# Chapter 10. Compiling and link-editing a program on z/OS (20 marks)

Student Name: Yu-Hsuan Huang

Student Marist Userid: KC03CCA

## 10.9.1 Exercise: Compiling and linking a program

In this section, use at least two programming languages to compile and link a program using the following JCL data set: yourid.LANG.CNTL(language). Where language is either ASM, ASMLE, C, C2, COBOL, COBOL2, PL1, or 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 is creating the load modules that will be executed in the next exercise.

Perform the following steps:

Notes:

* The JCL needs to be modified to specify the high-level qualifier (HLQ) of the student submitting the jobs.
* To submit the jobs, enter SUBMIT on the ISPF command line. After the job completes, you need to use SDSF to view the output of the job.

**Assembler**

1. Submit the following data set to compile and link a complex Assembler language program:

yourid.LANG.CNTL(ASMLE)

From SDSF, XDC all job output to yourid.LANG.OUTPUT(ASMLE).

2. Submit the following data set to compile and link a simple Assembler language program:

yourid.LANG.CNTL(ASM)

From SDSF, XDC all job output to yourid.LANG.OUTPUT(ASM).

**C**

Note: for all the C exercises please use the following JCLLIB statement.

//ZSCHOLIB JCLLIB ORDER=(KC02315.ZSCHOLAR.PROCLIB)

3. Submit the following data set to compile and link a complex C language

program:

yourid.LANG.CNTL(C)

From SDSF, XDC all job output to yourid.LANG.OUTPUT(C).

4. Submit the following data set to compile and link a simple C language

program:

yourid.LANG.CNTL(C2)

From SDSF, XDC all job output to yourid.LANG.OUTPUT(C2).

**COBOL**

Change the LNGPRX parameter on EXEC to LNGPRFX=IGY420.

5. Submit the following data set to compile and link a complex COBOL language

program:

yourid.LANG.CNTL(COBOL)

From SDSF, XDC all job output to yourid.LANG.OUTPUT(COBOL).

6. Submit the following data set to compile and link a simple COBOL language

program:

yourid.LANG.CNTL(COBOL2)

From SDSF, XDC all job output to yourid.LANG.OUTPUT(COBOL2).

**PL/I**

Note: For all the PL/I exercises please use the following JCLLIB statement.

//ZSCHOLIB JCLLIB ORDER=(KC02315.ZSCHOLAR.PROCLIB)

7. Submit the following data set to compile and link a complex PL/I language

program:

yourid.LANG.CNTL(PL1)

From SDSF, XDC all job output to yourid.LANG.OUTPUT(PL1).

8. Submit the following data set to compile and link a simple PL/I language

program:

yourid.LANG.CNTL(PL12)

From SDSF, XDC all job output to yourid.LANG.OUTPUT(PL12).

## 10.9.2 Exercise: Executing a program

Do not attempt to run any of the following jobs if you have not successfully completed the exercise in 10.9.1, “Exercise: Compiling and linking a program” because they will end in errors.

The following exercise contains actions to execute, for each language sample, the load module that was previously stored when a compile and link job was run.

For the interpreted languages, you execute the source members directly from: yourid.LANG.SOURCE(language). Where language is either of CLIST or REXX.

Perform the following steps:

**Assembler**

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 following message:

HI GEORGIAN COLLEGE STUDENT.

From SDSF, XDC all job output to yourid.LANG.OUTPUT(USEASMLE).

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.

From SDSF, XDC all job output to yourid.LANG.OUTPUT(USEASM).

**C**

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.

From SDSF, XDC all job output to yourid.LANG.OUTPUT(USEC).

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.

From SDSF, XDC all job output to yourid.LANG.OUTPUT(USEC2).

**Cobol**

5. Submit the following data set to execute a complex COBOL language program:

yourid.LANG.CNTL(USECOBOL)

This example prints out: HELLO WORLD ON date AT time.

From SDSF, XDC all job output to yourid.LANG.OUTPUT(USECOBOL).

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.

From SDSF, XDC all job output to yourid.LANG.OUTPUT(USECOBO2).

**PL/I**

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.

From SDSF, XDC all job output to yourid.LANG.OUTPUT(USEPL1).

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.

From SDSF, XDC all job output to yourid.LANG.OUTPUT(USEPL12).

**TSO CLIST**

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, enter: TSO EX ‘yourid.LANG.SOURCE(CLIST)’

When prompted, enter the HLQ yourid.

Copy and paste the first few lines of output produced.

ENTER THE HIGH-LEVEL QUALIFIER YOU WOULD LIKE TO QUERY

KC03CCA

DATASET KC03CCA.ASSIGN1.REXX IS ON VOLUME KCTR36

DATASET KC03CCA.ASSIGN1.WORK IS ON VOLUME KCTR18

DATASET KC03CCA.ASSIGN2.CLIST IS ON VOLUME KCTR39

DATASET KC03CCA.ASSIGN2.HINTS IS ON VOLUME KCTR05

DATASET KC03CCA.ASSIGN2.JCL IS ON VOLUME KCTR37

DATASET KC03CCA.ASSIGN2.LOAD IS ON VOLUME KCTR07

DATASET KC03CCA.ASSIGN2.OUTPUT IS ON VOLUME KCTR54

DATASET KC03CCA.ASSIGN2.PANELS IS ON VOLUME KCTR53

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, enter: TSO EX ‘yourid.LANG.SOURCE(CLIST2)’

Copy and paste the output produced.

HELLO WORLD

\*\*\*

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, enter: TSO EX ‘yourid.LANG.SOURCE(REXX)’

When prompted, enter the HLQ yourid.

Copy and paste the first few lines of output produced.

ENTER THE HIGH-LEVEL QUALIFIER YOU WOULD LIKE TO QUERY

KC03CCA

DATASET KC03CCA.ASSIGN1.REXX IS ON VOLUME KCTR36

DATASET KC03CCA.ASSIGN1.WORK IS ON VOLUME KCTR18

DATASET KC03CCA.ASSIGN2.CLIST IS ON VOLUME KCTR39

DATASET KC03CCA.ASSIGN2.HINTS IS ON VOLUME KCTR05

DATASET KC03CCA.ASSIGN2.JCL IS ON VOLUME KCTR37

DATASET KC03CCA.ASSIGN2.LOAD IS ON VOLUME KCTR07

DATASET KC03CCA.ASSIGN2.OUTPUT IS ON VOLUME KCTR54

DATASET KC03CCA.ASSIGN2.PANELS IS ON VOLUME KCTR53

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, enter: TSO EX ‘yourid.LANG.SOURCE(REXX2)’

Copy and paste the output produced.

Hello World from

\*\*\*