Intro to ECL / IPF

Hands on Lab Manual

ECL Basic Exercises

Using ECL

Exercise 1: Creating Files in ECL

Note: Run these from the user screen

1. Creating Permanent Files:

The @CAT statement catalogs a file in a master file directory without having it assigned to the run.
To use @CAT as a transparent statement, enter two masterspace (@) characters instead of one
Format :

```
        @CAT[,options] filename[,type/reserve/granule/maximum,pack-id-1/...
/pack-id-n,,,ACR-name]
```

Steps :

a. To create and catalog a file named SAMPLE FILE, use:
   @CAT,P SAMPLEFILE
b. To use the transparent statement to create and catalog a file named SAMPLEFILE2, use:
   @@CAT,P SAMPLEFILE2
c. Verify that the files are created using the @PRT command.
   @PRT SAMPLEFILE.
   @PRT SAMPLEFILE2.

2. List Elements in a file:

Use the @PRT statement with the appropriate options to list all matching elements.

Steps:

a. To list all symbolic elements in a file named DPS, use:

```
@PRT,TSV SYS$LIB$*DPS(3)./*
```

3. Copy File to File :

The @COPY statement copies a file or element to another file. Mass storage files can be either word addressable or sector formatted

All parameters of the @COPY statement are optional

Format:

```
@COPY[,options] [name-1,name-2,{ number-of-files | subtype-list } ]
```

Example :

```
@COPY,AOV FI./VERS,FO
```

All non deleted executable and omnibus elements (Option O) in program file FI having a version name of VERS are copied into program file FO. Element and version names remain unchanged

Steps:

a. Create and catalog a new file.

```
@CAT,P SAMPLEFILE3
```

b. Assign the file to be able to modify it in the run.

```
@ASG SAMPLEFILE3
```

c. Make a copy of an existing file with all elements

```
@COPY SYS$LIB$*DPS(3).,SAMPLEFILE3
```

d. Release the assignment

```
@FREE SAMPLEFILE3
```

Exercise 2: Deleting Files in ECL

1. The Delete command deletes cataloged files and program file elements:

Format:

```
@DELETE[,options] name-1 [,name-2,...,name-n,subtype-list]
```

Steps:

   a. To delete the file named SAMPLEFILE3, use:

   ```
   @DELETE SAMPLEFILE3
   ```

2. When you want to delete a cataloged file, only the N option applies.
   The N option is available for the file DELETE command
   You can also use the @FREE,D statement to delete files

Steps:

   a. You need to assign the file using @ASG

   ```
   @ASG SAMPLEFILE2
   ```

   b. To delete the file, use:

   ```
   @FREE,D SAMPLEFILE2
   ```

3. Deleting Elements - One or more elements can be deleted in a program file, based on element name,version name and element type
   Example :
   The following examples illustrate the use of the @DELETE statement:

   ```
   @DELETE FLAP.,TARE5.,ZEBRA4.,BAKER
   ```

Steps:

a. Repeat exercise 2 and make a copy of the file.
b. Find a list of elements in this file.

```
@PRT,TLA SAMPLEFILE3.
```

c. Delete the DPS$CFBK absolute element

```
@DELETE,A SAMPLEFILE3.DPS$CFBK
```

d. Verify that the element is deleted (marked with *)

```
@PRT,TLA SAMPLEFILE3.
```

4. Undeleting Elements - One or more elements can be undeleted (changed from deleted to current) in a program file.
   If a program file initially has no deleted elements, an element @DELETE can be followed by an @DELETE,U with the same options and specifications to restore the program file elements and procedure table entries to their original state

Steps :

a. To undelete the element that was deleted in the previous exercise, use:

```
@DELETE,UA SAMPLEFILE3.DPS$CFBK
```

b. Verify that the element is now available

```
@PRT,TLA SAMPLEFILE3.
```

Exercise 3: Assign Files and Use Alias

1. Use @ASG statement to assigns a random-access, mass storage file to your run

   Format

   @ASG[,options] filename[,type/reserve/granule/maximum /placement][,pack-id-1/.../pack-id-n,,,ACR-name]

