Computer Labs for the

Introduction to the New Mainframe: z/OS Basics course

Instructor Version

The following labs are from the Introduction to the New Mainframe: z/OS Basics course/Redbook. These labs have been run and tested using the computer hub at Marist College in Poughkeepise, NY. You may need to adjust these labs to run on a computer system other than the one at Marist College.

Chapter 2 Exercises (Section 2.16)

To display the CPU configuration

1. Access SDSF from the ISPF primary option menu.
2. In the command input field, enter /D M=CPU and press Enter.
3. Use the ULOG option in SDSF to view the command display result.

To display the page data set usage

1. In the command input field, enter /D ASM and press Enter.
2. Press PF3 to return to the previous screens.

To display information about the current Initial Program Load (IPL)

1. Use ULOG option in SDSF to view the command display result.
2. In the command input field, enter /D IPLINFO and press Enter.

Instructor Notes

1. Display the shared consoles on several systems by entering the command D C on those systems.
2. Use the ROUTE command to send a command to one other system, and then to all the systems, for example RO *OTHER, D IPLINFO RO *ALL, D IPLINFO.
3. Show that WLM has only one policy active across the sysplex: D WLM.
4. If a FLEX-ES system is used for class work, view and explain its configuration and usage.

Chapter 4 Exercises (Section 4.7)

These lab exercises will help you develop skills in using TSO, ISPF and the z/OS UNIX command shell. These skills are required for performing lab exercises in the remainder of this text. To perform the lab exercises, each student or team needs a TSO user ID and password (for assistance, see the instructor).

4.7.1 Logging on to z/OS and entering TSO commands

Establish a 3270 connection with z/OS using a workstation 3270 emulator and log on with your user ID (we will call this *yourid*). From the TSO READY prompt (after you have keyed in **=x** to exit out of ISPF into native TSO), enter the following commands:

PROFILE
What is the prefix value? Make a note of this; it is your user ID on the system.

PROFILE NOPREFIX
This changes your profile so TSO will not place a prefix at the beginning of your commands. Specifying PROFILE PREFIX (with a value) or NOPREFIX (by itself) tells the system whether to use a value (such as your user ID) to find files in the system. NOPREFIX tells the system not to bother limiting the results to files beginning with your user ID (for example) as it would otherwise do by default.

LISTC
The LISTCAT command (or LISTC, for short) lists the data sets in a particular catalog. Your 3270 emulator has a PA1 (attention) key. You can use the PA1 key to end the command output.
Note: When you see the three asterisks (***), it indicates that your screen is filled. Press Enter or PA to continue.

PROFILE PREFIX(userid)
This command specifies that your user ID is to be prefixed to all non-fully-qualified data set names. This will filter the results of the next command:

LISTC
What is displayed?

ISPF (or ISPPDF)
Enter into the ISPF menu-driven interface of TSO. **Note:** On some systems, you will also need to select option P to access the main ISPF screen.

4.7.2 Navigating through the ISPF menu options

From the ISPF Primary Option Menu, do the following:

1. Select **Utilities**, then select **Dslist** from the Utility Selection Panel.

2. Enter SYS1 on the Dsname Level input field and press Enter. What is displayed?
3. Use F8 to page down or forward, F7 to page up or backward, F10 to shift left and F11 to shift right. Exit with F3.
4. Enter SYS1.PROCLIB on the Dsname Level input field and press Enter. What is displayed?
5. Enter v in the command column to the left of SYS1.PROCLIB. This is a partitioned data set with numerous members. Place an s to the left of any member to select the member for viewing. Press F1. What specific help is provided?
6. Enter =0 on the ISPF command or option line. What is the first option listed in this ISPF Settings panel? Change your settings to place the command line at the bottom of the panel. It is effective on exit from the Settings panel.
7. Enter PFSHOW OFF and then PFSHOW ON. What is the difference? How is this useful?
8. Exit back to the ISPF Primary Option Menu. What value is used to select Utilities?
9. Select **Utilities**.
10. In the Utilities Selection Panel, what value is used to select Dslist? Exit back to the ISPF Primary Option Menu. On the option line, enter the Utilities selection value followed by a period, then enter the Dslist selection value. What panel is displayed?
11. Exit back to the ISPF Primary Option Menu. Place the cursor on the Status entry at the very top of the panel and press Enter. Select the Calendar value and press Enter, then select the Session value. What changed?
12. Now set your screen to the original configuration, using the Status pull-down and selecting **Session**.

4.7.3 Using the ISPF editor

From the ISPF Primary Option Menu, do the following:

1. Go to the DSLIST utility Panel and enter *yourid***.**JCL in the Dsname Level field. Press Enter.
2. Place e (edit) to the left of *yourid***.**JCL. Place s (select) to the left of member EDITTEST. Enter PROFILE on the edit command line, observe the data is preceded by profile and message lines. Read the profile settings and messages, then enter RESET on the command line. What is the result?
3. Enter any string of characters at the end of the first data line, then press Enter. On the command line, enter CAN (cancel). Press Enter to confirm the cancel request. Again, edit EDITTEST in the data set. Were your changes saved?

   Tip: As you become more familiar with ISPF, you will learn the letters and numbers for some of the commonly used options. Preceding an option with the = key takes you directly to that option, bypassing the menus in between.

   You can also go directly to nested options with the = sign. For example, =3.4 takes you directly to a commonly used data set utility menu.

4. Move the cursor to one of the top lines on your display. Press F2. The result is a second ISPF panel. What occurs when F9 is entered repeatedly?
5. Using F9, switch to the ISPF Primary Option Menu, then press F1 to display the ISPF Tutorial panel.
6. From the ISPF Tutorial panel, select **Edit**, then **Edit Line Commands**, then **Basic Commands**. Press Enter to scroll through the basic commands tutorial. As you do so, frequently switch (F9) to the edit session and exercise the commands in EDITTEST. Repeat this same scenario for Move/Copy commands and shifting commands.
7. From the ISPF Tutorial panel, select **Edit**, then **Edit Primary Commands**, then **FIND/CHANGE/EXCLUDE commands**. Press Enter to scroll through the FIND/CHANGE/EXCLUDE commands tutorial. As you do so, frequently switch (F9) to the edit session and exercise the commands in EDITTEST.
8. Enter =X on the ISPF help panel to end the second ISPF panel session. Save and exit the Edit Panel (F3) to return to the ISPF Primary Option Menu.

4.7.4 Using SDSF

From the ISPF Primary Option Menu, locate and select **System Display and Search Facility (SDSF)**, which is a utility that lets you look at output data sets. Select **More** to find the SDSF option (5), or simply enter =M.5. The ISPF Primary Option Menu typically includes more selections than those listed on the first panel, with instructions on how to display the additional selections.

1. Enter LOG, then shift left (F10), shift right (F11), page up (F7) and page down (F8). Enter TOP, then BOTTOM on the command input line. Enter DOWN 500 and UP 500 on the command input line. You will learn how to read this system log later.
2. Observe the SCROLL value to the far left on the command input line.

   Scroll ===> PAGE

   Tab to the SCROLL value. The values for SCROLL can be:

C or CSR	Scroll to where you placed the cursor
P or PAGE	Full page or screen
H or HALF	Half page or half screen

3. You will find the SCROLL value on many ISPF panels, including the editor. You can change this value by entering the first letter of the scroll mode over the first letter of the current value. Change the value to CSR, place the cursor on another line in the body of the system log, and press F7. Did it place the line with the cursor at the top?
4. Enter ST (status) on the SDSF command input line, then SET DISPLAY ON. Observe the values for Prefix, Best, Owner, and Susanne. To display all of the current values for each, enter * as a filter, for example:

   ```
   PREFIX *
   OWNER *
   DEST
   ```

   The result should be:

   ```
   PREFIX=* DEST=(ALL) OWNER=*
   ```

5. Enter DA, to display all active jobs. Enter ST to retrieve the status of all jobs in the input, active, and output queues. Once again, press F7 (page up), F8 (page down), F10 (shift left), and F11 (shift right).

4.7.5 Opening the z/OS UNIX shell and entering commands

From the ISPF Primary Option Menu, select Option 6, then enter the OMVS command. From your home directory, enter the following shell commands:

id	Shows your current id.
date	Shows time and date.
man date	Manual of the **date** command. You can scroll through the panels by pressing Enter. Enter quit to exit the panels.
man man	Help for the manual.
env	Environment variables for this session.
type read	Identifies whether read is a command, a utility, an alias, and so forth.
ls	List a directory.
ls -l	List the current directory.
ls -l /etc.	List the directory /etc.
cal	Display a calender of the current month.
cal 2005	Display a calender of the year 2005.
cal 1752	13 days? [Answer: Yes, all UNIX calendars have 13 days missing from September 1752.] Optional: To find out why, ask a History major!
exit	End the OMVS session.

4.7.6 Using the OEDIT and OBROWSE commands

Another way to start the OMVS shell is by entering the TSO OMVS command on any ISPF panel. From your home directory, enter the following shell commands:

cd /tmp	This is a directory that you have authority to update.
oedit myfile	This opens the ISPF edit panel and creates a new text file in the current path. Write some text into the editor. Save and exit (F3).
ls	Display the current directory listing in terse mode.
ls -l	Display the current directory listing in verbose mode.

myfile	*myfile* can be any file you choose to create.
obrowse myfile	Browse the file you just created.
exit	End the OMVS session.

Instructor Notes

For users of the FLEX-ES emulator system, there is a key-mapping panel for the functions shown in Table 4-1. To access it, click on the keyboard icon in the upper right of your FLEX-ES user screen.

Some possible classroom demonstrations follow:

- Demonstrate that on the TSO logon screen, you can place an S in front of the RECONNECT option at the bottom of the screen. This allows a user who has lost a connection to a TSO session to regain that connection and avoid having to ask the operator to cancel that running session. The S can be saved there by default.
- z/OS central storage is divided into frames of 4K each, enough to hold one page. When active address spaces need more central storage, z/OS pages out infrequently used storage frames to make them available for use. z/OS uses paging data sets allocated on DASD containing 4K slots of storage to hold these paged-out frames temporarily. In z/OS, the pageable link pack area (PLPA) contains programs that can be paged out. To check the paging activity of your system, enter the following commands:
  ```
  D ASM
  D ASM,PLPA
  ```
  – How many slots are allocated to hold paged-out PLPA frames?
  – How many have been used?
  – What is the name of the paging data set?
  – On which direct access storage device (DASD) volume does it reside?

z/OS can also page out the common storage area (CSA). To see this for yourself, enter the following command:

```
D ASM,COMMON
```

Refer to Figure 4-10 "SORT panel"

- Use this as a demonstration and proceed through the panels with the students.
- In Example 4-4, we must use TSO line commands as we use the panels.
- Use as SORTIN input data sets ZPROF.AREA.CODES.
- Use as SYSIN sort control data sets ZPROF.SORT.CNTL.

Refer to 4.4, “z/OS UNIX interactive interfaces”

The REXX support for z/OS UNIX is not really an interactive interface, but we chose to introduce it here because it is most often used in TSO or in the shell.
The SYSCALL environment is not built into TSO/E, but an external function call called SYSCALLS will initialize the environment. Note that the shell is the initial
host environment, which means that the SYSCALL environment is automatically initialized. A difference between the REXX support and shell scripts is that a REXX EXEC can be invoked from a C program, while a shell script can only be interpreted from the shell. A REXX EXEC can be called from a shell script.

Refer to 4.7.5, “Opening the z/OS UNIX shell and entering commands”

Entering the command **cal 1752** from the z/OS UNIX shell might yield a surprising result, unless you happen to be a calendar expert. September of that year seems to be missing 11 days, but it’s not a bug in UNIX. In 1752, the month of September was officially shortened to 19 days in some parts of the world to allow for a change from the Julian calendar to the Gregorian calendar. So, no bug—UNIX is just doing its homework!

Chapter 5 Exercises (Section 5.17)

These lab exercises help you develop skills in working with data sets using ISPF. To perform the lab exercises, you or your team require a TSO user ID and password (for assistance, see the instructor).

Tip: The 3270 Enter key and the PC Enter key can be confused with each other. Most 3270 emulators permit the user to assign these functions to any key on the keyboard, and we assume that the 3270 Enter function is assigned to the right-hand CTRL key. Some z/OS users, however, prefer to have the large PC Enter key perform the 3270 Enter function and have Shift-Enter (or the numeric Enter key) perform the 3270 New Line function.

5.17.1 Exploring ISPF Option 3.4

One of the most useful ISPF panels is Option 3.4. This terminology means, starting from the ISPF primary option menu, select Option 3 (Utilities) and then Option 4 (Dslist, for data set list). This sequence can be abbreviated by entering
3.4 in the primary menu, or =3.4 from any panel.

Many ISPF users work almost exclusively within the 3.4 panels. We cover some of the 3.4 functions here and others in subsequent exercises in this text. Use care in working with 3.4 options; they can effect changes on an individual or system-wide basis.

z/OS users typically use ISPF Option 3.4 to check the data sets on a DASD volume or examine the characteristics of a particular data set. Users might need to know:

- What data sets are on this volume?
- How many different data set types are on the volume?
- What are the DCB characteristics of a particular file?

