01 | 257 | 17.25 |
| BC011 | X04 | 257 | 37.95 |
| BC016 | X03 | 257 | 70.15 |
| BC017 | X04 | 257 | 27.60 |
| BR001 | Z07 | 257 | 40.25 |
| BR002 | Z07 | 257 | 45.00 |
| BR003 | Z07 | 257 | 45.00 |
| BR029 | Z07 | 257 | 45.00 |
| BR031 | Z07 | 257 | 45.00 |
| BR034 | ZZZ | 259 | 34.50 |
| BS001 | X05 | 257 | 101.2 |
| BS002 | X05 | 257 | 358.5 |
| CC001 | X05 | 257 | 29.90 |
| CC008 | X04 | 257 | 17.25 |
| CC010 | X01 | 257 | 11.50 |
| CI098 | X00 | 272 | 65.55 |
| CI099 | X05 | 257 | 206.0 |
| CR001 | Z07 | 257 | 45.00 |
| CR002 | Z07 | 257 | 45.00 |
| CR004 | ZC7 | 257 | 39.10 |

```
RECORDS READ ......   38
        UPDATE FROM 8 10 TO 11 /                                    3
③           MATCH 1 8 WITH IBLDG1 /                                 4
            MOVE 14 16 INTO ID1FS                                   5

RECORDS READ FROM INPUT FILE.............      38
UPDATE RECORDS READ FROM UPDATE FILE.....       6
RECORDS WRITTEN TO OUTPUT FILE...........      38

        UPDATE FROM 11 10 TO 12 /                                   6
④           MATCH 1 8 WITH IBLDG1 /                                 7
            MOVE 17 24 INTO ADJ                                     8

RECORDS READ FROM INPUT FILE.............      38
UPDATE RECORDS READ FROM UPDATE FILE.....       6
RECORDS WRITTEN TO OUTPUT FILE...........      38

⑤
        MODIFY FROM 12 TO 11 BY /                                   9
⑥           IF KODE1 EQ 'SL' THEN /                                 10
⑦           MULTIPLY V1FS BY 1.15 GIVING V1FS                       11
ORECORD READ FOR MODIFY........      38
 RECORD WRITTEN AFTER MODIFY...      38
        LIST FROM 11 FIELDS IBLDG1 ID1FS ADJ V1FS                   12
```

| IBLDG1 | ID1FS | ADJ | V1FS |
|---|---|---|---|
| EMR | L77 | 275.5 | 0.00 |
| EMR | L77 | 275.5 | 0.00 |
| TRN | 7T7 | 280.0 | 0.00 |
| AC001 | X05 | 280 | 33.35 |
| AC002 | X04 | 280 | 59.80 |
| AC003 | XXX | 257 | 5.75 |
| AR001 | Z05 | 280 | 45.00 |
| AR002 | Z05 | 280 | 45.00 |
| AR005 | Z07 | 280 | 45.00 |
| AR013 | Z07 | 280 | 45.00 |
| AR014 | Z13 | 280 | 14.37 |
| AR022 | ZZZ | 300 | 14.37 |
| AR023 | Z07 | 280 | 45.00 |
| BB001 | B04 | 258 | 386.4 |
| BB002 | B04 | 258 | 248.4 |
| BC001 | X04 | 257 | 5.75 |
| BC002 | X06 | 257 | 10.35 |
| BC003 | X04 | 257 | 73.60 |
| BC010 | X01 | 257 | 17.25 |
| BC011 | X04 | 257 | 37.95 |
| BC016 | X03 | 257 | 70.15 |
| BC017 | X04 | 257 | 27.60 |
| BR001 | Z07 | 257 | 40.25 |
| BR002 | Z07 | 257 | 45.00 |
| BR003 | Z07 | 257 | 45.00 |
| BR029 | Z07 | 257 | 45.00 |
| BR031 | Z07 | 257 | 45.00 |
| BR034 | ZZZ | 259 | 34.50 |
| BS001 | X05 | 257 | 101.2 |
| BS002 | X05 | 257 | 358.5 |
| CC001 | X05 | 257 | 29.90 |
| CC008 | X04 | 257 | 17.25 |
| CC010 | X01 | 257 | 11.50 |
| CI098 | X00 | 272 | 65.55 |
| CI099 | X05 | 257 | 206.0 |
| CR001 | Z07 | 257 | 45.00 |
| CR002 | Z07 | 257 | 45.00 |
| CR004 | Z07 | 257 | 39.10 |

```
RECORDS READ ......   38
        END                                                    13
```

## Appendix A

## TEST PROBLEM 3

## WINDOW, NEWFIELD AND PULL A SEQUENTIAL STRUCTURE FILE

### 1. Problem Purpose

This example demonstrates how to window a structure file based on geographic coordinates. Remember, the coordinates must already be encoded and stored in the structure records. The NEWFIELD option is used to demonstrate how to access just a portion of a predefined attribute values. The PULL command uses the newly defined field to create a subset of the original sequential structure file.

### 2. List of Input Cards for the Run

```
READ TYPE STRUCT
LIST FROM 8 FIELDS IDRCH1 COLE ROWN
WINDOW FROM 8 TO 11 XCOOR COLE YCOOR ROWN /
     XMIN 493000 /
     XMAX 494000 /
     YMIN 4510000 /
     YMAX 4520000
NEWFIELD TRIB 3 4 CHAR
NEWFIELD DMGRCH 5 8 CHAR
PULL FROM 11 TO 12 BY /
     IF DMGRCH EQ '1009'
LIST FROM 12 FIELDS TRIB DMGRCH IBLDG1 IDCAT V1FS V1FC V1FO COLE ROWN
END
```

### 3. Output Description

Item 1 is the WINDOW command used to create a subset of a structure inventory file based on geographic coordinates. In this case, UTM coordinates were stored in the structure file as attributes COLE for the x-coordinate and ROWN for the y-coordinate. The SIDEDT program interprets the WINDOW command and informs you of the coordinates window (item 2) and how many records read from the input file fill within the coordinate window and written to scratch output file (item 3).