2. When you assign a previously created file, you must pass security validation in order to gain access to the file

   You receive access according to the validation rules for mandatory access control and discretionary access control

3. Assigning Files on Shared Devices

Steps:

   a. Use assign on a new file.Assume that the following examples are all in the same run:

      @ASG NEWBASE1.

   b. The system initially assumes that NEWBASE1 is a cataloged file. If it cannot find the file, the system assigns the file as a temporary file

   c. Catalog another file named NEWBASE2

      @CAT,P NEWBASE2

   d. Assign the other existing file to the same run.

      @ASG,A NEWBASE2.

      This statement assigns the file NEWBASE2 to the run. The A option means the file has been previously cataloged

Exercise 4 : Erase and release a File

1. The @ERS statement releases mass storage from cataloged or temporary, sector-addressable or word-addressable files
   All @ERS parameters are optional
   Format :
   @ERS[,options] [filename-1,filename-2, . . . ,filename-n]

Steps:

a. Create a new file and catalog it. Ensure the max size of the file matches the file with content that we will copy in the next step.
   @CAT,P TESTFILE1,///9999
b. Copy an existing file with elements to this file
   @COPY SYS$LIB$*DPS(3).,TESTFILE1
c. Verify that the file is listed in the output of the PRT statement and has all the elements
   @PRT,TL TESTFILE1.
d. Erase the file storage using the ERS statement
   @ERS,I TESTFILE1
e. Verify that the file is now empty
   @PRT,TL TESTFILE1.

Exercise 5 : Miscellaneous Commands in ECL

1. Use Pack command to rewrites an entire program file, removing specified types of elements (depending on the options specified) and all elements marked as deleted

Format:

```
@PACK[,options] [filename-1,filename-2,...,filename-n, subtype-list]
```

Steps:

a. Create a new file and catalog it. Ensure the max size of the file matches the file with content that we will copy in the next step.

   ```
   @CAT,P TESTFILE2,///9999
   ```

b. Copy an existing file with elements to this file

   ```
   @COPY SYS$LIB$*DPS(3).,TESTFILE2
   ```

c. Verify that the file is listed in the output of the PRT statement and has all the elements

   ```
   @PRT,TL TESTFILE2.
   ```

e. Delete the DPS$CFBK absolute element

   ```
   @DELETE,A TESTFILE2.DPS$CFBK
   ```

f. Verify that the element is deleted (marked with *)

   ```
   @PRT,TL TESTFILE2.DPS$CFBK
   ```

g. Use PACK to remove the deleted element

   ```
   @PACK TESTFILE2.
   ```

h. Verify that the element is removed from the file

   ```
   @PRT,TL TESTFILE2.DPS$CFBK
   ```

2. Using @PRT you can list files and elements with the LIST processor.

Format

```
@PRT[,options] [name-1,name-2,...,name-n]
```

The @PRT statement reports the files from the local MFD and from the shared MFD, if a shared directory is available

The @PRT,T statement provides the following three levels of information detail for listing the Table of Contents

- Long format (@PRT,TL)
- Short format (@PRT,T)
- Summary format (@PRT,TN)

Steps:

a. Display a full list of files

   ```
   @PRT
   ```

b. Create a new file and catalog it. Ensure the max size of the file matches the file with content that we will copy in the next step.

   ```
   @CAT,P TESTFILE2,///9999
   ```

c. Copy an existing file with elements to this file

   ```
   @COPY SYS$LIB$*DPS(3).,TESTFILE2
   ```

d. Display the short format of the table of contents of the new file

   ```
   @ASG TESTFILE2
   @PRT,T TESTFILE2.
   ```

e. Display the long format of the table of contents of the new file

   ```
   @PRT,TL TESTFILE2.
   ```

f. Display the summary format of the table of contents of the new fie

   ```
   @PRT,TN TESTFILE2.
   ```

3. The @QUAL statement lets you use implied directory-IDs or qualifier names for subsequent control statements involving references to external file names. For @PRT, the @QUAL,D statement does not affect the selection of directories

Format:

```
@QUAL[,options] [directory-id#][qualifier]
```

The summary information output for all formats is as follows:

filename type, a/b elements, c/d tracks.

