The Software Toolworks
Walt Bilofsky, Prop.

14478 GLORIETTA DRIVE
SHERMAN OAKS, CALIFORNIA 91423

TELEPHONE
(213) 986-4885

RECEIVED JUN - 6 1981

MASTER CATALOG SYSTEM (Ver. 8.1)
and
UTILITY PROGRAMS

Release 1.1
February 1981

Catalog System copyright (c) 1980, 1981 William W. Moss, M.D. CMP copyright (c) 1980 Walter Bilofsky. FIND and CHANGE copyright (c) 1980 Robert Wesson. CHECK and FDUMP copyright (c) 1980 James J. Gillogly. Sale of this software conveys a license for its use on a single computer owned or operated by the purchaser. Copying this software or documentation by any means whatsoever for any other purpose is expressly prohibited.

I. INTRODUCTION

One of the most powerful aspects of your computer is its ability to store many files containing text and other data. But in order to make use of this power, it is necessary to be able to keep track of all the files, which may be stored on a large number of diskettes. The Catalog System provides an automated way to accomplish this.

Your computer is not just a filing cabinet for text files; it is capable of more than simply storing data and printing it out. It can compare, search and revise text files. The five utility programs included in this package provide you with simple but powerful commands which you may use to perform these operations on your files.

The programs on the disk are:

Catalog System:

PCSD.ABS	Creates a master catalog on a single 5" drive.
PCDD.ABS	Creates a master catalog on dual 5" drives.
PCDK.ABS	Creates a master catalog on dual 8" drives.
SYSCAT.ABS	Types or prints reports from the master catalog.
DELCAT.ABS	Deletes disks from the master catalog.

Utility Programs:

CMP.ABS	Compares two files byte by byte.
FDUMP.ABS	Dumps files in up to five selectable formats.
FIND.ABS	Searches text files for a string or pattern.
CHANGE.ABS	Changes one string or pattern to another.
CHECK.ABS	Computes CRC checksum of files.

II. MASTER CATALOG SYSTEM

Version 8.1

By Dr. William W. Moss

1. INTRODUCTION

The HDOS Master Catalog System makes a complete catalog of all your diskettes. The catalog can be updated a single disk at a time and does not require the reading of all the disks each time a catalog listing is desired. There are a variety of listing options including selection of specific file names, extensions, volumes, or any combination of the three. File names can be specified using "wild cards". In addition the listing can be restricted to include only files created within certain date limits and to include or exclude files with specified flags. The listing output can be redirected to any HDOS device at run time. Three programs (PUTCAT, SYSCAT, and DELCAT) make up the system and are described below.

2. PUTCAT.

PUTCAT reads the directory of each disk and enters it in the master catalog. Three versions are supplied, for use with either single or multiple drive 5-inch disk systems or with multiple drive 8-inch disk systems. In each case, you provide a master catalog volume disk on which PUTCAT writes the master catalog. This should generally be a bootable disk with the Master Catalog System programs on it.

2.1. PCSD (Single drive 5-inch disk system)

(1) To run, mount the Master Catalog Volume in SY0: and type 'PCSD'.

(2) The system will respond with a request to load the first disk to be cataloged into SY0:. When the disk directory has been read, the request message will appear again. This continues indefinitely until the Master Catalog Volume is replaced in SY0:, at which time the directory files are written to the master disk.

(3) If the computer memory is filled before all the disks have been cataloged, the program will ask for the Master Catalog Volume to be reloaded so that the catalog files can be written and the memory freed up. After the files have been written you are again asked to insert a disk to be cataloged, and the process continues.

2.2. PCDD (Multiple drive 5-inch disk system)

(1) To run, mount the Master Catalog Volume in SY0: and type 'PCDD'.

(2) You are asked to place the disk to be cataloged in SY1:. As each disk is inserted, the corresponding catalog file is written to

the master disk on SY0:. When all disks have been cataloged, you should enter a control-C (^C; hold down CTRL and type C) and the program is terminated.

2.3. PCDK (Multiple drive 8-inch disk systems)

(1) To run, mount the Master Catalog Volume as the system disk (usually SY0:) and type 'PCDK'.