The NEWFIELD command (item 4) defines a 2 character attribute, 'TRIB', starting in position 3 and ending in position 4. The second NEWFIELD command defines a four character attribute, 'DMGRCH', in positions 5 through 8. These two new attributes may then be used as any originally defined attribute. In this example, the 6 character damage reach attribute (IDRCH1) is composed of a 2 character tributary code and a 4 character damage reach within tributary code. The new DMGRCH is used to select a subset of the windowed file, i.e., all structures in damage reach 1009 (item 5).

### 4. Program Output

The following pages are reproductions of the program execution.

```
*********************************************************
*        SID EDIT PROGRAM                               *
*        USERS MANUAL -DRAFT IN PROGRESS                *
*        UPDATED JULY 1983                              *
*                                                       *
*  RUN DATE 23 SEP 83  TIME  12:56:40                   *
*********************************************************
```

```
*****************************************
*  U.S. ARMY CORPS OF ENGINEERS         *
*  THE HYDROLOGIC ENGINEERING CENTER    *
*  609 SECOND STREET, SUITE B           *
*  DAVIS, CALIFORNIA  95616             *
*  (916) 440-2105 (FTS) 448-2105        *
*****************************************
```

```
 SSSSS   IIIIIII  DDDDDD   EEEEEEE  DDDDDD   TTTTTTT
S     S     I     D     D  E        D     D     T
S           I     D     D  E        D     D     T
 SSSSS      I     D     D  EEEEE    D     D     T
      S     I     D     D  E        D     D     T
S     S     I     D     D  E        D     D     T
 SSSSS   IIIIIII  DDDDDD   EEEEEEE  DDDDDD      T
```

```
READ TYPE STRUCT                                                    1
LIST FROM 8 FIELDS IDRCH1 COLE ROWN                                 2
```

| IDRCH1 | COLE | ROWN |
|---|---|---|
| 080902 | | |
| 080909 | | |
| 080909 | | |
| 080909 | 495725 | 4523100 |
| 080909 | 495845 | 4523100 |
| 080909 | 495550 | 4522750 |
| 080909 | 495628 | 4522974 |
| 080909 | 495619 | 4522952 |
| 080909 | 495592 | 4522886 |
| 080909 | 495627 | 4522933 |
| 080909 | 495650 | 4522915 |
| 080909 | 495520 | 4522710 |
| 080909 | 495637 | 4522994 |
| 081008 | | |
| 081008 | | |
| 081008 | 493389 | 4519817 |
| 081008 | 493380 | 4519718 |
| 081008 | 493375 | 4519675 |
| 081008 | 493415 | 4519675 |
| 081008 | 493477 | 4519650 |
| 081008 | 493575 | 4519650 |
| 081008 | 493550 | 4519625 |
| 081008 | 493400 | 4519875 |
| 081008 | 493397 | 4519789 |
| 081008 | 493385 | 4519760 |
| 081008 | 493538 | 4519712 |
| 081008 | 493562 | 4519100 |
| 081008 | 493433 | 4519650 |
| 081008 | 493393 | 4519846 |
| 081008 | 493455 | 4519675 |
| 081009 | 493350 | 4519475 |
| 081009 | 493418 | 4519506 |
| 081009 | 493350 | 4519425 |
| 081009 | 493550 | 4519525 |
| 081009 | 493550 | 4519525 |
| 081009 | 493381 | 4519543 |
| 081009 | 493387 | 4519537 |
| 081009 | 493425 | 4519500 |

```
RECORDS READ ......   38
      ┌ WINDOW FROM 8 TO 11 XCOOR COLE YCOOR ROWN /                    3
      │       XMIN 493000 /                                            4
①     │       XMAX 494000 /                                            5
      │       YMIN 4510000 /                                           6
      └       YMAX 4520000                                             7

  ┌ COLE     WILL BE USED FOR THE X COORDINATE FIELD
  │ ROWN     WILL BE USED FOR THE Y COORDINATE FIELD
②   THE COORDINATE WINDOW WILL BE:
  │     MINIMUM X =     493000.000 MAXIMUM X =     494000.000
  └     MINIMUM Y =    4510000.000 MAXIMUM Y =    4520000.000

  ┌ RECORDS READ FROM INPUT FILE.....        38
③ │ RECORD WRITTEN TO OUTPUT FILE....        23

      ④ NEWFIELD TRIB 3 4 CHAR                                         8
        NEWFIELD DMGRCH 5 8 CHAR                                       9
      ⑤ PULL FROM 11 TO 12 BY /                                        10
            IF DMGRCH EQ '1009'                                        11
ORECORDS READ FOR PULL......        23
 RECORD SELECTED AND OUTPUT.         8

        LIST FROM 12 FIELDS TRIB DMGRCH IBLDG1 IDCAT V1FS V1FC V1FO COLE ROWN   12
1                                                SIDEDT  PROGRAM
```

| TRIB | DMGRCH | IBLDG1 | IDCAT | V1FS | V1FC | V1FO | COLE | ROWN |
|---|---|---|---|---|---|---|---|---|
| 08 | 1009 | CC001 | COM | 26 | 80 | | 493350 | 4519475 |
| 08 | 1009 | CC008 | COM | 15 | 19 | | 493418 | 4519506 |
| 08 | 1009 | CC010 | COM | 10 | 14 | | 493350 | 4519425 |
| 08 | 1009 | CI098 | IND | 57 | 199 | | 493550 | 4519525 |
| 08 | 1009 | CI099 | IND | 180 | 900 | | 493550 | 4519525 |
| 08 | 1009 | CR001 | RES | 40 | 22.1 | | 493381 | 4519543 |
| 08 | 1009 | CR002 | RES | 40 | 22.1 | | 493387 | 4519537 |
| 08 | 1009 | CR004 | RES | 34 | 19.2 | | 493425 | 4519500 |

```
RECORDS READ ..... 8
        END
```

## Appendix A

## TEST PROBLEM 4

## MERGE AND MODIFY SEQUENTIAL DAMAGE FUNCTION FILES

### 1. Problem Purpose

Often after a sequential damage function file is created, new damage functions will need to be added to the file. The MERGE command is used to carry out this function. This example also shows how to modify the merged file using the MODIFY command.

