Assignment #5: Bash Scripting

Due Date: Wednesday, February 23 at 5:00 PM.

Type of Assignment: Individual

Submission Procedure: Upload copies of your programs, output, and assignment report to your private git repository in the directory:

`Assignments/Assignment_5/Submission`

Instructions:

Complete the following problems. Each problem involves writing a Bash program and then running the program. Include your programs with your submission as well as a report containing the command or commands for running each program and the corresponding output. If the output consists of a number of files, include these files with your submission.

Problem 1:

Write a Bash shell script program that takes two integer arguments. The second argument is assumed to be greater than the first. The output of the program is a counting of numbers beginning at the first number and ending with the second number.

Test your program using the numbers 6 and 21 as input.

Problem 2:

Write a Bash shell script program that takes the name of a file or directory as an argument and reports if the file is a directory, regular file, or other and also reports if the user has read, write, and execute permission on the file or directory.

Test your program using the file `/usr/bin/nice` as input.

Hint: There are many ways to approach the problem, but you may find it useful to use some combination of the Unix commands `ls`, `read`, `head`, and `tail`. Note that you will want to use special options with these commands. For instance, you can use `ls -l file` to list the properties of a file named `file` and `head -c N` to read the first `N` characters.

Problem 3:

Write a Bash shell script program that will rename all the files with a particular file extension in a directory. This program is especially useful for renaming digital picture files. The first argument should be a base name while the second argument should be a file extension. For example, if you run your program as:

```
$ ./renumber MyPicture png
```

then the resulting files should have names like:

`MyPicture001.png`, `MyPicture002.png`, `MyPicture003.png`, etc.

Test your program using the `jpg` picture files in the zipped tar file:

```
Assignments/Assignment_5/Materials/Pictures.tar.gz
```

located within the `Shared` git repository. Rename the files using the base name `Picture`.

Problem 4:

Write a Bash shell script program that computes the average grade from a simple grade file. The grade file will contain multiple lines and each line will contain a grade and its descriptive information. These items will be separated by a semicolon. You need to read the file line by line, parse the grade of each file, and sum them up to get the total score. The grade in each line will be an integer. The contents of each file will look like:

Assignment1, Needs improvement;75
Assignment2, Improvement;85
Assignment3, More improvement;95
Assignment4, Excellent work;100

After your program obtains the total score, it will need to divide the total score by the number of assignments to find the final grade. Your program should then display this average score, in floating point format with at least one digit after the decimal point, to the screen.

An example execution of the program should look as follows:

```
./grader Parr.txt
```

Test your program using the assignment grade files in the zipped tar file:

```
Assignments/Assignment_5/Materials/Assignments.tar.gz
```

located within the `Shared` git repository.

Hint: You may find it useful to employ the `cut` command with the `-d` flag.

Problem 5:

Write a Bash shell script program that organizes the files within a zipped tar file. Your program should first unzip the file. Then, your program should create subfolders for each file extension associated with the extracted files. Next, your program should place files within subfolders associated with their particular file extension. If a file does not have a file extension, it should not be placed within any particular subfolder. Finally, your program should zip the resulting folder structure (excluding, of course, the original zipped tar file). This new tar file should have the identifier `_clean` within its name. For instance, if the original zipped tar file was named `Data.tar.gz`, the new zipped tar file should be named `Data_clean.tar.gz`.

Test your program using the zipped tar file:

```
Assignments/Assignment_5/Materials/MISC.tar.gz
```

located within the `Shared` git repository.