Let's answer these questions using WORK02 as a sample volume, or another volume as specified by your instructor:

1. In the 3.4 panel, enter WORK02 in the Volume Serial field. Do not enter anything on the Option==> line or in the Dsname Level field.
2. Use PF8 and PF7 to scroll through the data set list that is produced.
3. Use PF11 and PF10 to scroll sideways to display more information. This is not really scrolling in this case; the additional information is obtained only when PF11 or PF10 is used.

   The first PF11 display provides tracks, percent used, XT, and device type. The XT value is the number of extents used to obtain the total tracks shown. The ISPF utility functions can determine the amount of space actually used for some data sets and this is shown as a percentage when possible.

The next PF11 display shows the DCB characteristics: DSORG, RECFM, LRECL, and BLKSIZE.

PS	Sequential data set (QSAM, BSAM)
PO	Partitioned data set
VS	VSAM data set
blank	Unknown organization (or no data exists)

RECFM, LRECL, and BLKSIZE should be familiar. In some cases, usually when a standard access method is not used or when no data has been written, these parameters cannot be determined. VSAM data sets have no direct equivalent for these parameters and are shown as question marks.

Look at another volume for which a larger range of characteristics can be observed. The instructor can supply volume serial numbers. Another way to find such a volume is to use option 3.2 to find where SYS1.PARMLIB resides, then examine that volume.

5.17.2 Allocating a data set with ISPF 3.2

ISPF provides a convenient method for allocating data sets. In this exercise, you create a new library that you can use later in the course for storing program source data. The new data sets should be placed on the *WORK02* volume and should be named *yourid*.LIB.SOURCE (where *yourid* is your student user ID).

For this exercise, assume that 10 tracks of primary space and 5 tracks for secondary extents is sufficient, and that 10 directory blocks is sufficient. Furthermore, we know we want to store 80-byte fixed-length records in the library. We can do this as follows:

1. Start at the ISPF primary menu.
2. Go to option 3.2, or go to option 3 (Utilities) and then go to option 2 (Data Set).
3. Type the letter A in the Option ==> field, but do not press Enter yet.
4. Type the name of the new data set in the Data Set Name field, but do not press Enter yet. The name can be with single quotes (for example, '*yourid*.LIB.SOURCE') or without quotes (LIB.SOURCE) so that TSO/ISPF automatically uses the current TSO user ID as the HLQ.
5. Enter *WORK02* in the Volume Serial field and press Enter.
6. Complete the indicated fields and press Enter:
    - Space Units = TRKS
    - Primary quantity = 10
    - Secondary quantity = 5
    - Directory blocks = 10

- Record format = FB
- Record length = 80
- Block size = 0 (this tells z/OS to select an optimum value)
- Data set type = PDS

This should allocate a new PDS on *WORK02*. Check the upper right corner, where the following message appears:

```
Menu RefList Utilities Help
-----------------------------------------------------------------
-
Data Set Utility Data set allocated

Option ===>
A Allocate new data set C Catalog data set
.....
```

5.17.3 Copying a source library

A number of source programs are needed for exercises in *ZPROF*.ZSCHOLAR.LIB.SOURCE on WORK02. There are several ways to copy data sets (including libraries). We can use the following:

1. Go to ISPF option 3.3 (Utilities, Move/Copy).
2. On the first panel:
   a. Type C in the Option==> field.
   b. Type '*ZPROF*.ZSCHOLAR.LIB.SOURCE' in the Data Set Name field. The single quotes are needed in this case.
   c. The Volume Serial is not needed because the data set is cataloged.
   d. Press Enter.
3. On the second panel:
   a. Type '*yourid*.LIB.SOURCE' in the Data Set Name field and press Enter. If this PDS does not exist, type 1 to inherit the attributes of the source library. This should produce a panel listing all the members in the input library:
   b. Type S before every member name and then press Enter.

   This copies all the indicated members from the source library to the target library. We could have specified '*ZPROF*.ZSCHOLAR.LIB.SOURCE(*)' for the input data set; this would automatically copy all the members. This is one of the few cases where wild cards are used with z/OS data set names.

4. Create another library and move several members from LIB.SOURCE into the new library. Call it '*yourid*.MOVE.SOURCE'. Verify that the moved members are in the new library and no longer in the old one. Copy those members back into the LIB library. Verify that they exist in both libraries.
5. Rename a member in the MOVE library. Rename the MOVE library to '*yourid*.TEST.SOURCE'.

5.17.4 Working with data set members

There are several ways to add a new member to a library. We want to create a new member named TEST2 to your library that we previously edited:

1. From the ISPF primary menu, use option 2.
2. Enter the name of your library without specifying a member name, for example *yourid*.JCL. This provides a list of member names already in the library.
3. Verify that member EDITTEST has the same contents you used earlier:
    a. If necessary, scroll so you can see member name EDITTEST.
    b. Move the cursor to the left of this line.
    c. Type S and press Enter.
    d. Look at your earlier work to assure yourself it is unchanged.
    e. Press PF3 to exit ("back out of") member EDITTEST. You will see the library member name list again.
4. Enter S TEST2 on the command line at the top of the screen and press Enter. (S TEST2 can be read as "select TEST2.") This creates member TEST2 and places the screen in input mode.
5. Enter a few lines of anything, using the commands and functions we discussed earlier.
6. Press PF3 to save TEST2 and exit from it.
7. Press PF3 again to exit from the ISPF Edit function.

Hereafter we will simply say "Enter xxx" when editing something or using other ISPF functions. This means (1) type xxx, and (2) press the Enter key. The New Line key (which has Enter printed on it) is used only to position the cursor on the screen.

5.18 Listing a data set and other ISPF 3.4 options

Go to the ISPF 3.4 panel. Enter *yourid* in the Dsname Level field and press Enter. This should list all the cataloged data sets in the system with the indicated HLQ. An alternative is to leave the Dsname Level field blank and enter WORK02 in the Volume Serial field; this lists all the data sets on the indicated volume. (If both fields are used, the list will contain only the cataloged data sets with a matching HLQ that appear on the specified volume.)

A number of functions can be invoked by entering the appropriate letter before a data set name. For example, position the cursor before one of the data set names and press PF1 (Help). The Help panel lists all the line commands that can be used from the data set name list of the 2.4 panel. Do not experiment with these without understanding their functions. Not all of these functions are relevant to this class. The relevant commands are:

E	Edit the data set.
B	Browse the data set.
D	Delete the data set.
R	Rename the data set.
Z	Compress a PDS library to recover lost space.
C	Catalog the data set.
U	Uncatalog the data set.

When a member list is displayed (as when a library is edited or browsed) several line commands are available:

S	Select this member for editing or browsing.
R	Rename the member.
D	Delete the member.

5.18.1 Performing a catalog search

The ISPF 3.4 option can be used for catalog searches on partial names. Use PF1 Help to learn more about this important function, as follows:

1. Select option 3.4.
2. Press PF1 for help and select **Display a data set list**. Press Enter to scroll through the information panels.
3. Then select **Specifying the DSNAME LEVEL**. Press Enter to scroll through the information panels.
4. Press PF3 to exit from the Help function.

Notice that the 3.4 DSNAME LEVEL field does not use quotes and the current TSO/E user ID is not automatically used as a prefix for names in this field. This is one of the few exceptions to the general rule for specifying data set names in TSO.

Instructor Notes

If you are using the FLEX-ES emulation system for this course, the system might have volumes Z6SYS1 and jZ6CIC1, which you can use to show students a large range of volume characteristics.

Chapter 6 Exercises (Section 6.14)

These lab exercises help you develop skills in creating batch jobs and submitting them for execution on z/OS. To perform the lab exercises, you or your team requires a TSO user ID and password (for assistance, see the instructor).

6.14.1 Creating a simple job

1. From ISPF, navigate to the Data Set List Utility panel and enter *yourid*.JCL in the Dsname Level field (described in an earlier exercise).
2. Enter e (edit) to the left (in the command column) of *yourid*.JCL. Enter s (select) to the left of member JCLTEST. Enter RESet on the editor command line.
3. Notice that only a single JCL line is in the data set, EXEC PGM=IEFBR14. This is a system utility that does not request any input or output and is designed to complete with a successful return code (0). Enter SUBMIT or SUB on the command line and press Enter.
4. Enter 1 in response to the message:

    IKJ56700A ENTER JOBNAME CHARACTER(S)

   The result will be the message:

    IKJ56250I JOB *yourid*1(JOB00037) SUBMITTED

   Note: Whenever you see three asterisks (***), it means there's more data to see. Press Enter to continue.

   When the job finishes, you should see the message:

    $HASP165 yourid1 ENDED AT SYS1 MAXCC=0 CN(INTERNAL)
5. Add (insert) a new first line in your file that will hold a JOB statement. The JOB statement must precede the EXEC statement. (Hint: Replicate (r) the single EXEC statement, then overwrite the EXEC statement with your JOB statement.) This JOB statement should read:

    //*yourid*A JOB 1

   Replace "yourid" with your team user ID, leave the "A", then submit this JCL and press PF3 to save the file and exit the editor.
6. From the ISPF Primary Option Menu, find SDSF (described in 7.9.5, "Using SDSF"). You can use the split screen function for a new screen session, giving you one session for the DSLIST and the other for SDSF.
7. In the SDSF menu, enter PREFIX yourid*, then enter ST (Status Panel). Both jobs that you submitted should be listed. Place S (select) to the left of either job, then page up and down to view the messages produced from the execution. Press PF3 to exit.

8. Edit JCLTEST again, and insert the following lines at the bottom:

```
//CREATE DD DSN=yourid.MYTEST,DISP=(NEW,CATLG),
// UNIT=SYSDA,SPACE=(TRK,1)
```

9. Submit the content of JCLTEST created above, press PF3 (save and exit edit), then view the output of this job using SDSF. Notice that you have two jobs with the same jobname. The jobname with the highest JOBID number is the last one that was run.
    a. What was the condition code? If it was greater than 0, page down to the bottom of the output listing to locate the JCL error message. Correct the JCLTEST and resubmit. Repeat until cond code=0000 is received.
    b. Navigate to the Data Set List Utility panel (=3.4) and enter *yourid*.MYTEST in the DSNAME level field. What volume was used to store the data set?
    c. Enter DEL / in the numbered left (command) column of the data set to delete the data set. A confirmation message may appear asking you to confirm that you want to delete the data set.
    d. We just learned that batch execution of program IEFBR14, which requires no inputs or outputs, returns a condition code 0 (success) if there were no JCL errors. Although IEFBR14 does no I/O, JCL instructions are read and executed by the system. This program is useful for creating (DISP=NEW) and deleting (DISP=(OLD,DELETE)) data sets on a DD statement.

10. From any ISPF panel, enter in the Command Field ==>

    TSO SUBMIT JCL(JCLERROR)

    Your user ID is the prefix (high-level qualifier) of data set JCL containing member JCLERROR.
    a. You will be prompted to enter a suffix character for a generated job card. Take note of the jobname and job number from the submit messages.
    b. Use SDSF and select the job output. Page down to the bottom. Do you see the JCL error? What are the incorrect and correct JCL DD operands? Correct the JCL error located in *yourid*.JCL(JCLERROR). Resubmit JCLERROR to validate your correction.

11. From any ISPF panel, enter TSO SUBMIT JCL(SORT). Your user ID is the assumed prefix of data set JCL containing member SORT.
    a. You will be prompted to enter a suffix character for a generated job card. Take note of the jobname and job number from the submit messages.
    b. Use SDSF and place a ? to the left of the job name. The individual listing from the job will be displayed. Place s (select) to the left of SORTOUT to view the sort output, then press PF3 to return. Select JESJCL. Notice the "job statement generated message" and the "substitution JCL" messages.

12. Let's purge some (or all) unnecessary job output. From SDSF, place a p (purge) to the left of any job that you would like to purge from the JES output queue.

13. From the ISPF panel, enter TSO SUBMIT JCL(SORT) and review the output.

14. From the ISPF panel, enter TSO SUBMIT JCL(SORTPROC) and review the output. You may not see the output in the SDSF ST panel. This is because the jobname is not starting with *yourid*. To see all output, enter PRE *, then OWNER yourid to see only the jobs that are owned by you.

15. What JCL differences exist between SORT and SORTPROC? In both JCL streams, the SYSIN DD statement references the sort control statement. Where is the sort control statement located?

> **Tip:** All JCL references to &SYSUID are replaced with the user ID that submitted the job.

16. Edit the partitioned data set member containing the SORT control statement. Change FIELD=(1,3,CH,A) to FIELD=(6,20,CH,A). Press PF3 and then from the ISPF panel enter TSO SUBMIT JCL(SORT). Review the job's output using SDSF. Was this sorted by code or area?
17. From the ISPF panel, enter TSO LISTC ALL. By default, this will list all catalog entries for data sets beginning with yourid. The system catalog will return the data set names, the name of the catalog storing the detailed information, the volume location, and a devtype number that equates to specific values for JCL UNIT= operand. LISTC is an abbreviation for LISTCAT.

6.14.2 Using ISPF in split screen mode

As discussed earlier, most ISPF users favor a split screen. This is easily done:

1. Move the cursor to the bottom (or top) line.
2. Press PF2 to split the screen.
3. Press PF9 to switch between the two screens.
4. Use PF3 (perhaps several times) to exit from one of the splits. The screen need not be split at the top or bottom. The split line can be positioned on any line by using PF2. More than two screens can be used. Try to use these ISPF commands:

   START
   SWAP LIST
   SWAP <screen number.>

6.14.3 Manipulating text in ISPF

After logging on to TSO/E and activating ISPF, look at the primary option menu.