### 2. List of Input Cards for the Run

```
READ TYPE DAMAGE
LIST FROM 8 FIELDS IT NSTAG PERCNT17
LIST FROM 9 FIELDS IT NSTAG PERCNT17
MERGE FROM 8 9 TO 11 KEYS IT
MODIFY FROM 11 TO 12 BY /
     IF IT EQ ' R1' /
     THEN MOVE 44. TO PERCNT17
LIST FROM 12 FIELDS IT NSTAG PERCNT17
END
```

### 3. Output Description

Because this test problem accesses a damage function file, the first command to the SIDEDT program must be the 'READ TYPE DAMAGE' command (item 1). Items 2 and 3 are listings of the original damage function file and the damage functions to be added, respectively. The MERGE command, (item 4) is used to add the damage functions of TAPE9 to the damage functions of TAPE8. The resultant file is TAPE11. The key field is the attribute IT, the damage function identifier. Because the 2 functions on TAPE9 have the same key as 2 functions on TAPE8, the MERGE command replaces the old functions with the two new ones. In effect, the command is acting like a replace function.

The merged file (TAPE11) is then used as input to the MODIFY command (item 5). The damage function ' R1' is modified to correct the value of the seventeenth percent damage value from 440000 to 44 percent.

### 4. Program Output

The following pages are a reproduction of the program execution.

```
*********************************************************
*        SID EDIT PROGRAM                               *
*        USERS MANUAL -DRAFT IN PROGRESS                *
*        UPDATED JULY 1983                              *
*                                                       *
*  RUN DATE 23 SEP 83  TIME  12:58:01                   *
*********************************************************
```

```
*****************************************
*  U.S. ARMY CORPS OF ENGINEERS         *
*  THE HYDROLOGIC ENGINEERING CENTER    *
*  609 SECOND STREET, SUITE B           *
*  DAVIS, CALIFORNIA  95616             *
*  (916) 440-2105 (FTS) 448-2105        *
*****************************************
```

```
 SSSSS   IIIIIII  DDDDDD   EEEEEEE  DDDDDD   TTTTTTT
S     S     I     D     D  E        D     D     T
S           I     D     D  E        D     D     T
 SSSSS      I     D     D  EEEEE    D     D     T
      S     I     D     D  E        D     D     T
S     S     I     D     D  E        D     D     T
 SSSSS   IIIIIII  DDDDDD   EEEEEEE  DDDDDD      T
```

```
(1) READ TYPE DAMAGE                                    1
    LIST FROM 8 FIELDS IT NSTAG PERCNT17                2
```

②

SIDEDT PROGRAM

| IT | NSTAG | PERCNT17 |
|---|---|---|
| C1 | 10 | |
| C2 | 10 | |
| C3 | 7 | |
| C4 | 12 | |
| C5 | 17 | 9.6 |
| C6 | 8 | |
| C7 | 12 | |
| C8 | 15 | 3.0 |
| R1 | 20 | 440000 |
| R2 | 20 | 55 |

RECORDS READ ..... 10

LIST FROM 9 FIELDS IT NSTAG PERCNT17

③

| IT | NSTAG | PERCNT17 |
|---|---|---|
| C1 | 10 | |
| R2 | 20 | 60 |

```
RECORDS READ ......    2
   (4)  MERGE FROM 8 9 TO 11 KEYS IT                                    4


RECORDS READ FROM FIRST INPUT FILE.....        10
RECORDS READ FROM SECOND INPUT FILE.....        2
RECORDS WRITTEN TO OUTPUT FILE.........        10


   (5)  MODIFY FROM 11 TO 12 BY /                                       5
            IF IT EQ ' R1' /                                            6
            THEN MOVE 44. TO PERCNT17                                   7
ORECORD READ FOR MODIFY.....        10
 RECORD WRITTEN AFTER MODIFY...     10
        LIST FROM 12 FIELDS I1 NSTAG PERCNT17                           8
```

| IT | NSTAG | PERCNT17 |
|---|---|---|
| C1 | 10 | |
| C2 | 10 | |
| C3 | 7 | |
| C4 | 12 | |
| C5 | 17 | 9.6 |
| C6 | 8 | |
| C7 | 12 | |
| C8 | 15 | 3.0 |
| R1 | 20 | 44 |
| R2 | 20 | 60 |

```
RECORDS READ ......   10
        END
```

## Appendix A

## TEST PROBLEM 5

## CREATE A RANDOM DAMAGE FUNCTION FILE

### 1. Problem Purpose

The purpose of this example is to create a random damage function file. When running the SID program using a random damage function file a sequential file of just DF cards is also required. This example will also create this necessary file (TAPE92).

### 2. Listing of Input Cards for the Run

```
READ TYPE DAMAGE
PULL FROM 8 TO 98 BY IF NSTAG GE 0
PULL FROM 8 TO 92 BY IF NSTAG GE 0
LIST FROM 98 FIELDS IT NSTAG SAGE1 PERCNT1
END
```

### 3. Output Description

The first PULL command (item 1) creates the random damage function file (TAPE98) from the sequential file (TAPE8). All damage functions with an NSTAG greater than or equal to zero will be written to the new file. (Note - this is one way to capture all the damage functions). The second PULL command (item 2) creates the required TAPE92. This second command must always be present.