(2) You are asked to 'Place disk to be cataloged in SY1: and hit 'return' <^C to stop>?'. Insert the disks to be cataloged in the drive sequentially as requested. When all disks have been cataloged, exit the program by typing control-C (hold down CTRL and type C).

(3) PCDK can be used with any of the disk drives (5-inch or 8-inch) in any combination. The catalogs can be read from a 5-inch drive and recorded on an 8-inch drive or vice versa. The default devices are SY0: for the Master Catalog Volume and SY1: for reading disks into the system. These defaults can be changed at run time by calling the program by typing 'PCDK/'. You will be asked 'Disk drive for recording catalogs (e.g. DK0)?'; respond with a valid disk device name such as SY1 or DK3, etc. A colon is not required after the names. Then you are asked 'Disk drive for reading disks to be cataloged?'; again respond with a legal disk device name. Since reading and recording take place simultaneously, the same device cannot be used for both purposes.

Note: If you are using a DK device, it may be necessary to explicity load the DK driver. To do this, give the HDOS command "LOAD DK:" before running PCDK.

(4) Although this program can be used with multiple drive 5-inch systems, it is recommended that PCDD be used, as that program is faster and easier to use on such systems.

3. SYSCAT.

This program produces catalog listings from the directory files created by 'PUTCAT'.

(1) To run, type 'SYSCAT'. The program will then ask the following questions. Default responses to all queries are shown in square brackets just before the question mark; pressing RETURN is equivalent to giving the default response.

(2) 'Include system files [N]?'
To include the 'S' flagged files in the listing, respond with 'Y' (yes), else just hit return. The exclusion of 'S' flagged files provides considerably more room in computer memory to catalog the non-system files, which are usually the ones of interest. (See 'System Limitations' below.)

(3) 'Output device specifications:'

'Use defaults [Y]?'

If answered 'yes', the output format is 66 lines to a page (Form length) with 60 lines printed (Page length) and 6 blank lines between pages. It is also assumed that the output device does not use formfeeds to go to the head of the page. The next query is then (4) below.

If answered 'no', you will be asked to input the 'Page length?'. This value can be from 20 to 255.

You are then asked 'Output formfeeds [N]?'. If you answer 'no', then 'Form length?' is requested; it must be at least two greater than the 'Page length' and less than 255.

(4) 'Output to console [Y]?'
If you want the listing to go anywhere except the system console, then enter 'N'. You will then be asked for the 'File name?'. Any valid file specification can be entered (i.e. SY1:MASCAT, TEST.TXT, AT:, LP:, etc.). The default output device and extension are 'SY0:' and '.CAT' respectively. If the output is not to a disk drive, the program pauses to allow you to set up the output device (e.g. turn it on, set the paper to the head of a page, etc.). It will wait until a key is depressed on the console before proceeding.

(5) The files for each cataloged volume are then read into memory. As each is read, a data line is typed on the output device including (a) the volume number (followed by an asterisk if the volume is bootable), (b) the number of free sectors on the disk, (c) the date it was last cataloged, and (d) the volume label.

(6) When the last volume's catalog has been read, the entries of all the catalog files are sorted alphabetically. This takes about one to ten seconds, depending on the number of entries.

(7) 'File specification?'
You respond to this question in a similar fashion to the request format for the HDOS 'CAT' command. This instructs the program which files are to be listed. The general format is

[filename].[ext]/[vol]

where [filename] is one to eight characters, [ext] is one to three characters, and [vol] is one to three numbers plus a letter from 'a' to 'h' which is used to indicate multiple volumes with the same volume number. Any field not specified is assumed to be a wildcard (i.e. all possibilities will match it). A field wildcard can be explicitly requested using an asterisk, '*', for that field. For example, '*.BAS' will list all files with the extension '.BAS'. Any single character in a field can be matched with all possibilities by using a question mark, '?', as a wildcard. (e.g. 'CAT???' will list all files beginning with 'CAT' which have six or fewer characters in the [filename]; while '/2??' will list the files on disks with volume numbers ranging from 20a to 29h. Any combination of wildcards in different fields is permitted.

(8) The default device for the catalog data files is SY0:. If these files are on another device, the default can be changed by calling the program with 'SYSCAT/'. SYSCAT asks 'Which disk drive to enter catalog data files?'; respond with any legal disk device name. A colon is not required in the device name.