1. Enter each option and write down its purpose and function. Each team should prepare a brief summary for one of the 12 functions on the ISPF panel (Items 0-11). Note that z/OS installations often heavily customize the ISPF panels to suit their needs.
2. Create a test member in a partitioned data set. Enter some lines of information, then experiment with the commands below. Use PF1 if you need help.

i	Insert a line.
Enter key	Press Enter without entering anything to escape insert mode.
i5	Obtain 5 input lines.

d	Delete a line.
d5	Delete 5 lines.
dd/dd	Delete a block of lines (place a DD on the first line of the block and another DD on the last line of the block).
r	Repeat (or replicate) a line.
rr/rr	Repeat (replicate) a block of lines (where an RR marks the first line of the block and another RR marks the last line).
c along with a or b	Copy a line after or before another line.
c5 along with a or b	Copy 5 lines after or before another line.
cc/cc along with a or b	Copy a block of lines after or before another line.
m, m5, mm/mm	Move line(s).
x, x5, xx/xx	Exclude lines.
s	Redisplay (show) the lines you excluded.
(	Shift right columns.
)	Shift left columns.
<	Shift left data.
>	Shift right data.

6.14.4 Submitting a job and checking the results

Edit member COBOL1 in the *yourid*.LIB.SOURCE library and inspect the COBOL program. There is no JCL with it. Now edit member COBOL1 in *yourid*.JCL.[1] Inspect the JCL carefully. It uses a JCL procedure to compile and run a COBOL program.[2] Follow these steps:

1. Change the job name to *yourid* plus additional characters.
2. Change the NOTIFY parameter to your user ID.
3. Add TYPRUN=SCAN to your job card.
4. Type SUB on the ISPF command line to submit the job.
5. Split your ISPF screen and go to SDSF on the new screen (you might have this already from an earlier exercise).
6. In SDSF go to the ST (Status) display and look for your job name.
   You may need to enter a PRE or OWNER command on the SDSF command line to see any job names. (A previous user may have issued a prefix command to see only certain job names.
7. Type S beside your job name to see all of the printed output:
   - Messages from JES2
   - Messages from the initiator
   - Messages from the COBOL compiler

[1] The matching member names (COBOL1) are not required; however, they are convenient.
[2] This is not exactly the COBOL procedure we discussed earlier. Details of these procedures sometimes change from release to release of the operating system.

    - Messages from the binder
    - Output from the COBOL program
8. Remove TYPRUN=SCAN when you are ready to run your job.
9. Use PF3 to “move up” a level and type ? beside your job name to display another output format.

The instructor can tell you the purposes of the various JES2 and initiator messages.

- Resubmit the job with MSGLEVEL=(1,1) on the JOB statement.
- Resubmit the job with MSGLEVEL=(0,0) on the JOB statement.

The MSGLEVEL parameter controls the number of initiator messages that are produced.

6.14.5 Creating a PDS member

There are several ways to create a new PDS member. Try each of the following, using your own user ID. In the following steps, TEST3, TEST4, TEST5, and TEST6 represent new member names. Enter a few lines of text in each case. Use the ISPF edit panel:

- Go to the ISPF primary menu.
- Go to option 2 (Edit).
- In the Data Set Name line, enter JCL(TEST3) (no quotes!)
- Enter a few text lines and use PF3 to save the new member.

A new member can be created while viewing the member list in edit mode:

- Use option 3.4 (or option 2) to edit *yourid*.JCL.
- While viewing the member list, enter S TEST4 in the command line.
- Enter a few text lines and use PF3 to save the new member.

A new member can be created while editing an existing member:

- Edit yourid.JCL(TEST1) or any other existing member.
- Select a block of lines by entering cc (in the line command area) in the first and last lines of the block.
- Enter CREATE TEST5 on the command line. This will create member TEST5 in the current library.

A new member can be created with JCL. Enter the following JCL in *yourid*.JCL(TEST5) or any other convenient location:

```
//yourid1 JOB 1,JOE,MSGCLASS=X
//STEP1 EXEC PGM=IEBGENER
//SYSIN DD DUMMY
//SYSPRINT DD SYSOUT=*
//SYSUT2 DD DISP=OLD,DSN= yourid.JCL(TEST6)
//SYSUT1 DD *
This is some text to put in the member
More text
/*
```

Save the member with this JCL. It will be used later.

6.14.6 Copying a PDS member

There are many ways to copy a library member. An earlier exercise used the ISPF 3.3 panel function to copy all the members of a library. The same function can be used to copy one or more members.

While editing a library member, we can copy another member of the library into it:

- Edit a library member.
- Mark a line in this member with a (after) or b (before) to indicate where the other member should be copied.
- Enter COPY xxx on the command line, where xxx is the name of another member in the current data set.

We can copy a member from another data set (or a sequential data set) as follows:

- Edit a member or sequential data set.
- Mark a line with A (after) or B (before) to indicate where to insert the new material.
- Enter COPY on the command line to display the Edit/View-Copy panel.
- Enter the full sequential data set name (with single quotes, if necessary) or a full library name (including member name) in the Data Set Name field.

Instructor Notes

If you are using the FLEX-ES emulation system for this course, in Step 2 (“Change the NOTIFY parameter to your user ID”), change the HLQ of IGY to IGY330.

Chapter 7 Exercises (Section 7.9)

7.9.1 Learning about system volumes

Use the ISPF functions to explore several system volumes. The following are of interest:

- Examine the naming of VSAM data sets. Note the words DATA and INDEX as the last qualifier.
- Find the spool area. This may involve a guess based on the data set name. How large is it?
- Find the basic system libraries, such as SYS1.PROCLIB and so forth. Look at the member names.
- Consider the ISPF statistics field that is displayed in a member list. How does it differ for source libraries and execution libraries?

7.9.2 Using a utility program in a job

z/OS has a utility program named IEBGENER to copy data. It uses four DD statements:

- SYSIN for control statements. We can code DD DUMMY for this statement, because we do not have any control statements for this job.
- SYSPRINT for messages from the program. Use SYSOUT=X for this lab.
- SYSUT1 for the input data.
- SYSUT2 for the output data.

The basic function of the program is to copy the data set pointed to by SYSUT1 to the data set pointed to by SYSUT2. Both must be sequential data sets or members of a library.

The program automatically obtains the data control block (DCB) attributes from the input data set and applies them to the output data set. Write the JCL for a job to list the *yourid*.JCL(TEST1) member to SYSOUT=X.

7.9.3 Examining the TSO logon JCL

The password panel of the TSO logon process contains the name of the JCL procedure used to create a TSO session. There are several procedures with different characteristics.

Look at the ISPFPROC procedure. The instructor can help find the correct library for ISPFPROC.

- What is the name of the basic TSO program that is executed?
- Why are there so many DD statements? Notice the concatenation.

Look for procedure IKJACCNT. This is a minimal TSO logon procedure.

7.9.4 Exploring the master catalog

Go to ISPF option 6 and do the following:

- Use a LISTC LEVEL(SYS1) command for a basic listing of all the SYS1 data sets in the master catalog.
- Notice that they are either NONVASM or CLUSTER (and associated DATA and INDEX entries). The CLUSTERs are for VSAM data sets.
- Use the PA1 key to end the listing (for help, see 3.3.3, “Using the PA1 key” on page 3-14).
- Use a LISTC LEVEL(SYS1) ALL command for a more extended listing.
  Note the volser and device type data for the NONVSAM data sets. This is the basic information in the catalog.
- Use LISTC LEVEL(xxx) to view one of the ALIAS levels and note that it comes from a user catalog.

Note: If you enter the profile command with NOPREFIX, it produces a system-wide display when you enter the commands LISTC and LISTC ALL. These commands allow you to see all of the entries in the master catalog, including ALIAS entries.

7.9.5 Using SDSF

From the ISPF Primary Option Menu, locate and select the System Display and Search Facility (SDSF). This utility allows you to display output data sets. The ISPF Primary Option Menu typically includes more selections than those listed on first panel, with instructions about how to display the additional selections.

Return to 6.14.1, “Creating a simple job” and repeat through Step 5 if needed. This will provide a job listing for this exercise.

SDSF Exercise 1

While viewing the output listing, assume that you want to save it permanently to a data set for later viewing. At the command input line, enter PRINT D. A window will prompt you to enter a data set name in which to save it. You can use an already existing data set or create a new one.

For this example, create a new data set by entering yourid.cobol.list. In the disposition field, enter NEW. Press Enter to return to the previous screen. Note that the top right corner of the screen displays PRINT OPENED. This means you can now print the listing. On the command input, enter PRINT. Displayed at the top right of the screen will be the number of lines printed (xxx LINES PRINTED). This means the listing has now been placed in the data set that you created. On the command line, enter PRINT CLOSE. At the top right screen you should now see PRINT CLOSED.

Now let's look at the data set you created, yourid.cobol.list, and view the listing. Go to =3.4 and enter your user ID. A listing of all your data sets should appear. Locate yourid.cobol.list and enter a B next to it in the command area. You should see the listing exactly as it appeared when you were using SDSF. You can now return to SDSF ST and purge (P) your listing, because you now have a permanent copy.

Return to the main SDSF panel and enter LOG to display a log of all activity in the system. Here, you can see much the information that the Operations Staff might see. For example, at the bottom of the list, you might see the outstanding Reply messages to which an operator can reply.

```
/R xx,/DISP TRAN ALL
```

Scroll to the bottom to see results. Note that operator commands from the SDSF LOG command must be preceded by a forward slash (/) so that it is recognized as a system command.

Now, enter M in the command input and press F7; this will display the top of the log. Type F and your user ID to display the first entry associated with your user ID. Most likely this will be when you logged onto TSO. Next enter F youridX, where X represents one of the jobs you submitted above. Here you should see your job being received into the JES2 internal reader, and following that a few lines indicating the status of your job as it runs. Perhaps you might see a JCL error, or youridX started | ended.

SDSF Exercise 2

This exercise uses the Print functions above. Save the log into a data set exactly as you did in the Print exercise.

SDSF Exercise 3

In this exercise, you enter operator commands from the Log screen. Enter the following at the Command input line and look at the resulting displays:

/D A,L	This lists all active jobs in the system.
/D U,,,A80,24	This lists currently online DASD VOLUMES.
/V A88,OFFLINE	Scroll to the bottom to see results (M F8).
/D U,,,A88,2	Check its Status; note that VOLSER is not displayed for offline volumes. While a volume is offline, you can run utilities such as ICKDSF, which allows you to format a volume.
/V A88,ONLINE	Scroll to the bottom and see the results.
/D U,,,A88,2	Check its status; VOLSER is now displayed.
/C U=*yourid*	Cancels a job (your TSO session in this case).
Logon *yourid*	Log back onto your ID.

7.9.6 Using TSO REXX and ISPF

In the data set USER.CLIST there is a REXX program called ITSODSN. This program can be run by entering the following at any ISPF Command input: TSO ITSODSN. It will prompt you to enter the name of the data set you want to create. You do not need to enter yourid, as TSO will add it to the name if your prefix is active. It will give you a choice of two types of data sets, sequential or partitioned, and asks you what volume you want to store the data set on. It will then allocate the data set with your user ID appended to it. Go to =3.4, locate the data set, and examine it with an S option to be sure it is what you want.

REXX Exercise 1

In the REXX program you will find several characteristics of the data set that have been coded for you, for example the LRECL and BLKSIZE. Modify the program so that the user is prompted to enter any data set characteristics as they wish. You may also change the program in any other way that you like. Hint: Make a backup copy of the program before you begin.

REXX Exercise 2

REXX under TSO and batch can directly address other subsystems, as you have already seen in this program when it directly allocates a data set using a TSO command enclosed in quotes. Another way of executing functions outside of REXX is through a host command environment. A few examples of host command environments are:

TSO	
MVS	For REXX running in a non-TSO environment
ISPEXEC	Access to the ISPF environment under TSO

Modify the REXX program so that after the data set is allocated it opens it up with the ISPF Edit command, enters some data, exits with PF3 and then uses =3.4 to examine your data set. Remember that if the data set is partitioned (PO), you have to open up a member. You can use whatever you want as a member name in the format: yourid.name(membername).

Hints:

- It is easier to use the second format of the host command environment above.
- Notice the use of the REXX “if then else” logic and the “do end” within the logic.
- Use the command: ADDRESS ISPEXEC “edit DATASET(….)”

Chapter 10 Exercises (Section 10.9)

These lab exercises help you develop skills in preparing programs to run on z/OS. To perform the lab exercises, you or your team require a TSO user ID and password (for assistance, see the instructor).

10.9.1 Exercise: compiling and linking a program

In this section, use at least two programming languages to compile and link; see the JCL in:

yourid.LANG.CNTL(language)
where “language” is one of ASM, ASMLE, C, C2, COBOL, COBOL2, PL1, PL12.

Do this exercise before attempting the exercise in 10.9.2, “Exercise: executing a program”. The results of successfully running each job in this exercise will be to create the load modules which will be executed in the next exercise.

Note: The JCL will need to be modified to specify the high-level qualifier (HLQ) of the student submitting the jobs. In addition, any jobs referring to Language Environment data sets might also need to be modified. See the comment boxes for more information.