### 4. Program Output

The following pages are a reproduction of the program execution.

```
**********************************************************
*        SID EDIT PROGRAM                                *
*        USERS MANUAL -DRAFT IN PROGRESS                 *
*        UPDATED JULY 1983                               *
*                                                        *
*  RUN DATE 23 SEP 83  TIME  12:58:47                    *
**********************************************************
```

```
****************************************
*  U.S. ARMY CORPS OF ENGINEERS        *
*  THE HYDROLOGIC ENGINEERING CENTER   *
*  609 SECOND STREET, SUITE B          *
*  DAVIS, CALIFORNIA  95616            *
*  (916) 440-2105 (FTS) 448-2105       *
****************************************
```

```
 SSSSS   IIIIIII  DDDDDD   EEEEEEE  DDDDDD   TTTTTTT
S     S     I     D     D  E        D     D     T
S           I     D     D  E        D     D     T
 SSSSS      I     D     D  EEEEE    D     D     T
      S     I     D     D  E        D     D     T
S     S     I     D     D  E        D     D     T
 SSSSS   IIIIIII  DDDDDD   EEEEEEE  DDDDDD      T
```

```
      READ TYPE DAMAGE                                                  1
  (1) PULL FROM 8 TO 98 BY IF NSTAG GE 0                                2
ORECORDS READ FOR PULL.....        10
 RECORD SELECTED AND OUTPUT.       10

  (2) PULL FROM 8 TO 92 BY IF NSTAG GE 0                                3
ORECORDS READ FOR PULL.....        10
 RECORD SELECTED AND OUTPUT.       10

      LIST FROM 98 FIELDS IT NSTAG SAGE1 PERCNT1                        4
```

| IT | NSTAG | SAGE1 | PERCNT1 |
|---|---|---|---|
| C1 | 10 | -2 | 0 |
| C2 | 10 | -3 | 0 |
| C3 | 7 | 0 | 0 |
| C4 | 12 | -1 | 0 |
| C5 | 17 | -2 | 0 |
| C6 | 8 | 0 | 0 |
| C7 | 12 | 0 | 0 |
| C8 | 15 | -2 | 0 |
| R1 | 20 | -8 | |
| R2 | 20 | -9 | 0 |

```
RECORDS READ ......   10
        END                                                    5
```

## Appendix A

## TEST PROBLEM 6

## CREATE A RANDOM STRUCTURE INVENTORY FILE

1. Problem Purpose

This example demonstrates how to create a random structure file which is a subset of an input sequential structure file.

2. List of Input Cards for the Run

```
READ TYPE STRUCT
PULL FROM 8 TO 99 BY /
     IF IDRCH1 EQ '081008'
LIST FROM 99 FIELDS IDRCH1 IBLDG1 STOPO DELTZ DELTB DELTG
END
```

3. Output Description

Item 1 shows how to use the PULL command to do two functions at the same time. The first function is to create the random file (TAPE99) from the sequential file (TAPE8). The second function is to only include a subset of the input file, i.e., only structures located in damage reach '081008', on the random file.

4. Program Output

The following pages are a reproduction of the program execution.

```
*******************************************************
*       SID EDIT PROGRAM                              *
*       USERS MANUAL -DRAFT IN PROGRESS               *
*       UPDATED JULY 1983                             *
*                                                     *
*  RUN DATE 23 SEP 83  TIME  13:02:32                 *
*******************************************************
```

```
****************************************
* U.S. ARMY CORPS OF ENGINEERS         *
* THE HYDROLOGIC ENGINEERING CENTER    *
* 609 SECOND STREET, SUITE B           *
* DAVIS, CALIFORNIA  95616             *
* (916) 440-2105 (FTS) 448-2105        *
****************************************
```

```
 SSSSS   IIIIIII  DDDDDD   EEEEEEE  DDDDDD   TTTTTTT
S     S     I     D     D  E        D     D     T
S           I     D     D  E        D     D     T
 SSSSS      I     D     D  EEEEE    D     D     T
      S     I     D     D  E        D     D     T
S     S     I     D     D  E        D     D     T
 SSSSS   IIIIIII  DDDDDD   EEEEEEE  DDDDDD     T
```

```
        READ TYPE STRUCT                                                    1
        PULL FROM 8 TO 99 BY /                                              2
   (1)      IF IDRCH1 EQ '081008'                                           3
0RECORDS READ FOR PULL.....        38
 RECORD SELECTED AND OUTPUT.       17

        LIST FROM 99 FIELDS IDRCH1 IBLDG1 STOPO DELTZ DELTB DELTG           4
1                                                    SIDEDT  PROGRAM
```

| IDRCH1 | IBLDG1 | STOPO | DELTZ | DELTB | DELTG |
|---|---|---|---|---|---|
| 081008 | BB001 | 258 | -4 | | |
| 081008 | BB002 | 265.0 | -4 | | |
| 081008 | BC001 | 258 | -6 | | |
| 081008 | BC002 | 257 | -.01 | | |
| 081008 | BC003 | 258 | -3 | | |
| 081008 | BC010 | 258 | -.01 | | |
| 081008 | BC011 | 258 | -1 | | |
| 081008 | BC016 | 258 | -.01 | | |
| 081008 | BC017 | 259 | -1 | | |
| 081008 | BR001 | 262 | -2 | | |
| 081008 | BR002 | 260 | -2 | | |
| 081008 | BR003 | 259 | -2 | | |
| 081008 | BR029 | 262 | -2 | | |
| 081008 | BR031 | 262 | -2 | | |
| 081008 | BR034 | 260 | -.01 | | |
| 081008 | BS001 | 259 | -.01 | | |
| 081008 | BS002 | 258 | -2 | | |

# APPENDIX B

# SID STRUCTURE CARDS

# APPENDIX B
# STRUCTURE CARDS
# (for SID program)

## 1 STRUCTURE CARDS

The structure cards which follow (SL and SD required, SO, SS and SA optional) provide the basic inventory data for the structures to be subjected to damage potential analysis. The SS and SA cards do not presently result in analysis. They have been defined so that future applications (that might be developed) could be accommodated in initial field data collection efforts.