4. SPECIAL SYSCAT FUNCTIONS

When a hyphen, '-', is entered as the first character in a file specification, it is interpreted as a special function. The character following the hyphen determines the function to be performed, as follows:

B	Break	Close old and open new output device (^B)
D	Date	Set date limits
E	Exit	Exit to HDOS (^C)
F	Flags	Set flag inclusion or exclusion
P	Page	Go to top of next output page
R	Reset	Reset one of the disk drives
V	Volume	Print entire catalog by volume sequentially

Special function definitions:

(1) Break The current output device is closed and a new output file opened. SYSCAT asks the same series of questions as at the start of the program. If the new output file requires a device driver which has not been previously loaded (e.g. LP:), then there may not be enough memory to load it, depending on the memory size and the number of entries in the catalog. If such a situation occurs, the program will not accept that device name for output, and will ask for another one.

(2) Date A listing can be specified to include only files which were created between certain dates. When this option is selected, SYSCAT asks, 'Set date limits [Y]?'. Answer 'no' only if the prior date limits are to be eliminated. If 'yes', then 'From?' and 'To?' are requested; answer with the inclusive dates desired using the HDOS date format (e.g. 20-MAR-80). If an invalid date is entered, the process begins again. If date limits are set, they are printed at the bottom of the catalog listing.

(3) Exit Closes the output file and terminates the program. This can also be accomplished by typing ctrl-C (hold down CTRL and type C).

(4) Flags Files marked by specified HDOS flags can be either included or excluded from the listing. SYSCAT asks, 'Set flag restrictions [Y]?'; respond 'no' only to abolish a prior restriction. If 'yes', then SYSCAT asks, 'Include (I) or Exclude (E)?'; respond with the appropriate letter depending on whether the flagged

files are to be included in or excluded from the listing. 'Which flags?'; respond with any combination of the three allowed HDOS file flags, 'S', 'W', or 'L'. If more than one flag is requested, only one of them need be set to meet the test for inclusion or exclusion. As with date limits, if flag restrictions are set, they are printed at the bottom of the listing.

(5) Page Fills the remainder of the current output page with blank lines and starts a new page.

(6) Reset To direct output to a disk volume which has not been loaded, the reset option can be used to mount that disk. SYSCAT will ask 'Reset which disk drive?'; respond with a legal HDOS disk device name, such a SY1: or DK3:, etc. If the HDOS system disk is to be reset, enough free memory must be available to load both of the HDOS overlays, otherwise an error message will be printed. Once the reset has been completed, a 'Break' is performed (as above), so that a new output file can be opened.

(7) Volume Prints a listing of the entire catalog by volume number sequentially. All files are listed unless date or flag restrictions have been set previously.

6. CONTROL-KEY INTERRUPTS

At any time during the running of the program, it can be immediately aborted by a control-C (^C). This is the same as using the '-E' function except that it will stop the output at once and exit to HDOS.

To abort a listing but not exit the program, type a control-B (^B). The current output file is closed and the 'Break' function is executed (as above).

7. DELCAT

If a disk is removed from use and not replaced by another with the same volume number, it should be deleted from the catalog by running 'DELCAT'.

(1) To run, mount the Master Catalog Volume on SY0: and type 'DELCAT'. (Note: see (3) for using "SY1:")

(2) DELCAT asks for the number of the volume to be deleted. The labels and subvolume letter designations ('a' through 'h') for each disk with the specified volume number are listed. You are asked 'Which volume to delete ('a' through 'h')?'. Respond with the appropriate letter. If you do not wish to delete a volume, simply press 'return' in response. The process continues until a control-shift-C (^C) is entered.

(3) The default device for DELCAT is SY0:. If the catalog data files are on a disk mounted on another drive, the default can be changed by typing 'DELCAT/' when calling the program. In this case, DELCAT asks 'Which disk drive (e.g. DK0)?'; respond with a legal disk device name and the data files will be read from that disk.

8. SYSTEM LIMITATIONS

8.1. Volume numbers

The Master Catalog System keeps track of the various disks entered by means of the disk volume numbers and labels. Thus, all disks must have different numbers and/or different labels to avoid confusion. If different disks with the same number are cataloged, they are assigned subvolume letter designations. Up to eight disks can be cataloged with a single volume number and are designated 'a' through 'h' in the catalog. When a disk is updated in the catalog, the same letter is retained to identify it.