The summary information output for all deleted elements format is:

*File contains e deleted element[s] and f track[s] of deleted text

Steps:

a. Specify a Qualifier for the rest of the steps in this exercise
   @QUAL,D TEST
b. Create a new file and copy the content from another existing file with elements into it, as done earlier
   @CAT,P TESTFILE2,///9999
   @COPY SYS$LIB$*DPS(3).,TESTFILE2
c. Use the PRT statement to list the summary contents of the file.
   @PRT,TN TESTFILE2.
   Verify that the output reflects the usage of the qualifier
d. Change the qualifier and repeat step b
   @QUAL,D TEST2
e. Create a new file and copy the content from another existing file with elements into it, as done earlier
   @CAT,P TESTFILE3,///9999
   @COPY SYS$LIB$*DPS(3).,TESTFILE3
f. Verify that the output of PRT lists the new file with the new qualifier
   @PRT,TN TESTFILE3.
g. Cleanup
   @DELETE TESTFILE2; @DELETE TESTFILE3

IPF Basic Exercises

Using IPF

Exercise 6: Starting IPF

1. Calling IPF 1100
    a. To call IPF 1100, type the following call in the user screen and press ENTER:

    ```
    >@IPF
    ```

    b. IPF 1100 responds with the current date and then shows a prompt for commands.
    c. Type HELP
    d. Enter 1 to view the General Overview of the help
    e. Enter - to go back to the menu
    f. Enter 2 to view the Command Menu
    g. Select 1 to see the complete list
    h. Select 1 to view the help for the ACCEPT command
    i. Navigate back to the main menu using -

2. Using IPF 1100 Commands
    a. IPF 1100 commands have this format : command-name keyword=user-supplied-value keyword=user-supplied-value
    We can specify above keywords in any order to CREATE command or equivalent

b. The format of all keyword parameters is :
KEYWORD=user-supplied-value

c. Here is an example of an IPF 1100 command format:
CREATE FILE=filename

d. Create 3 files named TESTFILE1, TESTFILE2, TESTFILE3

```
CREATE FILE=TESTFILE1
CREATE TESTFILE2
CREATE TESTFILE3
```

e. You can abbreviate most IPF 1100 command verbs and keywords to their first four characters

Examples :

CREATE	CREA
POSITION	POSI
DEVICE_TYPE	DEVI_ TYPE

f. Test 2 other commands with their abbreviations

Enter HELP -> 2 -> 1 -> 1 to view the help for the ACCEPT command

Enter ACCE VARI=$T1

Enter hello, followed by Enter

ENTER DISPLAY VALUE=$T1

Enter HELP -> 2 -> 1 -> 11 to view the help for the DECLARE command

Enter DECL VARI=$T1 VALUE=CURRENT

You should see a message indicating that the declare command can only be used within an IPF 1100 procedure

3. Receiving Feedback

a. Set the system variable $COMPLETIONS to TRUE.

```
$COMPLETIONS := TRUE
```

b. IPF 1100 displays a completion notice for doing this.

c. Enter a PURGE command to delete file TESTFILE2.

```
PURGE TESTFILE2
```

d. A completion notice verifies deletion of the file.

e. Change $COMPLETIONS to FALSE.

```
$COMPLETIONS := FALSE
```

f. Enter a PURGE command to delete file TESTFILE3

```
PURGE TESTFILE3
```

g. Since you requested no completion notices, IPF 1100 deletes the file and only returns the prompt and the cursor

4. Controlling Screen Output Paging

a. Use $PAGING for determining whether IPF 1100 generates page breaks

Enter the following two commands.

```
$PAGING := TRUE
REVIEW VARIABLES=SYSTEM
```

You should see one page of the list of the variables, with a message prompting you to enter stop to cancel output.