### 1.1 SL CARD

This required card provides identification codes, locational information, structure elevations, and printout controls. The numbers in parentheses under FIELD are the card column numbers for input.

| FIELD | VARIABLE | RECORD FIELD | VALUE | ATTRIBUTE TYPE | DESCRIPTION |
|---|---|---|---|---|---|
| 0 | KODE1 | (1-2) | SL | (C) | Card identification. |
| 1 | IDRCH1 | (3-8) | AN | (C) | Damage reach identification code that will be used for structure damage potential aggregation for damage potential function construction, summary printout, and file transfer. The structure is presumed to be located within this specified damage reach. |
| 2 | IBLDG1 | (9-16) | AN | (C) | Structure identification code. Used for all subsequent accounting, and storage and retrieval of data for this structure. |
| 3 | ROWN | (17-24) | + | (R) | If optional coordinate values are used (see page 17 main text for discussion), this value is the row or north coordinate point. Any rectilinear coordinate system may be used such as row/column or the Universal Transverse Mecator (UTM) system. |

APPENDIX B
STRUCTURE CARDS
(for SID program)

1.1 SL CARD (continued)

| FIELD | VARIABLE | RECORD FIELD | VALUE | ATTRIBUTE TYPE | DESCRIPTION |
|---|---|---|---|---|---|
| 4 | COLE | (25-32) | + | (R) | If coordinates are used, (optional) this value is the column or east coordinate point. |
| 5 | ADJ | (33-40) | + | (R) | Elevation of the reference flood at the structure (in feet). Used (in conjunction with damage reach reference flood elevation) to adjust structure elevation-damage potential at site to the index location. See discussion page 8 in main text. |
| 6 | STOPO | (41-48) | + | (R) | Elevation of reference point selected for structure (in feet). Must be input as either 1) first floor elevation or 2) ground elevation. If elevation is input as ground elevation, will be adjusted to first floor by addition of DELTG (SL.9) below. The first floor elevation corresponds to the zero stage value on stage damage function (DF, DP, PC (or DD) cards). If left blank, or assigned as zero stage, values on DP card are assumed to be elevation values. See text page 17. |
| 7 | DELTZ | (49-56) | +,- | (R) | Difference between water surface elevation that can cause damage to begin at first floor For example, if a basement opening exists that would admit water at some elevation above the basement floor, damage might not begin until water reaches that elevation. If the point is below first floor elevation, elevation difference input should be negative (e.g., preceded by a minus sign). See text page 19. |

| | | | | | |
|---|---|---|---|---|---|
| 8 | DELTB | (57-64) | +,- | (R) | Difference between elevation of basement floor and first floor elevation. Elevation difference input would normally be negative (e.g., preceeded by a minus sign). Needed if structure has a basement and separate damage function is to be used for basement only. See text page 19. |

| FIELD | VARIABLE | RECORD FIELD | VALUE | ATTRIBUTE TYPE | DESCRIPTION |
|---|---|---|---|---|---|
| 9 | DELTG | (65-72) | +,- | (R) | Used only if elevation STOPO (SL.6) was input as ground elevation. Needed to adjust STOPO to first floor elevation. Difference between elevation of first floor and ground elevation. If first floor elevation is above ground, elevation difference is positive and should be so input. |
| 10 (73-74) | IFUNC | (73-74) | 0 | (I) | Analysis to be performed will use a single level damage function and only SD card will be included. SO, SS or SA cards are not used. |
| | | | 1 | | One additional structure card (either a SO, SS, SA card) will be included with the required SL and SD cards. |
| | | | 2 | | Two additional structure cards (either SO and SS, SO and SA, or SS and SA cards) will be included with the required SL and SD cards. |
| | | | 3 | | Three additional structure cards (SO, SS and SA cards) will be included with the required SL and SD cards. |
| 10(75) | NEWSTR | (75-75) | 0 | (I) | Structure will be considered as "existing" for analysis purposes. |
| | | | 1 | | Structure will be considered as "new" (e.g., does not presently exist but will be built at some future date) for analysis purposes. |

## APPENDIX B
## STRUCTURE CARDS
## (for SID program)

### 1.2 SD CARD

The required SD card specifies the damage category (for damage potential consolidation), damage function assignments, and values for structures and contents. The numbers in parentheses under FIELD are the column numbers for input.

| FIELD | VARIABLE | RECORD FIELD | VALUE | ATTRIBUTE TYPE | DESCRIPTION |
|---|---|---|---|---|---|
| 0 | KODE2 | (81-82) | SD | (C) | Card identification. |
| 1 | IDRCH2 | (83-88) | AN | (C) | Damage reach identification code (identical to SL.1). |
| 2 | IBLDG2 | (89-96) | AN | (C) | Building identification code (identical to SL.2). |
| 3 | IDCAT | (97-104) | AN | (C) | Damage category (specified on DC cards) JDCT (DC.2) to which this structure will be assigned for consolidation of damage potential of all structures. |
| 4 (25-27) | ID1FS | (105-107) | AN | (C) | Identification code for damage potential function to be assigned to this structure. Use appropriate DF card identification code, IT (DF.1). |
| 4 (28-32) | V1FS | (108-112) | + | (R) | Total value of structure in thousands of dollars ($1000). If damage function to be assigned to this value is a percent function, this value provides the conversion. Otherwise the value input here is used in various tables and summaries. |
| 5 (33-35) | ID1FC | (113-115) | AN | (C) | Identification code for damage potential function to be assigned to damage to contents for this structure. Use appropriate DF card identifier code, IT (DF.1). |