To submit the jobs, enter SUBMIT on the ISPF command line. Once the job completes, you will need to use SDSF to view the output of the job.

a. Submit the following data set to compile and link a complex Assembler language program:
   yourid.LANG.CNTL(ASMLE)
   Note: The student might need to modify the JCL for data sets beginning with CEE. Ask your system programmer what the high-level qualifier (HLQ) is for the Language Environment data sets. The JCL that might need to be changed is highlighted here:
   ```
   //C.SYSLIB    DD DSN=SYS1.MACLIB,DISP=SHR
   //            DD DSN=CEE.SCEEMAC,DISP=SHR
   //C.SYSIN     DD DSN=ZUSER##.LANG.SOURCE(ASMLE),DISP=SHR
   //L.SYSLMOD   DD DSN=ZUSER##.LANG.LOAD(ASMLE),DISP=SHR
   //L.SYSLIB    DD DSN=CEE.SCEELKED,DISP=SHR
   //            DD DSN=CEE.SCEELKEX,DISP=SHR
   ```
b. Submit the following data set to compile and link a simple Assembler language program:
   yourid.LANG.CNTL(ASM)
c. Submit the following data set to compile and link a complex C language program:

yourid.LANG.CNTL(C)
Note: The student might need to modify the JCL for data sets beginning with CEE and CBC. Ask your system programmer what the high-level qualifiers (HLQs) are for the Language Environment and C language data sets. The JCL that might need to be changed is highlighted here:
//STEP1 EXEC PROC=EDCCB,LIBPRFX=**CEE**,LNGPRFX=**CBC**,
// INFILE='ZUSER##.LANG.SOURCE(C)',
// OUTFILE='ZUSER##.LANG.LOAD(C),DISP=SHR'

d. Submit the following data set to compile and link a simple C language program:
yourid.LANG.CNTL(C2)
Note: The student might need to modify the JCL for data sets beginning with CEE and CBC. Ask your system programmer what the high-level qualifiers (HLQs) are for the Language Environment and C language data sets. The JCL that might need to be changed is highlighted here:
//STEP1 EXEC PROC=EDCCB,LIBPRFX=**CEE**,LNGPRFX=**CBC**,
// INFILE='ZUSER##.LANG.SOURCE(C2)',
// OUTFILE='ZUSER##.LANG.LOAD(C2),DISP=SHR'

e. Submit the following data set to compile and link a complex COBOL language program:
yourid.LANG.CNTL(COBOL)
Note: The student might need to modify the JCL for data sets beginning with CEE. Ask your system programmer what the high-level qualifier (HLQ) is for the Language Environment data sets. The JCL that might need to be changed is highlighted here:
//SYSIN   DD DSN=ZUSER##.LANG.SOURCE(COBOL),DISP=SHR
//COBOL.SYSLIB DD DSN=**CEE.SCEESAMP**,DISP=SHR
//LKED.SYSLMOD DD
DSN=ZUSER##.LANG.LOAD(COBOL),DISP=SHR

f. Submit the following data set to compile and link a simple COBOL language program:
yourid.LANG.CNTL(COBOL2)

g. Submit the following data set to compile and link a complex PL/I language program:
yourid.LANG.CNTL(PL1)
Note: The student might need to modify the JCL for data sets beginning with CEE. Ask your system programmer what the high-level qualifier (HLQ) is for the Language Environment data sets. The JCL that might need to be changed is highlighted here:
//SYSIN   DD DSN=ZUSER##.LANG.SOURCE(PL1),DISP=SHR
//PLI.SYSLIB   DD DSN=**CEE.SCEESAMP**,DISP=SHR
//BIND.SYSLMOD DD DSN=ZUSER##.LANG.LOAD(PL1),DISP=SHR

h. Submit the following data set to compile and link a simple PL/I language program: *yourid*.LANG.CNTL(PL12)

10.9.2 Exercise: executing a program

In this section, language examples you selected were compiled and linked in exercise 10.9.1, “Exercise: compiling and linking a program”. Do not attempt to run any of the following jobs if you have not successfully completed the previous exercise, because they will end in errors.

The following exercise contains actions to perform for each language sample to execute the load module that was previously stored when a compile and link job was run. For the interpreted languages, you will execute the source members directly from:

> *yourid*.LANG.SOURCE(language)
> where “language” is one of CLIST, REXX.

Note: The JCL will need to be modified to specify the HLQ of the student submitting the jobs. To submit the jobs, enter SUBMIT on the ISPF command line. Once the job completes, you will need to use SDSF to view the output of the job.

In order for these jobs to run successfully, the student will have had to complete the compile and link jobs in exercise 10.9.1, “Exercise: compiling and linking a program” in order to create the load modules in:

> *ZPROF*.LANG.LOAD

If these jobs did not run successfully, then the student could receive errors in the job log in SDSF similar to:

> CSV003I **REQUESTED MODULE** ASM **NOT FOUN**D
> CSV028I **ABEND806-04** JOBNAME=ZPROF2 STEPNAME=STEP1
> IEA995I SYMPTOM DUMP OUTPUT 238
> **SYSTEM COMPLETION CODE=806 REASON CODE=00000004**

The module name, JOBNAME and STEPNAME vary, according to which job had been submitted.

1. Submit the following data set to execute a complex Assembler language program:
   yourid.LANG.CNTL(USEASMLE)
   This example accesses z/OS Language Environment and prints the message:
   “IN THE MAIN ROUTINE”.
2. Submit the following data set to execute a simple Assembler language program:
   yourid.LANG.CNTL(USEASM)
   This example sets the return code to 15 and exits.
3. Submit the following data set to execute a complex C language program:
   yourid.LANG.CNTL(USEC)
   This example prints out the local date and time.

4. Submit the following data set to execute a simple C language program:
   yourid.LANG.CNTL(USEC2)
   This example prints out the message “Hello World”.
5. Submit the following data set to execute a complex COBOL language program:
   yourid.LANG.CNTL(USECOBOL)
   This example prints out the local date and time.
6. Submit the following data set to execute a simple COBOL language program:
   yourid.LANG.CNTL(USECOBO2)
   This example prints out the message “HELLO WORLD”.
7. Submit the following data set to execute a complex PL/I language program:
   yourid.LANG.CNTL(USEPL1)
   This example prints out the local date and time.
8. Submit the following data set to execute a simple PL/I language program:
   yourid.LANG.CNTL(USEPL12)
   This example prints out the message “HELLO WORLD”.
9. Execute the following complex CLIST language program:
   yourid.LANG.SOURCE(CLIST)
   This example prompts the user for a high-level qualifier (HLQ) and then produces a formatted catalog listing for that HLQ.
      On the ISPF command line type:
      TSO EX ‘*yourid*.LANG.SOURCE(CLIST)’
      When prompted, enter the HLQ *youri*d
10. Execute the following simple CLIST language program:
   yourid.LANG.SOURCE(CLIST2)
   This example prints out the message “HELLO WORLD”.

   On the ISPF command line type:
   TSO EX ‘*yourid*.LANG.SOURCE(CLIST2)’
11. Execute the following complex REXX language program:
   yourid.LANG.SOURCE(REXX)
   This example prompts the user for a high-level qualifier (HLQ) and then produces a formatted catalog listing for that HLQ.

   On the ISPF command line type:
   TSO EX ‘*yourid*.LANG.SOURCE(REXX)’
   When prompted, enter the HLQ *youri*d
12. Execute the following simple REXX language program:
   yourid.LANG.SOURCE(REXX2)
   This example prints out the message “HELLO WORLD”.

   On the ISPF command line type:
   TSO EX ‘*yourid*.LANG.SOURCE(REXX2)’

Instructor Notes

7.) The correct answer for these exercises is to have a job that had a return code of 0 for both the compile and link steps. This can be seen by looking in the job log in SDSF. A sample is listed below, but the actual look of the job log will change from one installation to another, so the headings might vary greatly.

```
IEF403I ZPROFC - STARTED - TIME=11.11.13
-JOBNAME STEPNAME PROCSTEP RC EXCP
-ZPROFC STEP1 COMPILE 00 1201
-ZPROFC STEP1 BIND 00 224
```

8.) With the exception of the simple Assembler language program, all of the jobs should finish with a return code of 0. The simple Assembler language program should have a return code of 15.

9.) This CLIST provides a formatted listing of the catalog for the HLQ that is entered when prompted. A sample is shown here:

```
DATASET ZPROF.BRODCAST IS ON VOLUME TOTSTD
DATASET ZPROF.HFS IS ON VOLUME TOTSTV
DATASET ZPROF.INST.CNTL IS ON VOLUME TOTSTA
DATASET ZPROF.LOG.MISC IS ON VOLUME TOTST4
DATASET ZPROF.SC04.ISPF42.ISPPROF IS ON VOLUME TOTST5
DATASET ZPROF.SC04.SPFLOG1.LIST IS ON VOLUME TOTST0
DATASET ZPROF.SC04.SPFTEMP0.CNTL IS ON VOLUME TOTSSN
DATASET ZPROF.ZSCHOLAR.$INDEX IS ON VOLUME TOTTS4
DATASET ZPROF.ZSCHOLAR.$INDEX.DEVELOP IS ON VOLUME TOTSSJ
DATASET ZPROF.ZSCHOLAR.AREA.CODES IS ON VOLUME TOTTSU
DATASET ZPROF.ZSCHOLAR.CLASS.LOAD IS ON VOLUME TOTSTJ
...
DATASET ZPROF.ZSCHOLAR.LIB.SOURCE IS ON VOLUME TOTSSA
DATASET ZPROF.ZSCHOLAR.PROCLIB IS ON VOLUME TOTTSN
DATASET ZPROF.ZSCHOLAR.PROCLIB.XMIT IS ON VOLUME TOTSSK
DATASET ZPROF.ZSCHOLAR.PROGRAM.LOAD IS ON VOLUME TOTTSO
DATASET ZPROF.ZSCHOLAR.SORT.CNTL IS ON VOLUME TOTSSK
DATASET ZPROF.ZSCHOLAR.SPUFI.CNTL IS ON VOLUME TOTST2
DATASET ZPROF.ZSCHOLAR.SPUFI.OUTPUT IS ON VOLUME TOTSTG
```

10.) Displays HELLO WORLD.
11.) The output is the same as in 9.
12.) The output is the same as in 10.

Chapter 11 Exercises (Section 11.8)

Create a CICS program. During this exercise, you might find it helpful to consult *CICS Application Programming Guide*.

Analyze and update the class program

- Think of a possible use of the COMMAREA.

  Think of passing data between programs called with LINK or XCTL. A generic program for error processing may be developed; all the invocations to it may be done passing the required error data through the COMMAREA. Also, the COMMAREA option of the return command is designed for passing data between successive transactions in a pseudo-conversational sequence.

  The state of a resource may be passed by the first transaction through COMMAREA in order to be compared to its current state by the second transaction. It may be necessary to know if this has changed since the last interaction before allowing an update. In Web applications, the business logic in a CICS application can be invoked using the COMMAREA interface.

- Several simple updates to the class program transaction may be done quite easily:

  - Include one additional output field in the screen. The maximum value of employee commissions could be an example.

    A new field has to be defined in the map source. Perhaps some literals have to be changed. Assemble the map and generate the new copy file. Modify the program to have another column in the SQL statement and move its content after retrieval to the corresponding new output field in the map. Execute the preparation job for the user program. New copies for program and map are required in the CICS session.

  - Create a transaction that could be like a main menu; one of the options would start the current program.

    Only two variable fields are required in the map for this transaction: the option field and the message line. Only one option has to be initially included, the one for the current ABCD transaction. The same mapset may be used to include the new map. The ABCD transaction has to be modified to do the RETURN TRANSID to the new transaction. Only the following resources have to be added to the CICS system: the new transaction and programs (user program and map).

- Learn about the CICS HANDLE CONDITION statement and find out where it may be used.

  Try to add error control to the RECEIVE CICS command. The MAPFAIL condition occurs when no usable data is transmitted from the terminal after a RECEIVE command.

Business transaction

Analyze a typical business transaction. Think of different CICS programs and transactions that could be needed to accomplish this. Draw a diagram to show the flow of the process.

The example that is developed in *CICS Application Programming Primer* could be appropriate. A department store with credit customers keeps a master file of its customers' accounts. The application performs the following actions:

- Displays customer account records
- Adds new account records
- Modifies or deletes existing account records
- Prints a single copy of a customer account record
- Accesses records by name

Instructor Notes

These exercises provide more practice with CICS programming. It might be necessary to consult CICS Application Programming Guide or other manuals for guidance.

You need to create an output file for SPUFI prior to the class, or you can show the students again how to do it, because they need to do it as well. You need a ZPROF.SPUFI.OUTPUT file, VB with record length of 4092, block length of 4096, 10 tracks should be enough.

More information about accessing files

As explained before, both VSAM files and databases can be accessed from a CICS program. In this section we cover only VSAM data sets. Access to databases in CICS is the same as in other environments.

Each file used in any application must be defined to CICS, and the most important information kept for each is the symbolic file name. When a CICS application program makes a file request, it always uses the symbolic file name.

We may differentiate several options when files are accessed:

Direct reading - you read a record in the file with the READ command. When reading a KSDS, you can identify the record you want by specifying its full key (RIDFLD option), or you can retrieve the first (lowest key) record whose key meets certain requirements. Example 11-2 illustrates how to read a record from a file named MASTER into a specified data area.