Hit enter a few times, then type stop and enter.

b. For displaying number of lines and number of columns of output,we can use the variables: $PAGELENGTH, and PAGEWIDTH

Enter the following two commands.

```
$PAGELENGTH := 10
REVIEW VARIABLES=SYSTEM
```

You should now see the output with 10 lines per page.

Enter the following two commands.

```
$PAGEWIDTH := 50
REVIEW VARIABLES=SYSTEM
```

You should now see the output display in 50 columns, with extra content wrapping to the next line.

c. Use the system variable $WAITTIME with $PAGING for a more flexible output environment

   Enter the following two commands.

   ```
   $WAITTIME := 5
   REVIEW VARIABLES = SYSTEM
   ```

   You should now see that the terminal displays 10 lines at a time and automatically scrolls to the next page after 5 seconds.

d. Use $PAGECALL for determining whether IPF 1100 pauses before or after it displays processor output. $PAGECALL can have the following values: NEITHER, BEFORE, AFTER, BOTH

   ```
   $PAGECALL := AFTER
   ```

e. Reset the Paging variables to complete the exercise

   ```
   $PAGELENGTH := 24
   $PAGEWIDTH := 80
   $PAGING := FALSE
   ```

5. Signing Off Your System

   a. To leave IPF 1100, enter

```
>LOGOFF
```

b. If you want to sign off your computer system, enter

```
>@FIN
```

c. You will receive an accounting summary on your screen

Exercise 7 : Working with System Variables

a. Set a system variable - SET command

Format:

```
SET system-variable:=value or system-variable:=value
```

Example to try:

```
SET $DELIMCHAR := "/"
```

b. SET Which gives error message

If you try to change the value of a system variable you are not allowed to change, IPF 1100 displays an error message:

```
**VARIABLE130 You may not change the value of this system variable
```

Example to try:

```
SET $BREAKPOINT := TRUE
```

c. Show the current value of a system variable - DISPLAY VALUE command

Example to try:

```
DISPLAY VALUE=$DATE
```

Exercise 8 : Working with User variables

a. Giving a Value to a Variable - SET command

Format

```
SET variable:=exp (where the word SET is optional)
```

Example to try:

```
SET %USRVAR1 := BLUE
```

b. Printing the Values of Variables - DISPLAY VALUE command

Format

```
DISPLAY VALUE=exp
```

Example to try

```
DISPLAY 10+17
        displays 27
```

c. Listing Variable Names - REVIEW command

```
REVIEW VARIABLES=USER
REVIEW VARIABLES=SYSTEM
```

d. Soliciting a User Variable Value - ACCEPT command

Format

```
ACCEPT VARIABLE=var
        [ PROMPT=string-exp ]
        [ FROM= { USER  OPERATOR } ]
```

Example to try

```
ACCEPT %X
ACCEPT VARIABLE=%NAME
```

Exercise 9 : System Functions

a. Using the lowercase function

   Format

   ```
   $function-name(arg-1 [,arg-2 ...])
   ```

   Example to try

   ```
   SET %X := $LOWERCASE(AbCdE)
   DISPLAY %X
           returns abcde
   ```

b. Finding the Fractional Portion of a Number ($FRACTION)

   ```
   DISPLAY $FRACTION(5.4)
   DISPLAY $FRAC(90.34)

   SET %portion := (-3.456)
   DISPLAY $FRAC(%portion)
   ```

c. Finding the Integer Portion of a Number ($INTEGER)

   ```
   DISPLAY $INTEGER(5.4)

   SET %Y := 10.5
   DISPLAY $INTE(%Y)
   ```

Exercise 10 : Working with Strings

a. Finding the Length of a String ($LENGTH)

```
SET $DELIMCHAR := /”/
DISPLAY $LENGTH(elephant)
        displays 8
DISPLAY $LENGTH("These are a few words")
        displays 21
```

b. Finding the Position of a Substring in a String ($SEARCH)

