CSCI 3232 Systems Software Assignment 5

Upload all your files to the correct dropbox folder in Folio before the deadline --- **11:30PM, Mar 5, Thursday, 2020.**

**Note: Make sure you have practiced all relevant sample codes in slides and Folio’s example codes before you start this assignment. Make sure your scripts have Unix style line endings. See slide 32 of 6\_shell\_basics.pptx for details. About makefile submission issue in Folio, see last slide in 3\_Pointers\_Functions.pptx.**

1. (15 pts) Write a bash script **A5p1.sh** to output the number of executable and non-executable files and subdirectories separately in the directory that is specified as the first command line argument to this script. **Do not** count recursively in subdirectories. **Do not** call any external Linux utilities such as “ls”. You may refer to the *testSearch.sh* example and page 7 of the “*Basic Bash Scripting Lab.docx”* documentation in Folio to find out how to test if an item in a directory is an executable or non-executable file or subdirectory. When grading, we will test your script using “./**A5p1.sh <dir>**” where <dir> can be any absolute (those starting with ‘/’) or relative directory (those not starting with ‘/’). A sample sun can look like the following (You do NOT need to submit screen shots. Instead submit your source file.):

[kwang@computer][~/temp]$./A5p1.sh <dir>

Number of executable files in <dir>: 7

Number of non-executable files in <dir>: 6

Number of subdirectories in <dir>: 5

1. (15 pts) Write a bash script **A5p2.sh** that accepts one command line argument a which is assumed to be an integer. Your script should output a list of integers starting from a and ending with 1 according to the iteration rule you are already familiar with from assignment 4 problem 2 (f(x)=x/2 if x is even; f(x)=(3x+1)/2 if x is odd). After that, also report the number of integers in the list. Don’t call any external programs. Implement all algorithms in your script. When grading, we will test your script using “./**A5p2.sh <integer>**”. A sample sun can look like the following (You do NOT need to submit screen shots. Instead submit your source file.):

[kwang@computer][~/temp]$./A5p2.sh 6

6 3 5 8 4 2 1

Length of the sequence: 7

1. (15 pts) Write a bash script **A5p3.sh** to print out 15 equations on 15 lines, one per line. Each of these equations should be of the form i\*j=k, where i is an integer that runs from 1 through 3, j runs from 1 through 5 and k is the product of i and j. (For example, if i is 2 and j is 3, k should be 6.) You will print 15 *different* equations because the number of possible combinations of i and j is 3\*5=15. Your script should use loop to achieve the functionality. Do not define or call any functions in this script.
2. (15 pts) Write a bash script **A5p4.sh** to print the same 15 equations as in problem 3. This time you need to define a function *printProduct* that takes two integer parameters a and b and prints out an equation of the form a\*b=c, where c is the product of the two parameters a and b. The main part of your script should use loop and call the function *printProduct* you defined. That is, you should call the function 15 times (with a different pair of arguments being passed each time) to display the 15 equations.
3. (30 pts) Write a bash script **A5p5.sh** that accepts one command line argument which is supposed to be a positive integer *n*. The script should print all odd integers from 1 through *n*. Write a C or C++ program **A5p5.c(pp)** to read from stdin as many integers as there are available and then print the squares of all these integers. You should test your code by **“./A5p5.sh <n> | ./A5p5**” assuming the compiled C/C++ program is A5p5. Submit both files. See the following for a sample run. (You do NOT need to submit screen shots. Instead submit your source file.)

[kwang@computer][~/temp]$bash A5p5.sh 10 | A5p5

1 9 25 49 81

1. (10 pts) Write a bash script **A5p6.sh** to call your bash script A5p2.sh with **all** integers from 26 through 35 on the command line. That is, you call A5p2.sh 10 times in your script **A5p6.sh** using a loop, each time supplying a unique integer between 26 and 35 (inclusive) on the command line for your script A5p2.sh. Make sure your 10 output lists of integers are clearly separated.
2. Submit a working makefile to compile your C/C++ program in problem 5. Make sure your makefile works by testing it. Without a working makefile, up to 2 pts could be deducted.

Checklist for 8 files to be submitted: A5p1.sh, A5p2.sh, A5p3.sh, A5p4.sh, A5p5.c(pp), A5p5.sh, A5p6.sh and makefile for problems 5.