CS241 : Assignment II

Prof. Hemangee K. Kapoor
Dr. Aryabartta Sahu
Department of CSE,
IIT Guwahati

Submission Due: August 14, 2018

August 8, 2018

Instructions for SCCS:

I. Install sccs by sudo apt-get install cssc (For Ubuntu)
II. Try these commands one after another:

```
mkdir sccs_test
cd sccs_test
echo 'hallo' > quit.c
mkdir SCCS
sccs create quit.c
sccs edit quit.c
echo 'bye' >> quit.c
sccs delta quit.c
```

The last command asks for a comment input. Then try out these questions.
1. Create a file named "iitg.txt" with five different versions and each version has some changes from its previous version.
2. How to add a new comment to a 1.3 version of the iitg.txt file ?
3. Display the changes made to the 4th version of the iitg.txt file.
4. How to check the changes and then retrieves a read-only copy of the new version iitg.txt file.
5. How to retrieve the latest version of the file iitg.txt, which was created before 18th Aug. 2018 at 5pm.
6. How to exclude delta changes made at the versions 1.2 and 1.4 should not be reflected on the current version of the iitg.txt file.

Makefiles

A makefile is a specially formatted text file that a unix program called 'make' can interpret. Make simplifies and reduce the time of the compilation of large projects that have a number of files.

Problem 7

In the directory name with problem7, **create a Makefile** that:

1. Has a dependency rule that will only compile the executable if main.c is present.

2. Compiles main.c into an executable named main.

3. Has a clean rule that deletes the executable main.

Follow up Commands:
You have to run the make and make clean command.
Makefile basics: https://www.youtube.com/watch?v=43-2t7CveRI
Tutorial makefile 2 using variable pattern comment:
https://www.youtube.com/watch?v=q7msgDa5-dc
Example: cse241_SD_Lab_make_example.pdf

Problem 8
In the directory name with problem8, create a Makefile that:

1. Has a dependency rule that will only compile the executable if all required object files are present.

2. Compiles cube.o, rectangle.o, cylinder.o, circle.o, and sphere.o into an executable named shapes.

3. Has dependency rules to compile each necessary .c file into its corresponding object file.

4. Has a clean rule that deletes all object files and the executable shapes.

Follow up Commands:
Make - Episode 4 - Rules: https://www.youtube.com/watch?v=fyTh3r4edZs

Problem 9
Create a makefile using generalize variables
Suppose you have a number of files in a project. How to generalize a makefile to avoid the redundancy of commands and quickly run the code by just writing a few lines of makefile code? (Hint:You can replace the target/dependency names with the patterns $@ (automatic variable), $<, $^, $? etc.)

1. Use **echo** to print the text and variables value/text on terminal.

2. How to print the text in different colors using makefile.

Follow up Commands:
Tutorial makefile 3 program saved in directory bin inc src
https://www.youtube.com/watch?v=Wt6KGlMkLz0
Make - Episode 3 - Patterns: https://www.youtube.com/watch?
v=PYtaEeMGsX8list=PLbuVyodeL1URagPxP5BZ_SKaI_A9G78yjindex=3pbjreload=10

Problem 10
Uses of Macro in makefile
In the third directory, create a Makefile that:

1. Makes use of the macro LIBS for the library and header file path.

2. Using the macro stated above, compiles the file lib_driver.c into an executable named lib_driver by linking to the libA.a and libB.a libraries and Lib_Header.h header file contained in another directory test/macro/libs2.

3. Has a clean rule that deletes the executable lib_driver.

Follow up Commands: Make - Episode 5 - Macros: https://www.youtube.com/watch?v=buxd7FcyAU4index=5list=PLbuVyodeL1URagPxP5BZ_SKaI_A9G78yj

Problem 11
Run multiple makefiles through a single makefile
In the fourth directory, create a Makefile that:

1. Makes use of two macros LIBS and EXES that correspond to the library and header file path and executables.
2. Combines the three previous Makefiles into a single Makefile that can compile the three executables with a single make command.
3. Has a clean rule that deletes all executables and object files.
4. Has comments explaining each macro, rule, and dependency.

Problem for the bonus marks
Uses of conditions and for loops in makefile
(Note: This feature is supported by make version 4.2 or its higher versions)

- Part1: Create a make file using if else conditions
- Part2: Create a make file that will use for loop

Follow up Commands:
You may refer the syntax from the tutorial: cse241_SD_Lab_make_tutorial

Debugging
This part of the assignment will make you learn how to debug C/C++ programs using gdb, a widely used command line debugger. In this part, you will use gdb to detect some of the bugs in a small program.
Step 1. Open Vim editor containing Cone.cpp.
Step 2. Compile the program using g++: g++ -g Cone.cpp -o cone
Step 3. Call gdb with the cone.

Problem 12

1. Execute cone by using gdb run command. The program executed normally by asking some of the input values. Record the output.
2. The program contains a bug. It is not tough to debug but let's explore gdb, set the breakpoint on first standard output statement. Record the message appeared on the gdb screen.
3. Execute the cone by using "r" or run command. Record the output on the first stop of the program. Also, report the arrow in the gdb window.
4. Afterward, run "c" or continue command. Report what happens.

Problem 13

1. Again execute the cone by "run" command. Use "next" to move to the first function call. Enter the inputs for the messages prompted on your screen. Print the entered values (height, radius and length) on gdb window. Also, use "display" command to print the value of surfaceArea.

2. Now execute the first assigned statement using the "next" command. What is the output?? However, we just assigned a value to the surfaceArea, check the value again and report the value that is it correct?? If the values are different give the reason?? Check the values of radius again is it correct??

3. Report the value of curvedSurfaceArea is it correct?? if not why is it so??

Problem 14

1. Terminate the debugger, fix the program and compile it again.

2. Run the gdb as before, set two breakpoints one at the first standard output statement and second at the "DisplayOutput()" statement. On reaching the first breakpoint display (using "display" command) the values of PI, radius, height, and length, volume, surfaceArea, and curvedSurfaceArea. Also followed by report the output of info display command.

3. Turn off the display of the radius, height, and length. Run the info display command again and record the output.

4. Use the step command to step into DisplayOutput() report whether the parameters values are correct??

Follow up Commands: For problem 15 and 16, run the make command.
Problem 15
A program fileDirOpen.c is given with a bug inside it. The objective of the given program is to list all the files and directory inside the passing command line argument. By default /etc/ is feed, other paths can be specified using command line argument (eg: , /usr, /home). Debug the given program using gdb.

Problem 16
A program MatrixMultiply.c is given to compute the matrix multiplication using pointers. Check whether the program output the current sets of results or terminate successfully. If not, debug the program using gdb and fix it correctly (Don't change the pointers with normal array).

Problem 1-16 are some set of problems. During the evaluation, we may ask similar kind of questions. For the doubts in the assignment, you can contact to Chinmaya (Problem 1-6), Sonal (Problem 7-11), Sukarn (Problem 12-16).