1.2 SD CARD (continued)

| FIELD | VARIABLE | RECORD FIELD | VALUE | ATTRIBUTE TYPE | DESCRIPTION |
|---|---|---|---|---|---|
| 5 (36-40) | VIFC | (116-120) | + | (R) | Total value contents in thousands of dollars ($1000). |
| | | | - | | Value of contents is a percentage of the structure value. Input as negative whole number (e.g., 50% is input as -50). |
| 6 (41-43) | ID1FO | (121-123) | AN | (C) | Identification code for damage potential function to be used for damage to "other" items. Use appropriate DF card identification code, IT (DF.1). |
| 6 (44-48) | V1FO | (124-128) | + | (R) | Total value of "other" items thousands of dollars ($1000). |
| | | | - | | Value of "other" is a percentage of the structure value. Input as a negative whole number (e.g., 5% is input as -5). |
| 7 | IADDR1 | (129-136) | AN | (C) | Space allowed for comment/record keeping. Could be used to record address, source of structure market values, land costs, or other miscellaneous information. |
| 8 | IADDR2 | (137-144) | AN | (C) | Same as above. |
| 9 | IADDR3 | (145-152) | AN | (C) | Same as above. |
| 10 | IADDR4 | (153-160) | AN | (C) | Same as above. |

## APPENDIX B
## STRUCTURE CARDS
## (for SID program)

### 1.3 SO CARD

The optional SO card (see IFUNC (SL.10)) provides for additional specification of analysis for the basement and above first floor categories for those users who desire to evaluate structures at three levels. In this case, the SD card is then used to provide only first floor information. The numbers in parentheses under FIELD are the card column numbers for input.

| FIELD | VARIABLE | RECORD FIELD | VALUE | ATTRIBUTE TYPE | DESCRIPTION |
|---|---|---|---|---|---|
| 0 | KODE3 | (161-162) | SO | (C) | Card identification. |
| 1 | IDRCH3 | (163-168) | AN | (C) | Damage reach identification code dentical to SL.1.). |
| 2 | IBLDG3 | (169-176) | AN | (C) | Building identification code (identical to SL.2.). |
| 3 (17-19) | IDBS | (177-179) | AN | (C) | Identification code for damage potential function to be assigned for damage to the structure of the basement. Use appropriate DF identifier code, IT (DF.1). |
| 3 (20-24) | VBS | (180-184) | + | (R) | Total value of the structure basement in thousands of dollars ($1000). |
| 4 (25-27) | IDBC | (185-187) | AN | (C) | Identification code for damage potential function to be assigned to damage to contents of the basement. Use appropriate DF identifier, IT (DF.1). |
| 4 (28-32) | VBC | (188-192) | + | (R) | Total value of the contents of the basement in thousands of dollars ($1000). |
| | | | - | | Value of the contents of the basement is a percentage of the structure value of the basement. Input as negative whole number (e.g., 50% is input as -50). |

1.3 SO CARD (continued)

| FIELD | VARIABLE | RECORD FIELD | VALUE | ATTRIBUTE TYPE | DESCRIPTION |
|---|---|---|---|---|---|
| 5 (33-35) | IDBO | (193-195) | AN | (C) | Identification code for damage potential function to be assigned for damage to "other" items of the basement. Use appropriate DF card identifier, IT (DF.1). |
| 5 (36-40) | VBO | (196-200) | + | (R) | Total value of the "other" items of the basement in thousands of dollars ($1000). |
| | | | - | | Value of the "other" items of the basement is a percentage of the structure value of the basement. Input as negative whole number (e.g., 50% is input as -50). |
| 6 (41-43) | IDAS | (201-203) | AN | (C) | Identification code for damage potential function to be assigned for damage to the structural portion above the first floor. Use appropriate DF card identifier, IT (DF.1). |
| 6 (44-48) | VAS | (204-208) | + | (R) | Total value of the structural portion above the first floor in thousands of dollars ($1000). |
| 7 (49-51) | IDAC | (209-211) | AN | (C) | Identification code for damage potential function to be assigned for damage to the contents above the first floor. Use appropriate DF card identifier, IT (DF.1). |
| 7 (52-56) | VAC | (212-216) | + | (R) | Total value of the contents above floor in thousands of dollars ($1000). |
| | | | - | | Value of the contents for above first floor is a percentage of the structural portion above first floor value. Input as negative whole number (e.g., 50% is input as -50). |

1.3 SO CARD (continued)

| FIELD | VARIABLE | RECORD FIELD | VALUE | ATTRIBUTE TYPE | DESCRIPTION |
|---|---|---|---|---|---|
| 8 (57-59) | IDAO | (217-219) | AN | (C) | Identification code for damage potential function to be assigned for damage to "other" items above first floor. Use appropriate DF card identifier, IT (DF.1). |
| 8 (60-64) | VAO | (220-224) | + | (R) | Total value of the "other" items the first floor in thousands of dollars ($1000). |
| | | | - | | Value of the "other" for above first floor is a percentage of the structural portion above first floor value. Input as negative whole number (e.g., 50% is input as -50). |
| 9-10 | | | | | Blank fields. |

# APPENDIX B
# STRUCTURE CHARACTERISTICS/DOCUMENTATION CARDS
# (for SID program)

## 2 STRUCTURE CHARACTERISTICS/DOCUMENTATION CARDS

The SS and SA cards have been formulated to provide a systematic data capture procedure for cataloging more precisely the characteristics of inventoried structures. The data contained on these cards are simply read and printed. Future plans for enhancement of SID capabilities include sort, display, and summary operations on these data items, and later, creation of analysis routines to permit refined nonstructural and other analysis.