Format

```
        $SEARCH(string1, string2[,start])
```

Example to try

```
DISPLAY $SEARCH(handicap, cap)
        displays 6
DISPLAY $SEARCH("New York, New York", York, 9)
        displays 15
DISPLAY $SEARCH(fish, tuna)
        displays 0
SET %NAME:=THOMASWATSON
DISPLAY $SEARCH(%NAME, WATS)
        displays 7
```

c. Returning a Substring within a String ($SUBSTRING)

Format

```
        $SUBSTRING (string, offset [,length])
```

Example to try

```
DISPLAY $SUBSTRING(FOOTBALL, 5, 4)
        displays BALL
```

```
DISPLAY $SUBSTRING(entertainment, 1, 5)
        displays enter
```

d. Case-insensitive removal of the character - $TRIM

Example to try

```
DISPLAY $TRIM("   abc   ")
DISPLAY $TRIM("   abc   ", RIGHT)
DISPLAY $TRIM("   abc   ", LEFT)
DISPLAY $TRIM("   abc   ", BOTH)
```

e. Adding a "pad" to a String ($PAD)

Format

```
$PAD(string, length [, side, character ])
```

Example to try

```
DISPLAY $PAD(abc,5)
DISPLAY $PAD(abc, 5, left)
DISPLAY $PAD(ABC, 6, BOTH, X)
```

f. Converting a Numeric Field to a String ($STRING)

```
DISPLAY $STRING(5)
```

g. Using More Complex Expressions

```
DISPLAY $SUBSTRING(CARDIOGRAM,
$SEARCH(PROGRAMMING, GRAM)+3,4)
```

Exercise 11: Using the workspace / lookspace in line mode

This section shows how to tell IPF 1100 which file you want to work on and how to save the changes that you make to it. You use EDIT 1100 to make changes to a copy of your file in your workspace.

IPF 1100 assigns the workspace when you call IPF 1100. The name of this file is Project-idoriginal-run-id

a. Create files to work with named SAMPLEFILE and SAMPLEFILE2

```
CREATE SAMPLEFILE
CREA SAMPLEFILE2
```

b. To copy the file into the workspace where you can update it, enter

```
OLD SAMPLEFILE,WORKSPACE
```

c. To copy the file into the lookspace where you can scan it (but not change it),

```
OLD SAMPLEFILE,LOOKSPACE
```

d. You can move from one area to the other by using the SWITCH command. When you want to edit the other area, use the SWITCH command, and specify which area you want to edit as follows:

```
SWITCH LOOKSPACE
SWITCH WORKSPACE
```

When one editing area is active, the next OLD command with no area specified copies the specified symbolic file into the active area.

e. To replace the SAMPLEFILE, enter

```
REPLACE
```

f. To copy data file SAMPLEFILE2. into the workspace and make updates, enter

```
OLD SAMPLEFILE2
```

Exercise 12: Using the NEW command

a. To create a new file in your workspace, use the NEW command. The format of this command is: NEW FILE = file name

```
NEW FILE=SAMPLEFILE3
```

b. In the NEW, OLD, SAVE, REPLACE, and NAME command examples, you can specify any of the following for filename:

   a. No file (for the SAVE and REPLACE commands)
      IPF 1100 assumes the name found in the system variable $FILENAME
   b. Just a file (for example, XYZ)
      IPF 1100 assumes your current work directory as the corresponding directory
   c. A fully qualified filename (such as,directory.file)
      (for example, ABC*DEF)
   d. If you are using data files (for example, ABC*DATA.), you cannot use the SAVE command.

c. After entering a NEW command, you can use EDIT 1100 commands to insert data into the workspace and manipulate it

   1. Set the system variable $DISPLAY, which you set to NUMBER to display line numbers

      ```
      SET $DISPLAY := NUMBER
      ```

   2. After telling IPF 1100 you want line numbers displayed before each image, erase the contents of your workspace and name it EXAMPLE. IPF 1100 assumes the directory name is that of your Work directory

