Programming Assignment 3: Shell Programming
CSC 4103: Operating Systems, Spring 2017

Due: April 30th, Sunday (by 11:59 PM)
Total Points: 10

Instructions: Compile and test-run your code on the classes server. Submit your work as instructed and verify your submission. The verify command will display your submission date/time. Include your name and classes account in all source code files. Necessary documentation should be included to instruct how to compile and run your code. Late submissions will be penalized at the rate of 10% per day late and no more than 3 days late.

Objective

To learn the use of Bash shell scripting.

Background

The Bash shell scripting is a powerful tool to execute a batch of Unix commands/programs for completing a complicated task. In the class, we have learned the basics of Bash Shell programming. This project is a programming practice to use shell for system information collection and visualization.

Programming Task

System runtime information is important for administrators to monitor the system status. In this project, you need to write a Bash shell script to automatically collect system usage information (CPU and disk) for a number of iterations, and generate two bargraph figures to report the system usage information. Your program should follow the guideline below.

(1) Your script, `sysbar.sh`, should give help info. Typing command "`$./sysbar.sh help`" should print help information to explain how to use your script.

(2) The script should accept 2 input parameters to allow a user to specify the frequency and the total number of iterations of collecting the CPU and disk usage info.

```
$./sysbar.sh <interval in seconds> <number of collections>
```

The first parameter specifies the interval (in seconds) between two consecutive iterations of collecting system status info, and the second parameter specifies the total number of information collections. For example, "`$./sysbar.sh 10 6`" instructs the program to collect the system usage information every 10 seconds, for one minute.

Note: Your script should provide basic check to prevent accepting incorrect input. For example, no more than two parameters should be accepted. The input numbers should be in a reasonable range (The definition of a "reasonable range" should be in your help info).

(3) As specified by the user input, the script should collect the CPU and disk usage information for a specified number of iterations with a specified interval between any two consecutive iterations. Your script should use `mpstat` command to collect the CPU utilization information (`usr`, `sys`, and `idle`) and use `iostat` command to collect the read and write bandwidth (`kB_read/s`, `kB_wrtn/s`). These data will be used as data

items for creating the stacked bargraph. The script needs to use the collected data to create two `.plot` files (`cpu_usage.plot` and `disk_usage.plot`) as input for the `bargraph.pl` to generate two `.eps` figures. See additional hints about the two commands in the next section below.

Note: The two plot files, "`cpu_usage.plot`"and "`disk_usage.plot`", should be produced by your script as output.

(4) Upon completion, the shell script should eventually generate two output `eps` figures, "`cpu_usage.eps`" and "`disk_usage.eps`". The two figures are bargraphs showing the results of your collected system usage information for CPU and disk usage, respectively. Here is the requirement for the generated figures.

a. The figure should be a stacked bar graph.
b. The X-axis should show the time of data collection in format of "HH:MM:SS".
c. The CPU bargraph figure should display three data items (`usr`, `sys`, and `idle`), generated by `mpstat`. That is, each bar in the figure should show the three data items collected at the corresponding time, being illustrated as three parts, each corresponding to one of the three data items. The Y-axis should be "Percentage (%)". The title should be "CPU Utilization".
d. The Disk bargraph figure should include two components (`kB_read/s`, `kB_wrtn/s`), generated by `iostat`.), generated by `mpstat`. That is, each bar in the figure should show the three data items collected at the corresponding time, being illustrated as three parts, each corresponding to one of the three data items. The Y-axis should be "Bandwidth (kB/sec)". The X-axis should show the time of data collection in format of "HH:MM:SS". The title should be "Disk Utilization".
e. Two eps files ("`cpu_usage.eps`" and "`disk_usage.eps`") should be generated.

Here are the example plot and `eps` files:
http://www.csc.lsu.edu/~fchen/class/csc4103-sp17/example.plot
http://www.csc.lsu.edu/~fchen/class/csc4103-sp17/example.eps

(5) The script **MUST** use functions to organize code into multiple parts.

Additional Hints

You may need to use several Linux commands/tools to complete this task.