### 2.1 SS CARD

The optional SS (see IFUNC (SL.10)) card provides for cataloging more detailed information on the structure to allow for potential, (not yet developed) more detailed, economic and nonstructural analysis. The numbers in parentheses under FIELD are the column numbers for input.

| FIELD | VARIABLE | RECORD FIELD | VALUE | ATTRIBUTE TYPE | DESCRIPTION |
|---|---|---|---|---|---|
| 0 | KODE4 | (241-242) | SS | (C) | Card identification. |
| 1 | IDRCH4 | (243-248) | AN | (C) | Identification code of damage reach to which this structure is assigned (identical to SL.1). |
| 2 | IBLDG4 | (249-256) | AN | (C) | Building identification code (identical to SL.2). |
| 3 (17-20) | YC | (257-260) | + | (R) | Year of completion of structure construction (e.g., 1952). Used as indicator of age of structure. |
| 3 (21-22) | SF | (261-262) | AN | (C) | Soil foundation types used to determine seepage/construction problems/potential. Up to 5 types (defined by user) may be specified. An example might be:<br>1 Gravel<br>2 Rock<br>3 Impervious<br>4 Swampy<br>5 Other |

## APPENDIX B
## STRUCTURE CHARACTERISTICS/DOCUMENTATION CARDS
## (for SID program)

2.1 SS CARD (continued)

| FIELD | VARIABLE | RECORD FIELD | VALUE | ATTRIBUTE TYPE | DESCRIPTION |
|---|---|---|---|---|---|
| 3 (23-24) | TG | (263-264) | AN | (C) | Categorization of structure types used as indicators of nature of construction and for statistical analysis. Up to 20 types (defined by user) may be specified. An example might be:<br>1 Colonial<br>2 Ranch<br>3 Row<br>4 Trailer, etc.<br>Etc. |
| 4 | CG | (265-266) | AN | (C) | Categorization of (25-26) construction type. Used as indicator for potential modifications. Up to 10 categories (defined by user) may be specified. An example might be:<br>1 Wood frame<br>2 Prefab<br>3 Masonry<br>4 Steel frame<br>Etc. |
| 4 (27-28) | NG | (267-268) | + | (I) | Code for number of floors (not including basement):<br>1 One floor<br>2 Two floors<br>3 More than two floors |
| 4 (29-30) | BG | (269-270) | | (R) | Code for presence of basement. |
| | | | 0 | | No basement.. |
| | | 1 | | | Structure has a basement. |
| 5 (33-34) | BT | (273-274) | AN | (C) | Categorization of basement type.<br>Up to 5 categories (user defined) may be specified. An example might be:<br>1 Full<br>2 Partial<br>3 None, slab foundation<br>Etc. |

2.1 SS CARD (continued)

| FIELD | VARIABLE | RECORD FIELD | VALUE | ATTRIBUTE TYPE | DESCRIPTION |
|---|---|---|---|---|---|
| 5 | BC | (275-276) | AN | (C) | Code for basement (35-36) construction type. Up to 10 types (user defined) may be specified. An example might be:<br>1 Wood frame<br>2 Prefab<br>3 Masonry<br>4 Steel frame, etc.<br>Etc. |
| 5 (37-40) | BSIZE | (277-280) | + | (R) | Basement area in hundred square feet |
| 6 (41-42) | NWB | (281-282) | + | (I) | Number of windows below the first floor. |
| 6 (43-45) | WAB | (283-285) | + | (R) | Average size of the window openings below the first floor (square feet). |
| 6 (46-48) | WBF | (286-288) | + | (R) | Elevation difference between the lowest window below the first floor and the first floor reference point. |
| 7 (49-50) | NOB | (289-290) | + | (I) | Number of "other" openings below the first floor. |
| 7 (51-53) | OAD | (291-293) | + | (R) | Average size of the "other" openings below the first flocr (square feet). |
| 7 (54-56) | OBF | (294-296) | + | (R) | Elevation difference between the lowest "other" openings below the first floor and the first floor reference point. |
| 8 (57-58) | | | | | Blank. |

2.1 SS CARD (continued)

| FIELD | VARIABLE | RECORD FIELD | VALUE | ATTRIBUTE TYPE | DESCRIPTION |
|---|---|---|---|---|---|
| 8 (59-60) | FC | (299-300) | AN | (C) | Code for first floor construction types. Up to 10 types (user defined) may be specified. An example might be:<br>1 Wood frame<br>2 Prefab<br>3 Masonry<br>4 Steel frame, etc.<br>Etc. |
| 8 (61-64) | FSIZE | (301-304) | + | (R) | First floor area in hundred square feet. |
| 9 (65-66) | NWF | (305-306) | + | (I) | Number of windows in the first floor. |
| 9 (67-69) | WAF | (307-309) | + | (R) | Average size of window openings on first floor (square feet). |
| 9 (70-72) | WDF | (310-312) | + | (R) | Elevation difference between the lowest window above the first floor and the first floor reference point elevation. |
| 10 (73-74) | NOF | (313-314) | + | (I) | Number of "other" openings above the first floor elevation. |
| 10 (75-77) | OAF | (315-317) | + | (R) | Average size of "other" openings above the first floor elevation (square feet). |
| 10 (78-80) | ODF | (318-320) | + | (R) | Elevation difference between the lowest "other" openings above the first floor and the first floor reference elevation. |

APPENDIX B
STRUCTURE CHARACTERISTICS/DOCUMENTATION CARDS
(for SID program)

## 2.2 SA CARD

The optional SA card (see IFUNC (SL.10)) provides for additional cataloging and naming of the structure (e.g., resident or business) and record keeping such as street address.

| FIELD | VARIABLE | RECORD FIELD | VALUE | ATTRIBUTE TYPE | DESCRIPTION |
|---|---|---|---|---|---|
| 0 | KODE5 | (321-322) | SA | (C) | Card identification. |
| 1 | IDRCH5 | (323-328) | AN | (C) | Identification code for the damage reach to which structure is assigned (identical to SL.1). |
| 2 | IBLDG5 | (329-336) | AN | (C) | Building/structure identification code (identical to SL.1). |
| 3 | RESID1 | (337-344) | AN | (C) | Name of resident or business. |
| 4 | RESID2 | (345-352) | AN | (C) | Same as above. |
| 5 | ADDR1 | (353-360) | AN | (C) | Street address. |
| 6 | ADDR2 | (361-368) | AN | (C) | Same as above. |
| 7 | ADDR3 | (369-376) | AN | (C) | Same as above. |
| 8 | CITY1 | (377-384) | AN | (C) | City or town. |
| 9 | CITY2 | (385-392) | AN | (C) | Same as above. |
| 10 | IZIP | (393-400) | + | (I) | Zip code. |

# APPENDIX C

## SID DAMAGE FUNCTION CARDS

# APPENDIX C
# DAMAGE FUNCTION CARDS
# (for SID program)

## 1 DAMAGE FUNCTION CARDS

These cards are required if NDFILE (J2.9) is not equal to 92. Three card types DF, DP and PC (or DD) are required for each damage function. There must be NODF (J2.1) sets of DF, DP, and PC (or DD) cards.