Example: Direct reading of a record

```
        EXEC CICS READ
            INTO(RECORD)
            FILE('MASTER')
            RIDFLD(ACCTNO)
```

Sequential reading (browsing) - You start a browse with the STARTBR command, identifying a particular record in the same way as for a direct read. This command only identifies the starting position for the browse; it does not retrieve a record. The READNEXT command reads records sequentially from this starting point.

Record update - To update a record, first retrieve it using one of the file control read commands that specifies the UPDATE option. After modification by the application program, the record is written back to the data set using the REWRITE command.

Record delete - To delete a single record, you may retrieve it for update with a READ UPDATE command, and then issue a DELETE command. You have the option of issuing a DELETE command specifying the RIDFLD option.

Record addition - Add records to a file with the WRITE command. They must always be written from an area specified by the application program.

Groups and lists

Resource definitions held on the CSD are organized into groups and lists:

Group - A collection of related resources on the CSD. Each resource that you define must belong to a group.

List - The name of groups that CICS installs at initial start. Groups do not have to belong to lists, and can be defined independently.

Chapter 12 Exercises (Section 12.14)

Use SPUFI in a COBOL program. You need a connection to DB2 to perform this exercise.

12.14.1 Step 1: Create files

Before you start with the DB2 exercise, you need to create two more PDSs:

- ZUSER##.DB2.INCLUDE, to store your DCLGENs
- ZUSER##.DB2.DBRM, to store your DBRMs

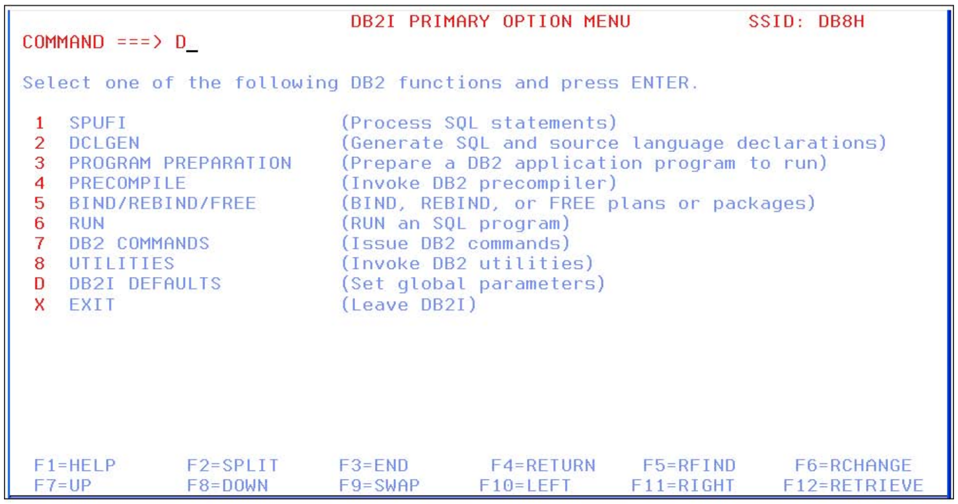
You can use ZUSER##.LANG.CNTL as base.

Furthermore, you also need a ZUSER##.SPUFI.OUTPUT file, which should be a flat file of record format VB, with a record length of 4092 and block length of 4096.

12.14.2 Step 2: DCLGEN

DCLGEN is an easy way to generate COBOL definition statements for the DB2 information that you use in an application program. These statements can then be included in the source program.

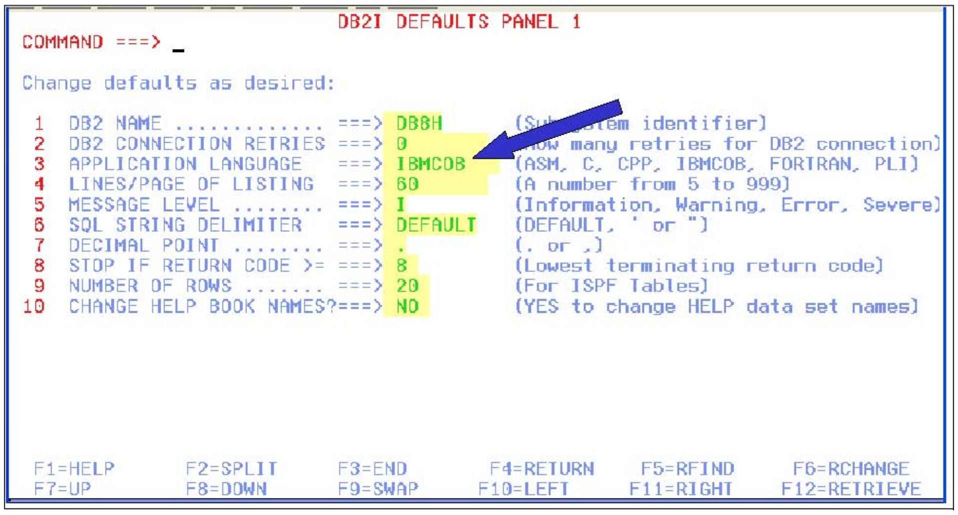
First, from the DB2I (DB2 Interactive) menu, choose D for DB2I Defaults (Figure 12-14) and press Enter.



```
                             DB2I PRIMARY OPTION MENU          SSID: DB8H
COMMAND ===> D_

Select one of the following DB2 functions and press ENTER.

 1  SPUFI                   (Process SQL statements)
 2  DCLGEN                  (Generate SQL and source language declarations)
 3  PROGRAM PREPARATION     (Prepare a DB2 application program to run)
 4  PRECOMPILE              (Invoke DB2 precompiler)
 5  BIND/REBIND/FREE        (BIND, REBIND, or FREE plans or packages)
 6  RUN                     (RUN an SQL program)
 7  DB2 COMMANDS            (Issue DB2 commands)
 8  UTILITIES               (Invoke DB2 utilities)
 D  DB2I DEFAULTS           (Set global parameters)
 X  EXIT                    (Leave DB2I)









 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 12-14   DB2I menu

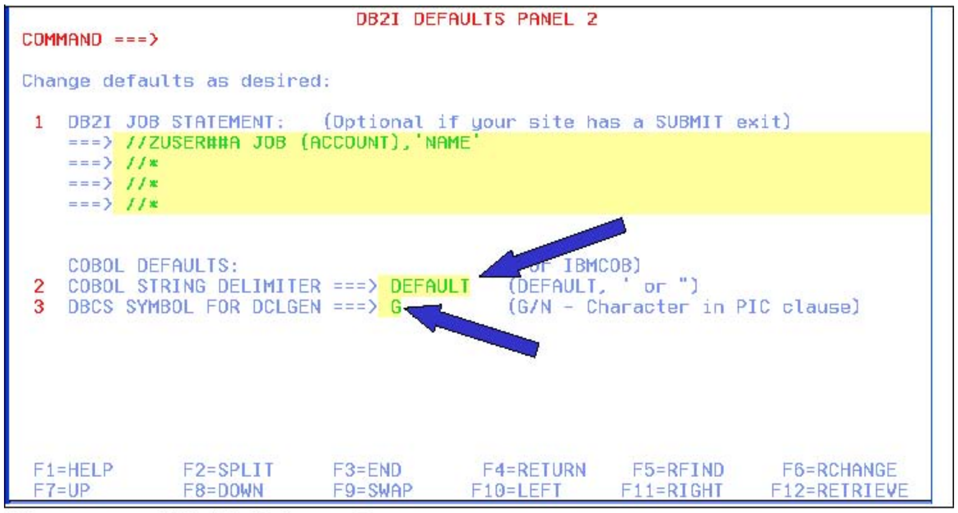
On the DB2I Defaults Panel 1, specify IBMCOB for option 3 Application Language (Figure 12-15).



```
                              DB2I DEFAULTS PANEL 1
COMMAND ===> _

Change defaults as desired:

 1  DB2 NAME ............. ===> DB8H       (Sub    tem identifier)
 2  DB2 CONNECTION RETRIES ===> 0           ow many retries for DB2 connection)
 3  APPLICATION LANGUAGE   ===> IBMCOB     (ASM, C, CPP, IBMCOB, FORTRAN, PLI)
 4  LINES/PAGE OF LISTING  ===> 60         (A number from 5 to 999)
 5  MESSAGE LEVEL ........ ===> I          (Information, Warning, Error, Severe)
 6  SQL STRING DELIMITER   ===> DEFAULT    (DEFAULT, ' or ")
 7  DECIMAL POINT ........ ===> .          (. or ,)
 8  STOP IF RETURN CODE >= ===> 8          (Lowest terminating return code)
 9  NUMBER OF ROWS ....... ===> 20         (For ISPF Tables)
10  CHANGE HELP BOOK NAMES?===> NO         (YES to change HELP data set names)


 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 12-15 DB2I default panel 1

Press Enter, and on DB2I Defaults Panel 2, specify DEFAULT for the COBOL string delimiter under option 2 and G for the DBCS symbol for DCLGEN for option 3. Press Enter (Figure 12-16).



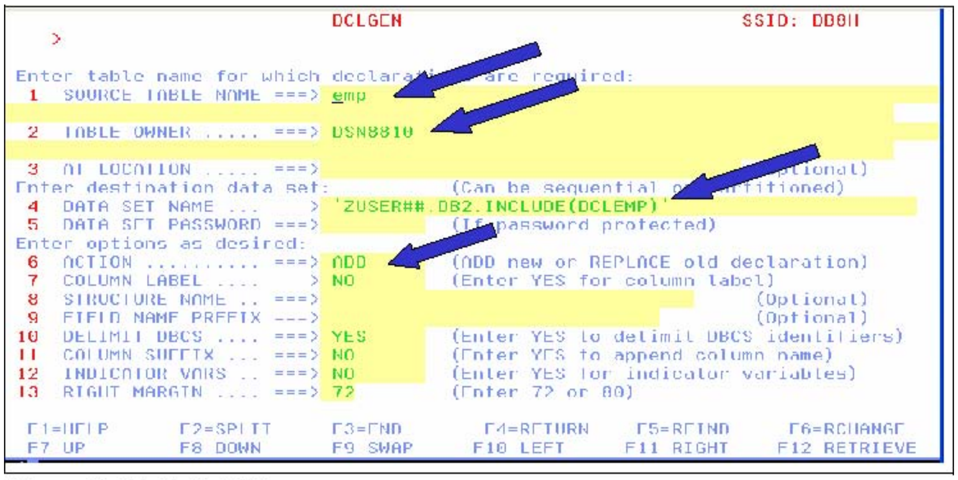
```
                              DB2I DEFAULTS PANEL 2
COMMAND ===>

Change defaults as desired:

 1  DB2I JOB STATEMENT:    (Optional if your site has a SUBMIT exit)
    ===> //ZUSER##A JOB (ACCOUNT),'NAME'
    ===> //*
    ===> //*
    ===> //*


    COBOL DEFAULTS:                         or IBMCOB)
 2  COBOL STRING DELIMITER ===> DEFAULT    (DEFAULT, ' or ")
 3  DBCS SYMBOL FOR DCLGEN ===> G          (G/N - Character in PIC clause)


 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 12-16 DB2I Default panel 2

This is just to make sure that you have the correct language.

After Enter, you are back on the main DB2I panel (Figure 12-14); now select option 2, DCLGEN.

You need to have a destination data set already allocated to hold your DCLGEN definition (ZUSER##.DB2.INCLUDE); it should be created for you. If you do not have one, go to the ISPF menu and create a PDS file.



```
                              DCLGEN                                 SSID: DB8H
  >

Enter table name for which declarations are required:
 1  SOURCE TABLE NAME ===> emp

 2  TABLE OWNER ..... ===> DSN8810

 3  AT LOCATION ..... ===>                                          (Optional)
Enter destination data set:          (Can be sequential or partitioned)
 4  DATA SET NAME ...    > 'ZUSER##.DB2.INCLUDE(DCLEMP)'
 5  DATA SET PASSWORD ===>           (If password protected)
Enter options as desired:
 6  ACTION .......... ===> ADD       (ADD new or REPLACE old declaration)
 7  COLUMN LABEL ....    > NO        (Enter YES for column label)
 8  STRUCTURE NAME .. ===>                                     (Optional)
 9  FIELD NAME PREFIX --->                                     (Optional)
10  DELIMIT DBCS .... ===> YES       (Enter YES to delimit DBCS identifiers)
11  COLUMN SUFFIX ... ===> NO        (Enter YES to append column name)
12  INDICATOR VARS .. ===> NO        (Enter YES for indicator variables)
13  RIGHT MARGIN .... ===> 72        (Enter 72 or 80)