      ```
      NAME NEW_NAME=EXAMPLE
      ```

   3. Erase and change the name of your workspace again, this time to MDR*WORK

      ```
      NAME NEW_NAME=MDR*WORK
      ```

4. List the directory to see if the file has been created

   ```
   LIST MDR*WORK
   ```

5. Enter a NEW command without specifying a filename. IPF 1100 displays an error message.

   ```
   NEW
   ```

6. Enter an acceptable NEW command including a filename.

   ```
   NEW MDR*WORK.TEST
   ```

7. Enter the EDIT 1100 command NUMBER, which lets you enter data into your workspace one line at a time.

   ```
   NUMBER
   ```

   IPF 1100 assigns a line number to each image you enter into your workspace, beginning at 10 and increasing the line number each time by 10

8. Enter the command *END.

   EDIT 1100 stops the automatic line numbering and input solicitation. IPF 1100 returns the cursor, waiting for your next command

Exercise 13: Using the SAVE command

Using a SAVE command does not change the name or contents of the workspace.Here are some examples of the SAVE command.

a. Create a new directory MDR*WORK2

```
CREA MDR*WORK2.
```

b. Tell IPF 1100 to display line numbers

```
SET $DISPLAY := NUMBER
```

c. Enter a NEW command, erasing the workspace and renaming it

```
NEW FILE=MDR*WORK2.TEST1
```

d. Enter an EDIT 1100 NUMBER command, which solicits lines of input and assigns line numbers to them starting at 0010.00 and incrementing each one by 10

```
NUMBER
```

e. Enter four lines of input, then a line with *END to terminate the automatic line numbering and input solicitation

```
LINE 1
LINE 2
LINE 3
LINE 4
*END
```

f. Enter a SAVE command without specifying a filename. IPF 1100 saves the contents of the workspace to a file named TEST1 of directory MDR*WORK2.

g. Verify that the file has been saved with the LIST command

```
LIST MDR*WORK2.
```

h. Enter another SAVE command, this time specifying a filename of MDR*WORK2.TEST2. IPF 1100 saves another copy of the workspace to a file named TEST2 in directory MDR*WORK2

```
SAVE FILE=MDR*WORK2.TEST2
LIST MDR*WORK2
```

Exercise 14: Manage Files and Directories

1. Working with Directories and Files

   You can save files that have some logical relation to each other within a directory. Each file on your system has a unique name to distinguish it from all other files

   Filenames can have these components: NAME*COLOR.BLUE

   where:

   NAME Is the qualifier,which is followed by an asterisk. If you omit the qualifier, IPF 1100 uses your project-id.

   COLOR. Is the name of a directory (or data file), which is followed by a period.

   BLUE Is the name of a file (unless you are using a data file).

   The expanded filename format is

   qualifierdirectory-name(cycle)/read-key/write-key.file/version-name

   a. Read and write keys may contain up to six characters, not including blank, comma, slash, period, or semicolon

      Keys are optional and specify whether the file is a read- or write-only file

   b. Using cycles - The cycle may be an integer in the range -31 to +1 (relative) and 0 to 999 (absolute)

      For example:

      CREATE COLOR(1).

      CREATE COLOR(2).

      CREATE COLOR(3).

      ATTACH COLOR. @attaches cycle 3 to your run

c. Use version-names , the version-name is from 1 to 12 characters from the set A-Z, 0-9, hyphen, or $

   For example:

```
GJH*DIRECNAME.FILENAME/VER-ONE
GJH*DIRECNAME.FILENAME/VER-TWO
GJH*DIRECNAME.FILENAME/VER-THREE
GJH*DIRECNAME.FILENAME/VER-FOUR
GJH*DIRECNAME.FILENAME/VER-FIVE
```

Steps:

a. View the help for the CREATE command

```
HELP CHOICE=COMMANDS LETTERS=CREATE
Enter 1
```

b. Create a file with no options or parameters

```
CREATE IPFFILE1
```

c. Verify that the file is created

```
LIST
```

d. View the content of the file

```
OLD IPFFILE1
PRINT ALL
```

   Since the file was newly created, a message will be displayed indicating that the workspace is empty.