### 1.1 DF CARD

The DF card identifies the damage function, specifies the number of depth tabulation values and flags the nature of the damage values and file source. If NDFILE (J2.9, is 92, the DF card image is required on tape or disk. If NDFILE (J2.9) is 92 or 98, damage function data is resident on random access file 98. DF cards must be included in the job stream to retrieve from the random access file those damage functions to be used in the specific computer run. If NDFILE (J2.9) is 92, provide DF in card image format on tape or disk, specifiying the appropriate identification codes (DF.1) and IDFILE (DF.4) as 1. If NDFILE (J2.9) is 98, provided DF cards as physical input specifying (DF.1), as before and DF.4 as 1.

| FIELD | VARIABLE | RECORD FIELD | VALUE | ATTRIBUTE TYPE | DESCRIPTION |
|---|---|---|---|---|---|
| 0 | KODE1 | (1-2) | DF | (C) | Card identification. |
| 1 (6-8) | IT | (6-8) | AN | (C) | Damage function identification code. (maximum of 30 characters). |
| 2 | NSTAG | (9-16) | + | (I) | Number of stage tabulation values - maximum of 20. |
| 3 | IDF | (17-24) | 0 | (I) | Damage values placed on PC cards are PERCENT damage values. |
| | | | 1 | | Damage values placed on DD cards are direct (actual) DOLLAR values. |
| 4 | IDFILE | (25-32) | 0 | (I) | Stage and damage data are physically on cards or DF, DP, PC (or DD)) card images exist on a computer disk file. NDFILE (J2.9) = 0 or 2. |
| | | | 1 | | Stage and damage data (DF, DP, PC (or DD)) are resident on a RANDOM IO File. NDFILE (J2.9) = 98. |

# APPENDIX C
# DAMAGE FUNCTION CARDS
# (for SID program)

## 1.2 DP CARD

The DP card specifies the stage values (of stage damage functions). The first 10 stage values are placed on the initial DP card and the remainder, if needed, are placed on a second DP card (maximum of 20). The initial stage tabulation value should correspond to the zero damage point. Values are input in ascending order but do not have to be of a uniform interval between values. Elevation values instead of stage values are assumed if STOPO (SL.6) is equal to zero.

| FIELD | VARIABLE | RECORD FIELD | VALUE | ATTRIBUTE TYPE | DESCRIPTION |
|---|---|---|---|---|---|
| 0 | KODE2 | (81-82) | DP | (C) | Card identification. |
| 1 | SAGE1 | (83-88) | + | (R) | First stage value. May be negative (use "-" sign) and should correspond to zero damage. |
| 2 | SAGE2 | (89-96) | + | (R) | Second stage value. May be negative (use "-" sign). |
| N | SAGEN | (233-240) | + | (R) | Same as above for NSTAG (DF.2) stage values. Continue for as many DP cards as needed. |

APPENDIX C
DAMAGE FUNCTION CARDS
(for SID program)

## 1.3 PC CARD

The PC cards specify the percent damage values corresponding to the stage values specified on the DP card(s). The first 10 values are placed on the initial PC card and the remainder, if needed, are placed on a second PC card. The first depth (DP.1) and associated percent damage (PC.1) values should correspond to zero damage point. Required if IDF (DF.3) is 0 or blank.

| FIELD | VARIABLE | RECORD FIELD | VALUE | ATTRIBUTE TYPE | DESCRIPTION |
|---|---|---|---|---|---|
| 0 | KODE4 | (241-242) | PC | (C) | Card identification. |
| 1 | PERCNT1 | (243-248) | + | (R) | Percent damage in whole numbers (e.g., 60% is input as 60) corresponding to first depth value. Initial value should be zero. |
| 2 | PERCNT2 | (249-256) | + | (R) | Percent damage (in whole numbers) corresponding to second depth value. |
| N | PERCNTN | (393-400) | + | (R) | Percent damage corresponding to N depth value. Continue for as many PC cards as needed. |

## APPENDIX C
## DAMAGE FUNCTION CARDS
## (for SID program)

### 1.4 DD CARD

The DD card is an optional alternative to the PC card to specify direct damage values corresponding to the stage values on the DP card(s). The first 10 direct damage values are placed on the first DD card and the remainder, if needed, are placed on a second DD card. It is good practice to have the first depth (DP.1) and direct damage (DD.1) values correspond to zero damage point. Required if IDF (DF.3) is 1.

| FIELD | VARIABLE | RECORD FIELD | VALUE | ATTRIBUTE TYPE | DESCRIPTION |
|---|---|---|---|---|---|
| 0 | KODE4 | (241-242) | DD | (C) | Card identification. |
| 1 | PERCNT1 | (243-248) | + | (R) | Direct damage in thousands of dollars ($1000) corresponding to initial depth value (should be zero). |
| 2 | PERCNT2 | (249-256) | + | (R) | Direct damage in thousands of dollars ($1000) corresponding to second depth value. |
| N | PERCNTN | (393-400) | + | (R) | Direct damage in thousands of dollars ($1000) corresponding to NSTAG (DF.2) depth value. 184 Continue on to an additional DD card as needed. |



GPO 785-065/20062