Caution: HDOS requires that disks all have unique volume numbers to assure correct operation. If multiple disks with the same volume number are used, it is easily possible to destroy a disk's directory and thereby lose all the files on that disk. Good practice dictates that unique volume numbers be used whenever possible. The HUG program 'DUP' is recommended for changing volume numbers to unique values.

8.2. Computer memory

The maximum number of files that can be cataloged at one time can be approximated by the following equation:

(# files) = (Kbytes memory) * 51 - 575

Thus, in a 48K system: (# files) = 48 * 51 - 575 = 1873 files.

Subtract 25 to 50 files for each device driver loaded.

8.3. Disk space

As each file catalog requires at least two sectors of disk space, at most 182 different disk catalogs can be stored on a single nonbootable 5-inch data disk (created using PCDK). The data disk must have enough free sectors to acommodate the largest catalog file which might be updated.

III. UTILITY PROGRAMS

1. Introduction

The Utility Programs allow you to perform certain operations on files containing text or other data. These programs are software tools. Just as a screwdriver, pliers, and a ruler help you manipulate and measure physical objects, these tools allow you to manipulate and measure files of data.

CMP performs a byte by byte comparison of two files. It is useful for finding the first point of difference between the files.

CHECK computes a CRC (cyclic redundancy checksum), which is a number computed from all the bytes in a file. It is useful for checking that a file has been copied or transmitted correctly.

FDUMP dumps the contents of a file, byte by byte or word by word. It prints out in up to five selectable formats: octal, hex, decimal, ASCII, and split octal. Occasionally, text files contain control characters which do not type out but can cause strange behavior; one use of FDUMP is to detect these characters.

FIND searches one or more text files for a given character string or text pattern. Since the names of the files to be searched can be specified by "wild card" constructs, it is easy to use FIND for jobs such as searching a lot of documents for references to a certain word or phrase, or searching program files for all uses of a variable.

CHANGE is similar to FIND, but generates an output file by replacing the specified string or pattern with a replacement string.

Note: These programs all read files in chunks as large as available memory permits. This reduces both the running time of the program and wear on the disks and drives. As a result, the programs may read, and then "think" for a long time without producing any output. This is normal operation and not a cause for concern.

2. Wild Cards

CHECK, FDUMP and FIND operate on a list of input file names. Often it is useful to specify a group of files having similar names. To make this easy, these programs recognize "wild card" file name constructs. Many HDOS and CP/M programs also recognize wild cards, but CHECK, FDUMP and FIND use a slightly different specification method, as follows:

In file names, the character "?" matches any single character (including a blank). The character "*" matches zero or more characters, including the character ".". Device names (e.g., LP:, SY2:) are not included in wild card matching, except that "?" may be used in place of the drive number.

As an example,

```
>check sy?:?i*a*
```

will print the checksum of each of the following files (assuming they exist):

```
sy1:file.asm
sy0:directa
sy1:findfile.mac
sy2:cia
```

The major difference from HDOS and CP/M is that here the filename and extension are not treated as separate fields. Thus, * is the same as *.*. Also, * has a more general meaning than in HDOS and CP/M wild cards.

3. FIND/CHANGE -- Line-oriented Pattern Matching Programs

Version 1.0 -- November 1980

Robert B. Wesson

3.1. Description

Find is a pattern-matching program which searches each line in one or several files for a given text pattern. Depending on the options selected, find will print out all lines containing that pattern, with or without line numbers, or lines which do not contain the expression, or just a count of the lines found. Find can be used to look for all occurrences of a specific string of characters. Or it can look for all strings which match a general pattern description called a regular expression, described in Section 3.2 below.

Suppose you wish to locate all the places you've used the variable "C4" in all your BASIC programs on a particular disk. The following interaction will accomplish this (program output is underlined):

```
> find *.BAS
Pattern:  C4

MAIN.BAS
00090 C4 = A1 + 10
01020 PRINT C4

SUB1.BAS

SUB2.BAS
00500 C4=C4-1:C5=C5+1
00590 PRINT "Out of bounds magnitude: ", C4: GOTO 900
```

Find is similar to both the find program of Kernighan and Plauger [1] and the grep UNIX utility program [2].