 F1=HELP      F2=SPLIT     F3=END      F4=RETURN    F5=RFIND     F6=RCHANGE
 F7 UP        F8 DOWN      F9 SWAP    F10 LEFT     F11 RIGHT    F12 RETRIEVE
```

Figure 12-17 DCLGEN

As Figure 12-17 shows, you need to specify the table, the table owner, your PDS file, and the action ADD. The resulting message should be:

> **EXECUTION COMPLETE, MEMBER DCLEMP ADDED**
> *** * ***

If the definition of the table changes, you must also change DCLGEN and use REPLACE.

12.14.3 Step 3: Test your SQL

Go to SPUFI; use your SPUFI.CNTL PDS. In that PDS you find the member SELECT. This is the SQL statement you will use in your program. The where-clause is not there, so that you can see all the results you can get. It also gives you the opportunity to know what departments are available in the table.

Surely, for more complex queries, this is common practice. As an application developer you are sure to execute the right SQL.

12.14.4 Step 4: Create the program

Here, you can create a program, or use the program that is supplied for you in LANG.SOURCE(COBDB2). This sample program calculates the average salary for one department. You specify the department and get the result. To end the program, enter 999.

To modify this program, add the following:

- Your variables (include the member you have created in step 1).
- Specify the SQL delimiters for COBOL.

If you search for “???” you will find the locations to do this.

12.14.5 Step 5: Complete the program

Edit the LANG.CNTL(COBDB2) job and make the changes stated at the top of the job.

You find the following steps in this job:

- Step PC: this is the DB2 precompile; it splits your source into two parts: the DBRM and the modified source.
- Steps COB, PLKED and LKED: these do the compile and linking of your modified source.
- Step BIND: this does the bind of the package and the plan.

  Question: If you needed to change your program, which bind could be left out? Feel free to change the program. Instead of the average, you can ask the minimum or maximum salary within a department (then you just need to change the SQL).
- Step Run: this runs the program in batch for two departments: A00 and D21.

12.14.6 Step 6: Run the program from TSO

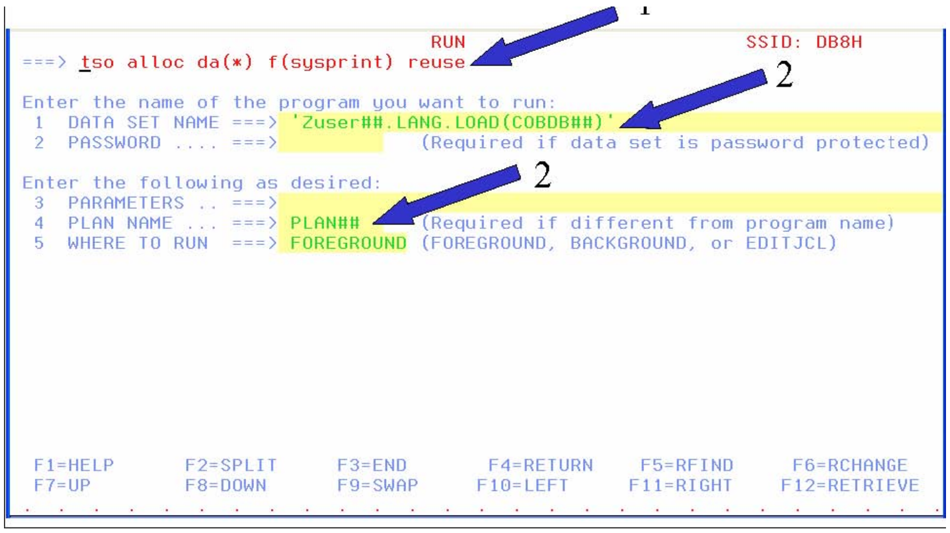
Instead of running your program in batch, try running it from the TSO READY prompt. To do so, you must allocate both files to your session (this must be done before you run the job).

Enter the following and press Enter after each line:

```
TSO alloc da(*) f(sysprint) reuse
tso alloc da(*) f(sysin) reuse
```

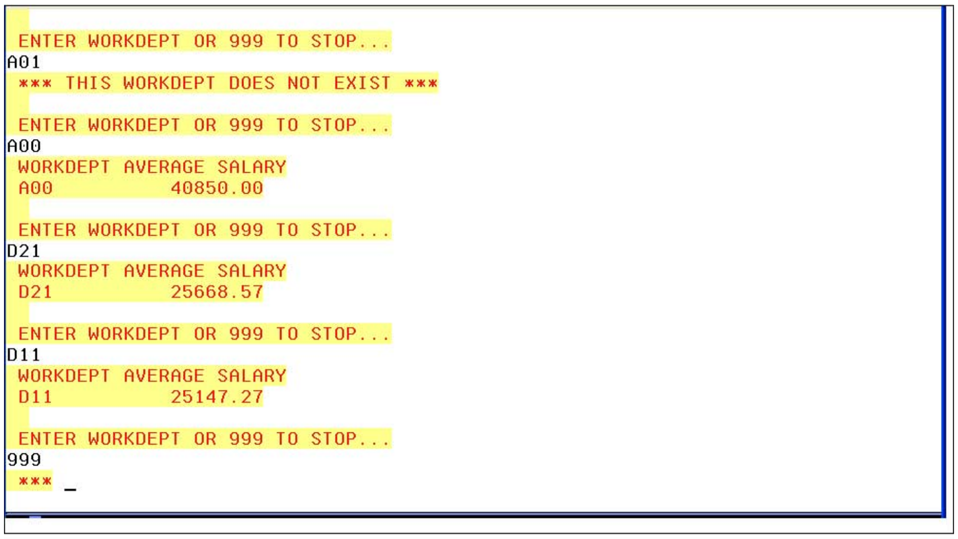
Then return to your DB2I screen.

Select option 6 RUN. Here, you enter the file name and the plan name (Figure 12-18).



```
                                   RUN                         SSID: DB8H
===> tso alloc da(*) f(sysprint) reuse

Enter the name of the program you want to run:
 1  DATA SET NAME ===> 'Zuser##.LANG.LOAD(COBDB##)'
 2  PASSWORD .... ===>              (Required if data set is password protected)

Enter the following as desired:
 3  PARAMETERS .. ===>
 4  PLAN NAME ... ===> PLAN##       (Required if different from program name)
 5  WHERE TO RUN  ===> FOREGROUND (FOREGROUND, BACKGROUND, or EDITJCL)

 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 12-18 Ready to execute



```
 ENTER WORKDEPT OR 999 TO STOP...
A01
 *** THIS WORKDEPT DOES NOT EXIST ***

 ENTER WORKDEPT OR 999 TO STOP...
A00
 WORKDEPT AVERAGE SALARY
 A00          40850.00

 ENTER WORKDEPT OR 999 TO STOP...
D21
 WORKDEPT AVERAGE SALARY
 D21          25668.57

 ENTER WORKDEPT OR 999 TO STOP...
D11
 WORKDEPT AVERAGE SALARY
 D11          25147.27

 ENTER WORKDEPT OR 999 TO STOP...
999
 *** _
```

Figure 12-19 The execution of the program

Instructor Notes

These exercises provide more practice with DB2. It might be necessary to consult the DB2 manuals. This exercise requires a connection to DB2.

Before students begin, they need to create two more PDSs. Both will normally only contain one member unless they want to do some more programming. You do need to grant the students the needed authority. We did not test it, because we had no additional userid, but we think it is covered in ZPROF.INST.CNTL(GRANTS). The additional exercises are in comments.

During the exercises, students perform three major steps:

1. Create a DCLGEN
2. Execute SQL with SPUFI
3. Change and run a COBOL program

DCLGEN is straightforward; everything is explained well. An error may happen if they press Enter twice, because the member then already exists.

The SQL is completely in ZUSER##.SPUFI.CNTL, but if they want, they can add (before the GROUP BY): WHERE WORKDEPT = ‘A00’, then they will only receive one row back.

Concerning the program: it is a simple program, reading only one row (to avoid making them work with cursors), so make sure that whatever SQL they use there, the result is only one row. If not, you get an error during execution. You find a solution of the program in ZPROF.INST.CNTL(COBDB2). The JCL to use is COBDB2J; the DCLGEN is also there.

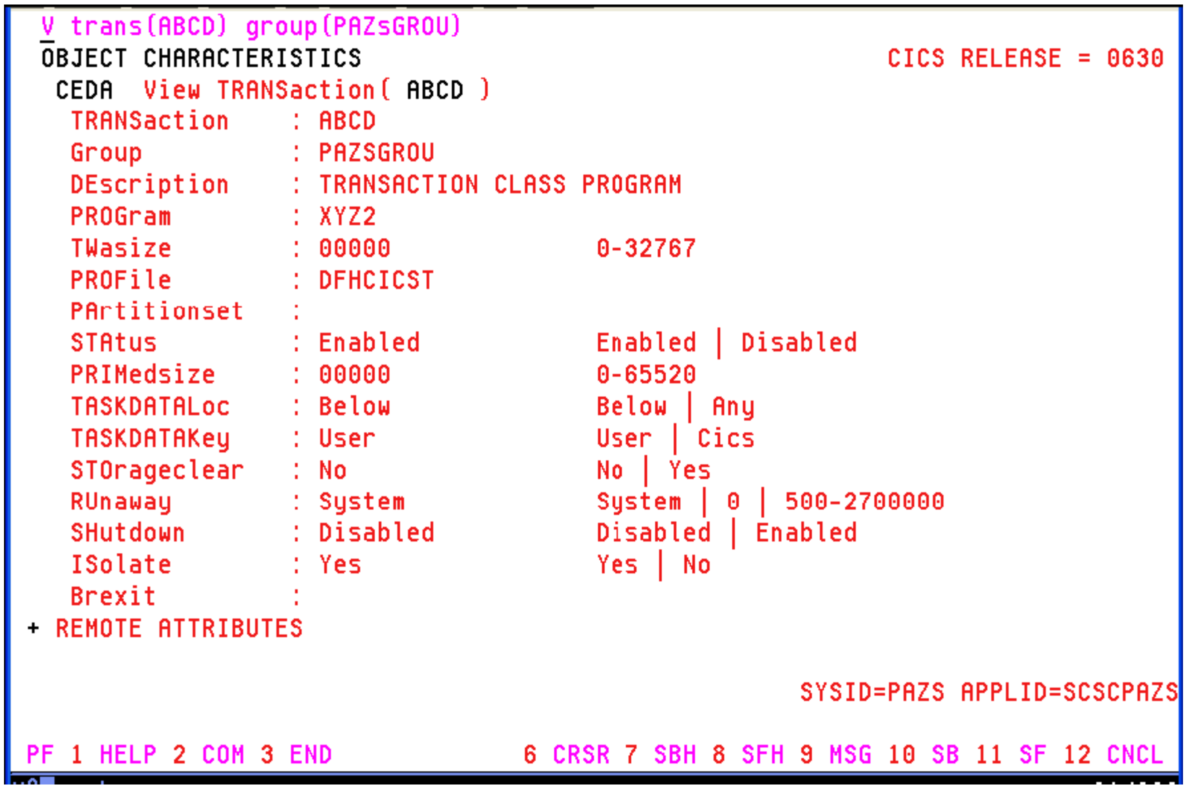
If that does not work, you have the DBRM and the LOADLIB also in the same directory. So if you bind the DBRM (you can use the online screens of DB2I) in the package and create a plan, the execution should also work.

In the Class directories, you have the option to use SQL (depending on the level of students’ familiarity with SQL), and an EXPLAIN exercise. The columns for a plan table are provided in SPUFI.CNTL(PLANTAB). Of course, you need a database in which you can create the table. If you want the students to do so, you must give them access to create tables in a database as well.

The exercise: The map and its assembling and link-editing; the program and its compile-link and go, the creation of the object in CICS, and the transaction itself as well. They will think it is not that much work (because you have prepared everything), but as you know, it is not always that simple. To make it easier for you, we will guide you through the hardest parts (also see Appendix E, “Class Program”).

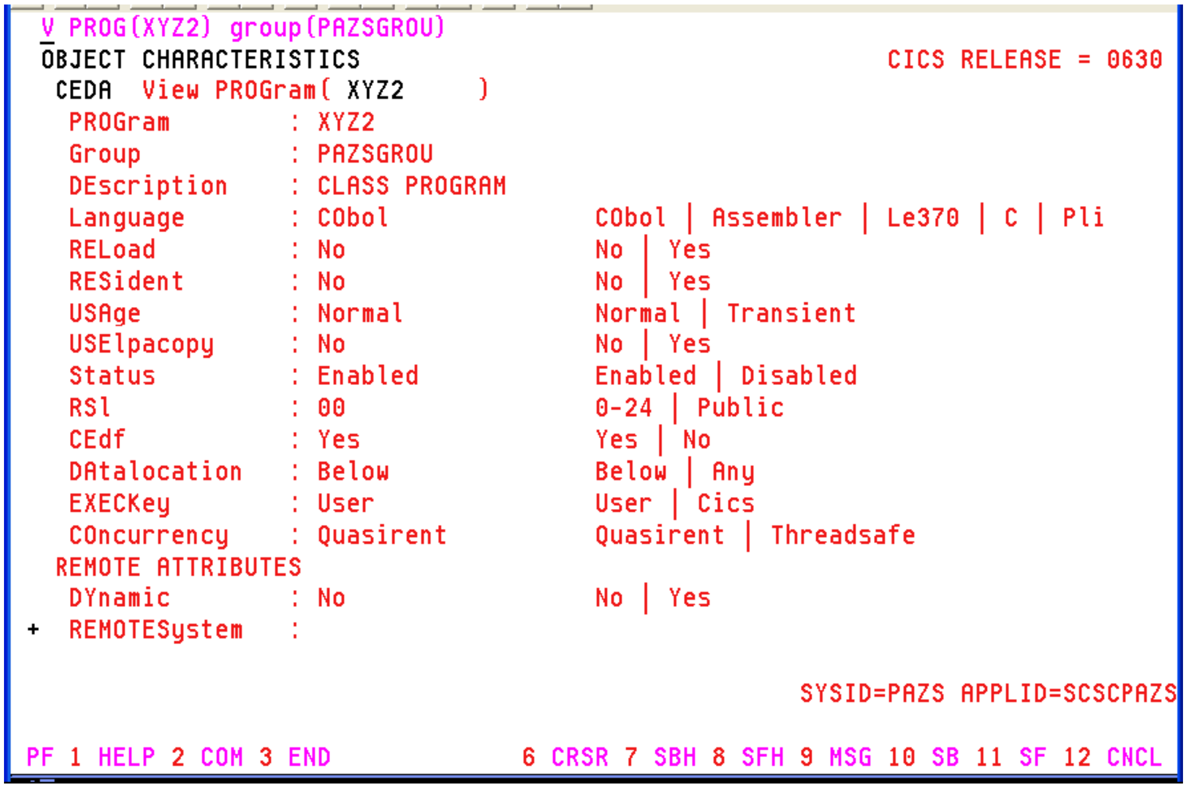
1. Assemble and link-edit the map (you could do this before the class, to avoid unpleasant surprises). This can be done through the execution of ZPROF.CLASS.SAMPLIB(MAPASSEM).