e. Add a few images (lines) to the file using the line editor

```
NUMBER
Line 1
Line 2
*END
```

f. Save the file

2. Using datafiles

   Example : MY*DATAFILE

   Here MY*DATAFILE. is a data file

   A data file always has a filename ending with a period with nothing following it

Steps:

a. Create a data file in the MDR*WORK2 directory
   ```
   CREA MDR*WORK2.DATAFILE.
   ```
b. Save the file
   ```
   REPLACE
   ```
c. Verify that the file is created
   ```
   LIST MDR*WORK2.
   ```

3. Specify your home directory when you call IPF 1100 by using the following command: >@IPF home-directory

   Example : >@IPF JLK*ACC

Steps:

a. EXIT out of the current ipf session
   ```
   LOGOFF
   ```
b. Enter a new session with home directory set to MDR*WORK2
   ```
   @IPF MDR*WORK2.
   ```

4. IPF 1100 automatically searches your work directory if you specify a filename without a directory name on commands, such as :

   ```
   OLD          ERROR
   NEW          OUT
   SAVE          IN
   REPLACE      LIST
   ```

COPY	NAME
CREATE	PURGE
DESCRIBE	

Steps:

a. List the contents of the current home directory
   LIST
   You should see only the contents of the MDR*WORK2 directory
b. Logoff and reenter the IPF window with the default home directory
   LOGOFF
   @IPF

5. Use the COPY command to copy data from one file or directory to another file or directory on your system

   The format of the COPY command is

   COPY FROM = filename-1 TO= filename-2

Steps:

a. Copy the ipf file created in the earlier exercise
   COPY FROM=IPFFILE1 TO=IPFFILE2
b. Verify that the copy is created
   LIST

6. We can use several other IPF 1100 commands that you can use to perform file maintenance on data files and directories
   a. Use the PACK command to remove the old files from a directory, use the PACK command
      PACK FILE=directory-name-list
      PACK FILE=MDR*WORK

b. Use ERASE command to completely remove all the files from a directory or to erase a data file

   Format is :

   ```
   ERASE FILE= filename-list
   ERASE FILE=MDR*WORK.
   ```

7. Rather than type a long filename every time you have to specify the file, you enter a USE command to associate your own choice of a name (or identifier) with the actual filename

   The format of the USE command is

   ```
   USE NAME=identifier       FILE=filename
       USE NAME=ALIAS1. FILE=IPFFILE1.
       OLD ALIAS1
       PRINT ALL
   ```

8. IPF 1100 usually assigns and frees files as it needs them

   For example, when you tell IPF 1100 to list the files in a directory, IPF 1100 takes care of assigning the directory. Assigning means that IPF 1100 can now access the file

   The ATTACH command provides this function

   Its format is

   ```
   ATTACH FILE= filename
   ```

Steps:

a. Create a new diretory MDR*WORK3.

   ```
   CREATE FILE=MDR*WORK3.
   ```

b. Attach the newly created directory to the run

   ```
   ATTACH FILE=MDR*WORK3.
   ```

9. Sometimes you may need to explicitly free the assignment of a file (for example, someone else may want to use it)

   Use the FREE command to do this

   Its format is

   FREE FILE=filename-list

   FREE FILE=MDR*WORK3.

10. Use DESCRIBE command - tells IPF 1100 to display information on your terminal about the directories or data files you specify

    The format of the DESCRIBE command is

    DESCRIBE FILE = filename-list

    The filename-list can contain directory names, data filenames, or both

    DESCRIBE FILE=MDR*WORK2.