Change is an extension of find which allows you to replace, extend, or delete the pattern you've found. In addition to a regular expression pattern to be matched, it requests a string which will replace the matched string. You can optionally ask change to check each such replacement with you before making it.

For example, suppose you think you've used the word "study" too often in a report. You might use change to create a revised version of the report as follows:

```
> change -v report.txt revised.txt
From pattern: study
To string: examine

occurred to us to examine the data more carefully. When we
OK? yes

participated in this examine. They helped us greatly.
OK? no

examineing how well our results correlate with previous
OK? no
```

The remainder of this documentation describes these twin programs. After a section defining the patterns that both recognize, each program is described in its own section.

3.2. Patterns

Both find and change ask for the pattern you wish to locate in the file(s). You must type a pattern like those described below (no delimiters, blanks are significant) followed by a RETURN.

The pattern may be a string of characters, or a more general pattern known as a regular expression. The following simple rules describe the patterns which can be requested:

c	Any single character matches itself.
.	A period (.) matches any single character.
^	Matches beginning of line.
$	Matches end of line.
[...]	Brackets match any character they contain. This construction is called a "character class." Abbreviations for continuous sequences of characters may be used; i.e., [a-z] matches any lower-case alphabetic character, and [a-zA-Z0-9] matches any alphanumeric character.

[^...] Matches any character not in the character class. For example, [^0-9] will match anything but a digit.

* Matches zero or more instances of the preceeding single expression. This construction is called a "closure." It matches the longest string possible. For example, [a-z]* will match the longest series of lower-case characters it can find, consistent with the rest of the pattern. The construction [a-zA-Z][a-zA-Z0-9]* will match Fortran identifiers.

+ Like *, but matches one or more occurrences of the previous pattern element. Saying [ABC]+ is exactly equivalent to saying [ABC][ABC]*

\c If c is a character with a special meaning (like * or .), \c is a normal c with no special connotations. Also, newlines (\n) and tabs (\t) may be included in pattern expressions. \\ matches the single character \.

Special characters lose their meanings when preceded by \, inside [...], or for:

^ not at beginning of pattern
$ not at end of pattern
*,+ at beginning of pattern

Special meaning of a character within brackets [...] is lost when escaped or for:

^ not at beginning
- at beginning or end

Some Example Patterns:

Pattern	Matches
^[A-Z]	a capital letter at the beginning of a line,
\.$	a period at the end of a line
[cC]h[a-z][a-z][a-z]	five-letter words beginning with "Ch" or "ch"
.*x	the longest string ending with x (including the string "x")
[^.,?;:"'!]	any line which does not contain a punctuation mark
pa[cht]*er	words like "patcher, "patter"

3.3. Using Find

To use find, type a command described by the following syntax: (Note: command line elements enclosed in brackets [...] are optional.)

```
find [-options] file1 [file2 ... filen]
```

Find will prompt for the pattern to be matched and begin processing. The files can be any legal filenames, including wildcards (* and ?), and the options are characters from the following set

-f Do not list filenames as they are being processed. Normally, the name of each file is listed in the output before the lines within it which match. This flag suppresses that listing. It is useful when you are redirecting the output (see below) to create another file of the matches and you do not wish that file to contain lines other than those matched.

-h print a help message on the terminal. Find will ignore all other elements on the command line, print a synopsis of usage and patterns, then return to system command level.

-n Preface each output line with a line number. Line numbering is reset with each new input file.

-x Output all lines except those which contain the pattern. This flag reverses the normal operation of find, allowing you to search for lines which do not contain some pattern.

Since find is written in C/80 [3], I/O redirection is allowed. Normally, find requests the pattern to be searched for from the terminal (TT:) and lists the matched lines there as well. By using the '<filename' and '>filename' conventions, you can cause find to read the pattern from a file, and send its output to a file. In the former case, find does not prompt for the pattern, but simply retrieves it from the named file. For example, the command

```
find -x -n -f inputfile <patfile >outfile
```

will cause find to get the pattern to be matched from 'patfile,' search 'inputfile' for lines which do not contain that pattern (-x), and output each such line prefaced by a line number (-n) to 'outfile.' The name 'INPUTFILE' will not appear in 'outfile' (-f).