2. Precompile, CICS translation, etc., of the COBOL program itself can be done through the execution of ZPROF.CLASS.SAMPLIB(CICSDB2P).
3. Do the CICS definitions as shown in the following screens (we did the view afterwards, to be sure that everything was working).
4. Through CEDA AD GROUP, you can add the group (here it is PAZSGROU).
5. Through CEDA DEF TRANS, define the transaction.

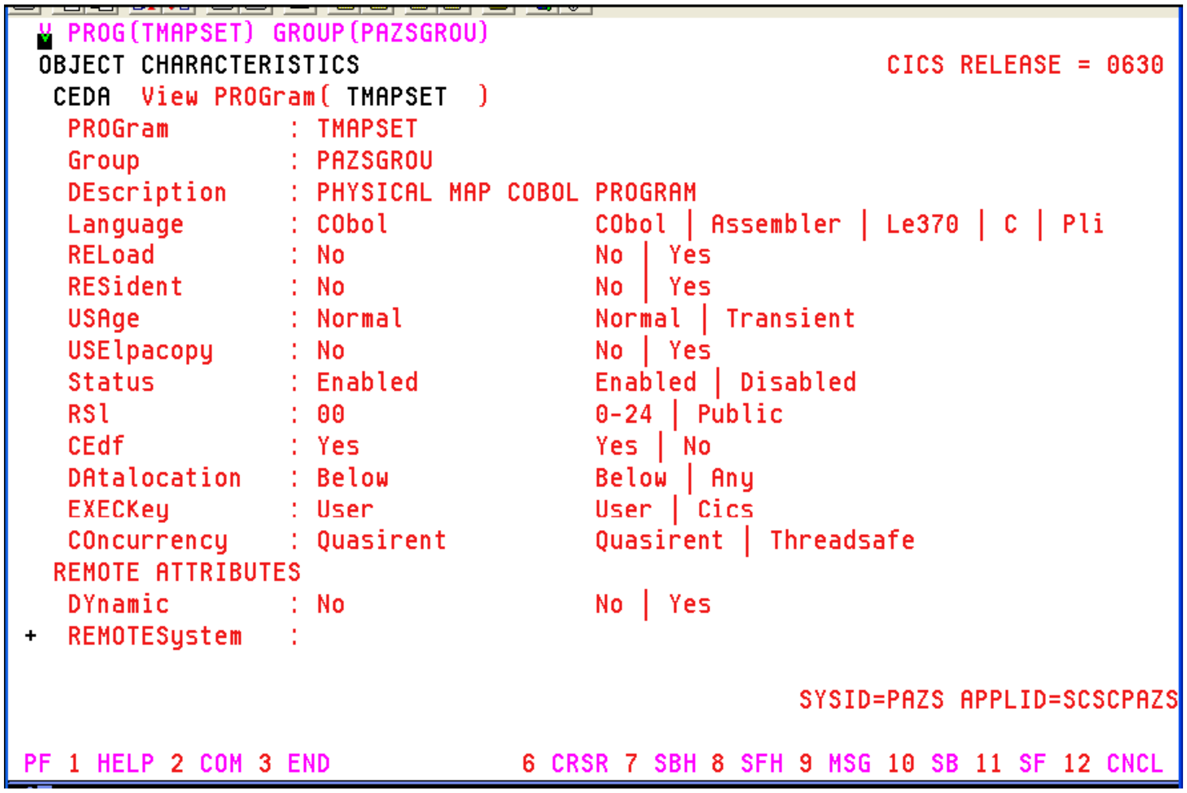


```
  V trans(ABCD) group(PAZsGROU)
  OBJECT CHARACTERISTICS                                            CICS RELEASE = 0630
   CEDA  View TRANSaction( ABCD )
    TRANSaction     : ABCD
    Group           : PAZSGROU
    DEscription     : TRANSACTION CLASS PROGRAM
    PROGram         : XYZ2
    TWasize         : 00000                0-32767
    PROFile         : DFHCICST
    PArtitionset    :
    STAtus          : Enabled              Enabled | Disabled
    PRIMedsize      : 00000                0-65520
    TASKDATALoc     : Below                Below | Any
    TASKDATAKey     : User                 User | Cics
    STOrageclear    : No                   No | Yes
    RUnaway         : System               System | 0 | 500-2700000
    SHutdown        : Disabled             Disabled | Enabled
    ISolate         : Yes                  Yes | No
    Brexit          :
 + REMOTE ATTRIBUTES

                                                               SYSID=PAZS APPLID=SCSCPAZS

 PF 1 HELP 2 COM 3 END              6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Through CEDA DEF PROG, define the program and the mapset:

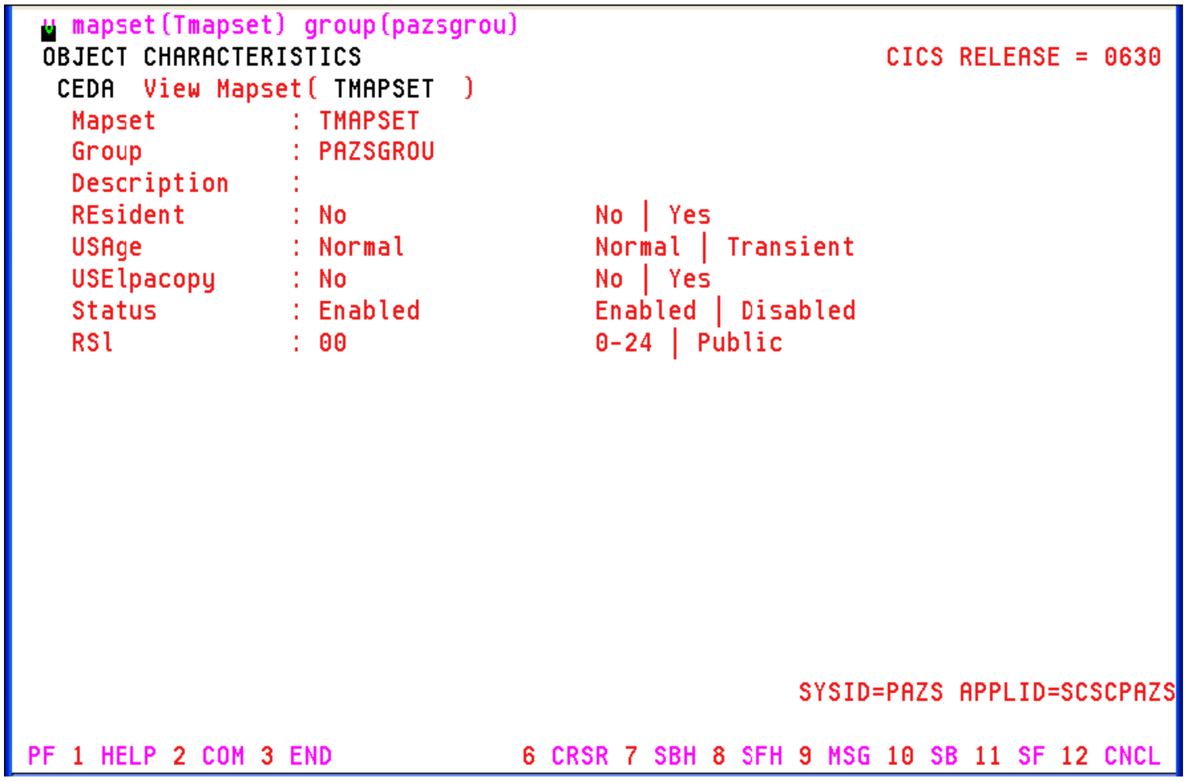


```
 V PROG(XYZ2) group(PAZSGROU)
 OBJECT CHARACTERISTICS                                      CICS RELEASE = 0630
  CEDA  View PROGram( XYZ2     )
   PROGram        : XYZ2
   Group          : PAZSGROU
   DEscription    : CLASS PROGRAM
   Language       : COboL              COboL | Assembler | Le370 | C | Pli
   RELoad         : No                 No | Yes
   RESident       : No                 No | Yes
   USAge          : Normal             Normal | Transient
   USElpacopy     : No                 No | Yes
   Status         : Enabled            Enabled | Disabled
   RSl            : 00                 0-24 | Public
   CEdf           : Yes                Yes | No
   DAtalocation   : Below              Below | Any
   EXECKey        : User               User | Cics
   COncurrency    : Quasirent          Quasirent | Threadsafe
  REMOTE ATTRIBUTES
   DYnamic        : No                 No | Yes
 + REMOTESystem   :

                                                         SYSID=PAZS APPLID=SCSCPAZS

PF 1 HELP 2 COM 3 END              6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```



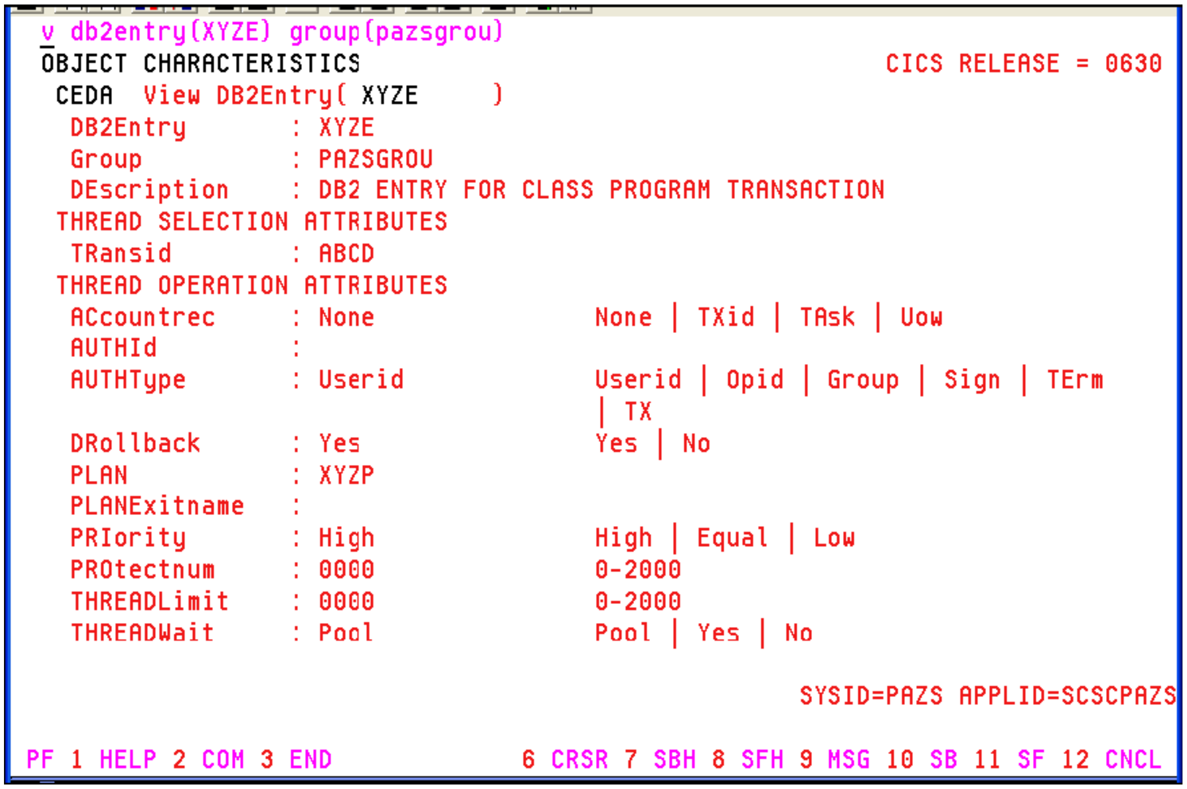
```
 V PROG(TMAPSET) GROUP(PAZSGROU)
 OBJECT CHARACTERISTICS                                      CICS RELEASE = 0630
  CEDA  View PROGram( TMAPSET  )
   PROGram        : TMAPSET
   Group          : PAZSGROU
   DEscription    : PHYSICAL MAP COBOL PROGRAM
   Language       : COboL              COboL | Assembler | Le370 | C | Pli
   RELoad         : No                 No | Yes
   RESident       : No                 No | Yes
   USAge          : Normal             Normal | Transient
   USElpacopy     : No                 No | Yes
   Status         : Enabled            Enabled | Disabled
   RSl            : 00                 0-24 | Public
   CEdf           : Yes                Yes | No
   DAtalocation   : Below              Below | Any
   EXECKey        : User               User | Cics
   COncurrency    : Quasirent          Quasirent | Threadsafe
  REMOTE ATTRIBUTES
   DYnamic        : No                 No | Yes
 + REMOTESystem   :

                                                         SYSID=PAZS APPLID=SCSCPAZS

PF 1 HELP 2 COM 3 END              6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

The mapset itself also needs to be defined. This can be done through CEDA DEF mapset(Tmapset) group (pazsgrou).



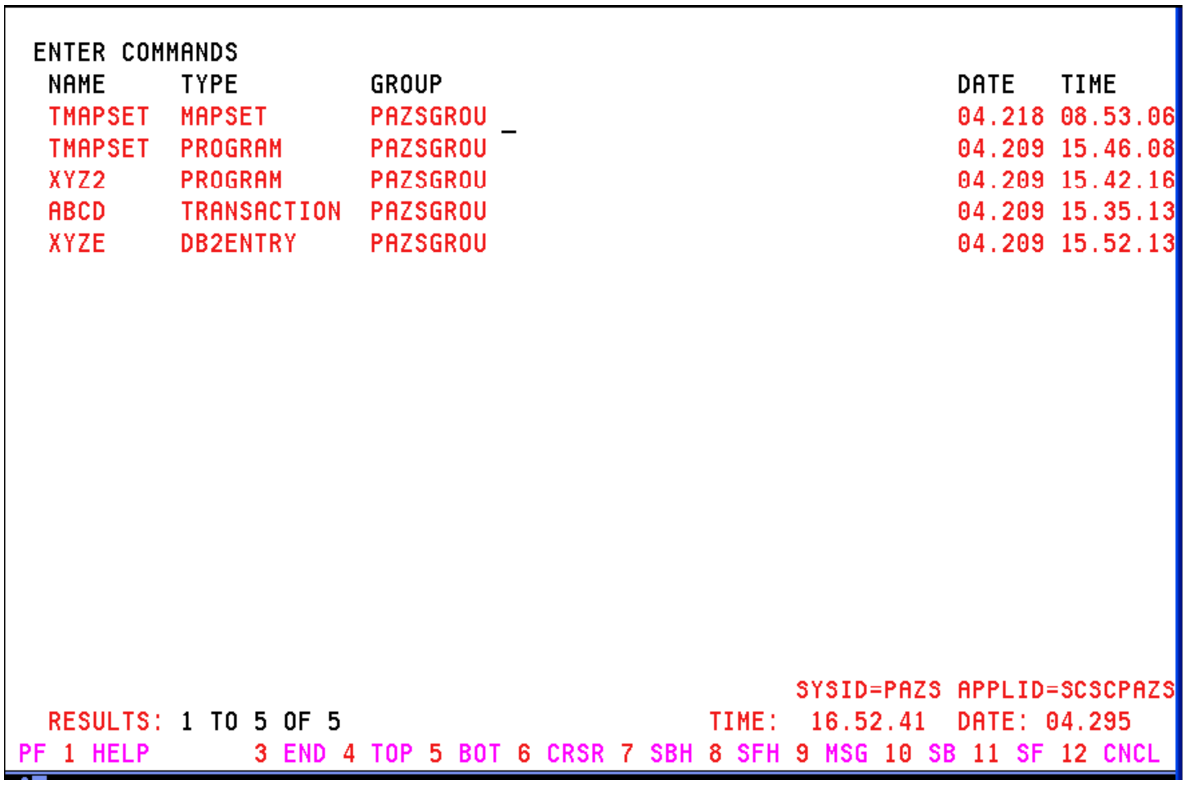
```
 v mapset(Tmapset) group(pazsgrou)
 OBJECT CHARACTERISTICS                                        CICS RELEASE = 0630
  CEDA  View Mapset( TMAPSET  )
   Mapset         : TMAPSET
   Group          : PAZSGROU
   Description    :
   REsident       : No                   No | Yes
   USAge          : Normal               Normal | Transient
   USElpacopy     : No                   No | Yes
   Status         : Enabled              Enabled | Disabled
   RSl            : 00                   0-24 | Public