Exercise 15: Using editing commands in screen mode

1. Full-Screen Display
   a. After you enter MODE FORMAT=SCREEN EDIT 1100 switches to full-screen mode
   b. The top two lines are the Command/Response region
   c. The third line is the status line
   d. The fourth line is the scale line
   e. The Screen Command region is a reserved field, comprising the second through fifth column of each line on the screen except, of course, for the top four lines of the screen
   f. The remainder of the screen shows data images in your workspace
   g. If EDIT 1100 reaches the end of your workspace, it displays the Message
      *END-FILE mode-type/first-line-#-on-screen/last-line-#-on-screen

2. Using full screen commands
   a. Use the first character of M for move, I for insert,..for editing Full-screen
   b. The remaining three columns in the screen command region specify a relative number of screen lines on which to perform the command
   c. Use the TAB key on your terminal to quickly move the cursor to the screen command and data image regions

3. How to Just Browse
   a. If you only want to look at data, you can specify BROWSE mode
   b. Enter the command:

MODE FORMAT=SCREEN TYPE=BROWSE

c. If you enter a general-editing command while in BROWSE mode, EDIT 1100 displays on the Response region the error message:

    **EDIT1400 You are not allowed to update in BROWSE mode.

d. You can also go into BROWSE mode by entering SET $READONLY:=TRUE

4. Moving a data image in your workspace
    a. Type M in the Screen Command region for the image you want to move
    b. Then you type MA in the Screen Command region of the data image where you want the image you are moving to be placed after
    c. Instead of specifying the image after which you want to place the image you are moving, you can specify the image before which you want to place the image you are moving
    d. Type the command MB in the screen command region of the image before which you want the images you are moving to be placed

5. To move more than one data image at a time (up to 999 data images)
    a. Type Mn in front of the first of a set of contiguous data images you want to move
    b. EDIT 1100 moves the data image you mark, and the next n-1 data images,to the destination you specify by the MA or MB command

6. Mark the start and end of a contiguous set of data images

a. Instead of counting how many contiguous images you want EDIT 1100 to move, you can just mark the start and end of a contiguous set of data images

b. Type MS and ME, respectively, in front of the first and the last of a contiguous set of images

7. Navigate forward or backward

   a. Use Relative positioning with the GO command.

   b. Type G or GO in the screen command region of an image to scroll to that position.

   c. Type G-2 to navigate backward by 2 images.

8. Copy images

   d. Enter C in the screen command region of the image you want to copy

   e. Use CA or CB to copy the image after or before a particular image

9. Delete images in your workspace using the delete command

   f. Type D in the Screen Command region of each screen line you want EDIT 1100 to delete

   g. EDIT 1100 deletes the images in your workspace that you mark with the D

10. To delete a number of contiguous lines

   a. Instead of marking each image in your workspace that you want to delete, you can tell EDIT 1100 to delete a number of contiguous lines

   a. Type the full-screen command Dn in the Screen Command region of the first of the set of contiguous lines you want to delete

b. This means delete n images beginning with the marked line (that is, delete this line and the n - 1 following images)
c. An alternate way to mark a set of contiguous images for deletion is to use the full-screen mode DS and DE commands
d. You type DS in the Screen Command region of the first image you want to delete, and type DE in the Screen Command region of the last of a set of contiguous images you want to delete

11. Enter INPUT mode
    a. Use N to enter INPUT mode inserting a line starting after the current image
    b. Use NB to insert a line before the current image.

12. Access specific images using form lines
    a. n - Image with line number n
    b. $C - Current image
    c. $T - Top or first image
    d. $B - Bottom or last image
    e. %var - Image with line number as per value of var
    You can use the above with commands such as GO

Exercise 16: Running Programs

1. Use the editing commands of EDIT 1100
    a. Use a NEW command to create the program with the name PROG1
    b. Add the following code using the editor in screen mode. Start from the 8th column

```
IDENTIFICATION DIVISION.
PROGRAM-ID. COB01.
ENVIRONMENT DIVISION.

DATA DIVISION.
WORKING-STORAGE SECTION.

01 WS-HELLO            PIC X(12).

PROCEDURE DIVISION.

MAIN-PARA.
        MOVE 'HELLO' TO WS-HELLO.
        DISPLAY WS-HELLO.
        STOP RUN.
```

    c. Enter your program in your workspace (using EDIT 1100)
    d. Use a TYPE command to tell IPF 1100 lang subtype

```
TYPE LANGUAGE=COB
```

    e. Enter a SAVE command
    f. Enter a RUN command