3.4. Using Change

The syntax of the command line for change is more resrictive than that of find. Only one input file and one output file can be specified:

```
change [-options] from-file [to-file]
```

If no "to-file" is specified, output will be to the terminal. If one is specified, it cannot be the same as the input file; i.e., you must change a file into another, not the same, file. The options allowed are:

-h Print a help message with a synopsis of usage and

patterns, then return to system command level.

-n Output line numbers as in find.

-v Verify each changed line with the user before writing it. Change will send the changed line to the terminal with the question "OK?" If your answer begins with "y," the line will be written to the output file as is; otherwise, the original line will be written as though no match had occurred.

After the command line has been processed, change will first request a "from pattern," which should be a regular expression as described above. Then it will request a "to string," which should be a sequence of characters which will replace the matched string in each line. The special character & means "the string in the original line which matched the pattern." Thus, you can perform simple replacement:

```
change from-file to-file
From pattern:  <bad text>
To string:  <good text>
```

or you can insert text:

```
change from-file to-file
From pattern:  <bad text>
To string:  &<suffix text> or <prefix text>& or ...
```

or you can delete text:

```
change from-file to-file
From pattern:  bad text
To string:
```

3.5. Restrictions and Internals

The option flags must preceed the input filenames in the command line. Redirected I/O can appear anywhere within the command line. Both programs will complain and return to system command level if any of the following are encountered:

- an option switch is not recognized
- a file cannot be opened
- the output filename is the same as the input filename (change only)
- an illegal pattern was requested
- an illegal replacement string was requested (change only)

Input file lines must be less than 1000 bytes in length; the pattern must be less than 256 bytes long. This latter restriction may cause problems with character class patterns, since the class expands to the length of its membership or so. For example, [a-z] takes 28 bytes after expansion. Other pattern types are not so

critical: ^ and $ take one byte each, single characters take two bytes, and * takes four bytes.

3.6. References and Notes

1. Kernighan, B. W. and Plaugher, P. J. Software Tools, Addison-Wesley Publishing Co., Reading, Mass., 1976.

2. For information about grep in particular and UNIX in general, see the UNIX Programmer's Manual, Bell Telephone Laboratories, Inc., 1979.

3. For more information about C/80, see its programming guide, available through "The Software Toolworks," 14478 Glorietta Dr., Sherman Oaks, CA 91423.

4. FDUMP - Display file contents

Dr. James J. Gillogly
2520 Chard Avenue
Topanga CA 90290
15 Nov 1980

4.1. DESCRIPTION

FDUMP reads a specified file or files and prints the contents in octal, decimal, hex, ascii, and/or split octal, either by individual bytes or by words. FDUMP is useful for debugging programs that do file output, inspecting the contents of system files, determining which characters are tabs and which are spaces, and many other applications.

4.2. USAGE

Give the command

```
fdump filename
```

or

```
fdump [flags] filename [filename ...]
```

The flags are described below. Filenames may be specified with the usual "*" and "?" conventions; e.g.

```
fdump syl:*.abs
```

will produce a dump of all executable files on drive syl:.

The flags control the way the data in the file is displayed.

The default command produces a listing with each byte displayed in octal and the addresses displayed in decimal. For example,

```
>fdump sy1:addr
```

produced the following results:

```
Fdump v1.0 15 Nov 80
SY1:ADDR
  0: 104 162 056 040 112 141 155 145 163 040 112 056 040 107 151 154
 16: 154 157 147 154 171 012 062 065 062 060 040 123 056 040 103 150
 32: 141 162 144 040 101 166 145 056 012 124 157 160 141 156 147 141
```

and so forth.

4.3. ADDRESS FORMATS

If you prefer the address in octal rather than decimal, give the command

```
>fdump -no filename
```

Similarly, the flag "-ns" specifies split octal addresses, "-nx" or "-nh" will produce hex addresses, and "-nd" will go back to decimal.