(1) `date` command can produce the current time.

```
$ date +%H:%M:%S'
```

(2) `mpstat` command can produce the CPU utilization information as follows.

```
PM  CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest  %gnice   %idle
PM  all    0.02    0.00    0.01    0.00    0.00    0.00    0.00    0.00    0.00   99.97
```

(3) `iostat` command can produce the disk utilization information as follows.

```
Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
```

```
sda               0.19         2.63       116.93    1865295   82977728
```

(4) You may use `awk` to retrieve a specified item from a row of data. For example,

```
$ echo $items | awk '{print $3;}'
```

The above example returns the 3rd item in the row of items.

(5) You may use `sleep` to pause for a few seconds (creating an internval). For example,

```
$ sleep 5
```

stops the execution of the shell script, sleeps for 5 seconds, and then resumes the execution.

(6) Using `bargraph.pl` to generate the stack bargraph figures:

Go to http://www.csc.lsu.edu/~fchen/class/csc4103-sp17/assignments/bargraph.pl and download the `bargraph.pl` script. This tool can help you produce the output bargraph files. You may use the following command to produce the eps file in your script.

```
$ perl bargraph.pl <plot file> > <output eps file>
```

For example, "`$ perl bargraph.pl example.plot > example.eps`" takes `example.plot` file as input and generates the output figure file "`example.eps`". So, your script should automatically create such a `.plot` file to feed the bargraph.pl command. Here is an example input "example.plot":

```
=stacked; A; B
title=This is an illustration title
colors=yellow,red
xlabel=This is X Label
ylabel=This is Y Label
=table
=norotate
yformat=%g

16:0:0          10    50
16:1:0          20    60
16:2:0          30    70
16:3:0          40    80
16:4:0          50    90
```

Explanation of the `example.plot` file:

a. "`=stacked; A; B`" specifies this figure is a stacked bar graph, and each bar includes two data items: the first one is "A" and the second one is "B". If your bar graph has three components, it should be specified accordingly with three fields and named appropriately (e.g., "`=stacked; usr; sys; idle`" for the CPU figure)
b. "`title=`" specifies the title of the figure
c. "`colors=`" specifies the color for each component in the stacked bars. If your bar graph has three components, it should be specified accordingly with three fields (e.g., "`colors=yellow,red,black`")
d. "`xlabel=`" specifies the X label.

e. "`ylabel=`" specifies the Y label.
f. The block of numbers specifies the data. Each row is corresponding to one bar in the figure. For each row, the first field is the X data, which is shown on the X axis, the second field is the first data item (corresponding to "A"), and the third field is the second data item (corresponding to "B").

In your shell script, your code periodically run the tools `mpstat` and `iostat`, collect system usage information, extract data from the output, for a specified number of iterations, and at the end, create two plot files, and run the bargraph.pl to generate two `eps` files.

Download the example plot and `eps` files here:
http://www.csc.lsu.edu/~fchen/class/csc4103-sp17/PA3-example.plot
http://www.csc.lsu.edu/~fchen/class/csc4103-sp17/PA3-example.eps

More information about the bargraph script can be found at the following link:
http://www.burningcutlery.com/derek/bargraph/.

Before you submit:

(1) Compile and test-run your code on the classes.csc.lsu.edu server. Your code should be written in programming languages as specified above, and be compilable and runnable in the classes server's Linux environment. Windows code will NOT be accepted.

(2) Submit your work as instructed before the deadline and verify your submission, which will display your submission date/time.

(3) Late submissions will be penalized by 10% per day late and no more than 3 days late.

Submitting Your Work

All files you submit must have a header with the following:

Name:	Your Name (Last, First)
Project:	PA-3 (Shell Programming)
File:	filename
Instructor:	Feng Chen
Class:	cs4103-sp17
LogonID:	cs4103xx

You need to use the server "**classes.csc.lsu.edu**" to work on the assignment. You can login to your account in the server using SSH. Create a directory **prog3** (by typing **mkdir prog3)** in your home directory in which you create your program or source code.

Make sure that you are in the **prog3** directory while submitting your program. Submit your assignment to the grader by typing the following command:

~cs4103_chf/bin/p_copy 3

This command copies everything in your prog2 directory to the grader's account. Check whether if all required files have been submitted successfully:

~cs4103_chf/bin/verify 3

Reference

http://www.burningcutlery.com/derek/bargraph/.