                                                          SYSID=PAZS APPLID=SCSCPAZS

PF 1 HELP 2 COM 3 END                 6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

You also need to define the DB2Entry through CEDA DEF DB2ENTRY.



```
 v db2entry(XYZE) group(pazsgrou)
 OBJECT CHARACTERISTICS                                       CICS RELEASE = 0630
  CEDA  View DB2Entry( XYZE     )
   DB2Entry        : XYZE
   Group           : PAZSGROU
   DEscription     : DB2 ENTRY FOR CLASS PROGRAM TRANSACTION
  THREAD SELECTION ATTRIBUTES
   TRansid         : ABCD
  THREAD OPERATION ATTRIBUTES
   ACcountrec      : None                None | TXid | TAsk | Uow
   AUTHId          :
   AUTHType        : Userid              Userid | Opid | Group | Sign | TErm
                                         | TX
   DRollback       : Yes                 Yes | No
   PLAN            : XYZP
   PLANExitname    :
   PRIority        : High                High | Equal | Low
   PROtectnum      : 0000                0-2000
   THREADLimit     : 0000                0-2000
   THREADWait      : Pool                Pool | Yes | No

                                                        SYSID=PAZS APPLID=SCSCPAZS

PF 1 HELP 2 COM 3 END             6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

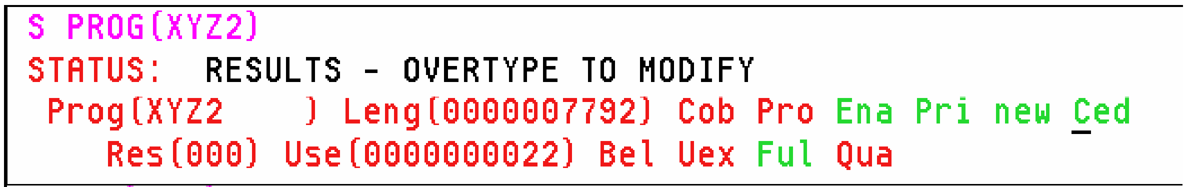
If you enquire all defined objects for group PAZSGROU, you should get this through CEDA DI:



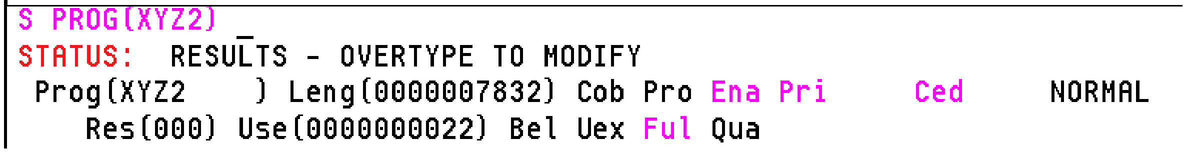
```
 ENTER COMMANDS
  NAME      TYPE          GROUP                                       DATE   TIME
  TMAPSET   MAPSET        PAZSGROU _                                  04.218 08.53.06
  TMAPSET   PROGRAM       PAZSGROU                                    04.209 15.46.08
  XYZ2      PROGRAM       PAZSGROU                                    04.209 15.42.16
  ABCD      TRANSACTION   PAZSGROU                                    04.209 15.35.13
  XYZE      DB2ENTRY      PAZSGROU                                    04.209 15.52.13











                                                           SYSID=PAZS APPLID=SCSCPAZS
  RESULTS: 1 TO 5 OF 5                                TIME:  16.52.41  DATE: 04.295
PF 1 HELP       3 END 4 TOP 5 BOT 6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

6. Start the transaction ABCD. Valid departments are A00, D11, D21.

It may be a good idea to show them also that they need to refresh the programs (and maps) if they change those as well. That usually happens through CEMT S PROG(XYZ2). Here are the two screens, one before the newcopy (with "new") and one after:



```
S PROG(XYZ2)
STATUS:  RESULTS - OVERTYPE TO MODIFY
 Prog(XYZ2    ) Leng(0000007792) Cob Pro Ena Pri new Ced
    Res(000) Use(0000000022) Bel Uex Ful Qua
```



```
S PROG(XYZ2) _
STATUS:  RESULTS - OVERTYPE TO MODIFY
 Prog(XYZ2    ) Leng(0000007832) Cob Pro Ena Pri     Ced     NORMAL
    Res(000) Use(0000000022) Bel Uex Ful Qua
```

The changes that were made are in the program XYZ2. They remove the leading zeroes in front of the salary (an easy way to see that this is the new version).

Chapter 13 Exercises (Section 13.6)

Use the OMVS shell for this exercise. Also, you will need to know:

- The location of the HTTP Server configuration file httpd.conf is /web/httpd1/httpd.conf
- The web address for the HTTP Server is http://zos.kctr.marist.edu

Do each of the following steps and answer the questions:

1. Browse the httpd.conf file of the HTTP Server product installed on z/OS. In which directory are the Web documents stored (F "URL translation rules")? Also, which port should be used? (F "Port directive")?
2. From a Web browser window, display the class HTTP Server. How is WebSphere plugged into this HTTP Server? (F "Websphere")?
3. Use OEDIT to create an HTML document in the Web documents folder. For the purpose of this exercise, we will be storing the HTML documents in the directory /Z17S/usr/lpp/internet/server_root/Docs/studentweb. Save your HTML document in this directory as *yourid*test.html. Here is an example:
```
<!doctype html public "//W3//Comment//EN">
<html>
<head>
<META content="text/html; charset=iso-8859-1">
<title> This is a simple HTML Exercise</title>
</head>
<body bgcolor="#FFFFFF">
<p>Hello World
</body>
</html>
```
4. Open a Web browser to your HTML document, for example:
   www.yourserver.com/youridtest.html
   yourserver should be replaced with the web address of the HTTP Server, followed by the pathname that comes after server_root where you stored your HTML file. A logon window should appear prompting you for a username and password. Use your TSO username and password.
   What needs to be done to install your own CGI?
5. Examine the httpd.conf file. Is the HTCounter CGI option "Date and Time" enabled? If so, change *yourid*text.html and add the following line to the body section:
```
<img src="/cgi-bin/datetime?Timebase=Local">
```
   Save the file. What has changed?

Chapter 16 Exercises (Section 16.12)

1. Find out which IEASYSxx members were used in the current IPL. Did the operator specify the suffix of an alternate IEASYSxx?
2. Did the operator specify any parameter in response to the message SPECIFY SYSTEM PARAMETERS? If the answer is Y, find the related PARMLIB members for that parameter and obtain the parameter value that would be active if that operator response hadn't occurred.
3. Do the following:
   - On your system, find out the IPL device address and the IPL Volume. Go to SDSF, enter ULOG, and then /D IPLINFO.
   - What is the IODF device address?
   - What is the LOADxx member that was used for IPL? What is the data set that contains this LOADxx member?
   - Browse this member; what is the name of the system catalog used by the system?
   - What is the name of the IODF data set currently used? Enter /D IOS,CONFIG.
   - The system parameters can come from a number of PARMLIB data sets. Enter /D PARMLIB. What are the PARMLIB data sets used by your system?

Chapter 18 Exercises (Section 18.11)

1. Try to log on to TSO after changing the initial logon procedure IKJACCNT to IKJACCN1. The expected message is:

   ```
   IKJ56483I THE PROCEDURE NAME IKJACCN1 HAS NOT BEEN
   AUTHORIZED FOR THIS USERID
   ```
2. Using your TSO user ID (now with your default logon procedure IKJACCNT), try to delete the data set *ZPROF*.JCL.NOT.DELETE, which is set up by the standard jobs in the supplied JCL. This is a protected data set and you can only read its content.
3. Execute the next sample JCL to obtain a DSMON utility report with the current RACF group tree structure (available in the sample JCL as member DSMON):

   ```
   //DSMONRPT JOB
   (POK,999),'DSMONREPORT',MSGLEVEL=(1,1),MSGCLASS=X,
   // CLASS=A,NOTIFY=&SYSUID
   /*JOBPARM SYSAFF=*
   //*
   //*  NOTE:
   //* REMEMBER THAT ICHDSM00 MUST BE RUN BY A USER WITH
   AUDITOR ATTRIBUTE
   //*
   //STEPNAME EXEC PGM=ICHDSM00
   //SYSPRINT DD SYSOUT=A
   //SYSUT2 DD SYSOUT=A
   //SYSIN DD *
   FUNCTION RACGRP
   /*
   ```

4. Verify that the SYS1.LINKLIB library is an APF-authorized library.
   - Using the DISPLAY APF command to display the entire APF list.
   - Using the ENTRY= operand in the DISPLAY APF command.
   - Using the DSNAME= operand in the DISPLAY APF command. Verify the entry number in the command display result in the syslog.
5. The following JCL example can be used to invoke the ADRDSSU utility and issue a WTOR message in the console. The WTOR command lets you write an ADR112A message to the system console. The ADR112A message requests that the operator perform some action, and then issue a reply. You can use WTOR, for example, to request that the operator mount a required volume or quiesce a database before your DFSMSdss job continues to process (available in the sample JCL as member ADRDSSU).

```
//WTORTEST JOB (POK,999),'USER',MSGLEVEL=(1,1),MSGCLASS=X,
// CLASS=A,NOTIFY=&SYSUID
//       EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=*
//SYSIN  DD *
WTOR 'TEST'
/*
```

DFSMSdss assigns the following routing code to the WTOR message:

1	Primary operator action

DFSMSdss assigns the following descriptor code to the WTOR message:

2	Immediate action required

In the SDSF main screen, choose the SR option (system requests) and reply with any response you want.