4.4. DATA FORMATS

The data may be displayed in a wide variety of formats. For example, the command

```
>fdump -a sy1:addr
```

produces an unambiguous ASCII listing of the same file shown above:

```
Fdump v1.0 15 Nov 80
SY1:ADDR
  0:   D   r   .       J   a   m   e   s       J   .       G   i   l
 16:   l   o   g   l   y  \n   2   5   2   0       S   .       C   h
 32:   a   r   d       A   v   e   .  \n   T   o   p   a   n   g   a
```

Using ASCII output mode, non-printable characters are identified as either escapes (using the C language conventions), control characters identified by '^', or octal numbers above 177.

The available formats are:

-a	ASCII byte
-o	octal byte (the default)
-d	decimal byte
-x	hexidecimal byte (-h also works)
-s	split octal (word only)
-ow	octal word
-dw	decimal word
-xw	hex word

Any combination of these formats may be used. For example, the command

```
>fdump -o -a -dw -xw addr
```

applied to the file of the previous example yields:

```
Fdump v1.0 15 Nov 80
SY1:ADDRESS
  0: 104 162 056 040 112 141 155 145 163 040 112 056 040 107 151 154
  0:   D   r   .       J   a   m   e   s       J   .       G   i   l
  0:   17522   11808   19041   28005   29472   18990    8263   26988
  0:    4472    2E20    4A61    6D65    7320    4A2E    2047    696C
 16: 154 157 147 154 171 012 062 065 062 060 040 123 056 040 103 150
 16:   l   o   g   l   y  \n   2   5   2   0       S   .       C   h
 16:   27759   26476   30986   12853   12848    8275   11808   17256
 16:    6C6F    676C    790A    3235    3230    2053    2E20    4368
 32: 141 162 144 040 101 166 145 056 012 124 157 160 141 156 147 141
 32:   a   r   d       A   v   e   .  \n   T   o   p   a   n   g   a
 32:   24946   25632   16758   25902    2644   28528   24942   26465
 32:    6172    6420    4176    652E    0A54    6F70    616E    6761
```

4.5. SKIPPING DATA

Data at the beginning of the file can be skipped by specifying in a flag the number of bytes to ignore. FDUMP will begin on the 16-byte boundary preceding the specified number. For example, use

```
fdump -3000 fdump.abs
```

to look at all but the first (about) 3000 bytes of this executable file. The number must be in decimal.

5. CMP - Compare Two Files

Walt Bilofsky
December 1980

CMP reads two data files and compares them byte by byte. If the files are identical, it says so. Otherwise, it prints the byte, sector and line number of the first difference. Optionally, CMP can print every difference.

To use CMP, give the command

```
cmp file1 file2
```

or

```
cmp [-l] [-t] file1 file2
```

or

```
cmp [-l] [-t] file1 file2 >outputfile.
```

where [-l] means you can specify "-l" or leave it out.

If the -l appears, each byte which is not the same in the two files is printed out; otherwise only the location of the first difference is reported.

It is possible for two text files to contain the same text but still differ as far as CMP is concerned. For example, if one HDOS text file contains a 0 byte, this can cause the files to appear different, although for text purposes the 0 byte is ignored. Or two CP/M text files can differ following the end of file (ctrl-Z) but still contain the same text.

This will not usually occur. However, if CMP reports a difference between two text files which seem identical, the -t argument can be specified. This will ignore the above differences and report only files in which the text differs. Note, however, that files which contain tabs will still be different from files which contain equivalent spaces.

CMP usually prints on the terminal. However, if the ">outputfile" argument appears, the output is written onto the file or device whose name is "outputfile".

6. CHECK - Checksum by File

Dr. James J. Gillogly
2520 Chard Avenue
Topanga CA 90290
April 1980

Check reads a specified file or files and types a 16-bit checksum for each, using the same CRC (cyclic redundancy checksum) as is used for checksumming of HDOS disks. Check is useful for verifying that two files are identical if the files cannot be compared directly (e.g. if they are on different disks, or in different physical locations).

To use Check, give the command

```
check filename filename ...
```

or

```
check filename filename ... >outputfile
```

If no filename is specified, the program will ask the user for one. Any number of files may be given, and wild card constructs may be used to specify file names. The checksums are computed and printed on the terminal. Or, if the ">outputfile" is specified, the output is written to the device or file named "outputfile".