CS 279 - Homework 4

Deadline

11:59 pm on **Friday, October 7**

Purpose

To get some more practice with bash test expression, to think some more about job control, and a bit more.

How to submit

You will complete **Problem 1** on the course Canvas site (giving you some more practice with `bash` test expressions), so that you can see if you are on the right track.

For the rest of the problems, you will create several files and then submit those to the course Canvas site.

NOTE: While I list the separate files you need to submit for each problem below, I am going to set up Canvas to *also* accept `.zip` files.

That is,

- you can submit each file to Canvas,
- OR, if you prefer, you may compress your files to be submitted into a single `.zip` file and submit that `.zip` file to Canvas.

Important notes

Assume, for all `bash` scripts in this course, that the following are required:

- Start each script with the line that is considered good style (and is a CS 279 course requirement), that specifies that this script should be executed using the `bash` shell
- After a blank line, put in one or more **comments** including at least the name of the shell script, your name, and its last modified date
- And follow these comments with a blank line.

Problem 1 - 10 points

Problem 1 is correctly answering the "HW 4 - Problem 1 - Short-answer questions on some test expressions" on the course Canvas site.

Problem 2

Consider the following output of a `jobs` command:

```
[1]   Running                 actions.sh &
[2]   Stopped                 emacs hw3-prob1.txt
[3]+  Stopped                 vi hw3-prob3.txt
[4]   Stopped                 ~st10/279submit
[5]-  Stopped                 nano brilliant.sh
[6]   Stopped                 doIt.sh
```

In a file named `hw4-prob2.txt`, put your name, and then your answers to each of the following.

2 part a

Assume that the above `jobs` command was just done. Give a command that would cause the `emacs` process shown above to become the foreground process. (Note: there is more than one correct answer!)

2 part b

Assume that the above `jobs` command was just done. What would become the foreground process as a result of the following command?

```
fg
```

2 part c

Assume that the above `jobs` command was just done. Write a command that would kill the `nano` process above. (Note: there is more than one correct answer!)

2 part d

Assume that the above `jobs` command was just done. Which of these jobs is the so-called current job, the one that was stopped most recently?

2 part e

Assume that the above `jobs` command was just done. What would become the foreground process as a result of the following command?

```
fg %do
```

Submit your resulting `hw4-prob2.txt`.

Problem 3

In a file named `hw4-prob3.txt`, put your name, and then your answers to each of the following.

3 part a

Assume that `do-things` is an executable bash script in the current directory that does not happen to need interactive input, nor does it happen to print to the screen, but its task does take a while to run.

Write a command you can type to start this script running as a background process.

3 part b

Write a command you can type to create a shell variable `COLOR_PREF`, giving it an initial value of your choice, such that any child shells *will* receive a copy of `COLOR_PREF`.

3 part c

For many Linux programs, while they are running in the foreground, what can you type to cause them to now start running in the background instead.

Submit your resulting `hw4-prob3.txt`.

Problem 4

FUN FACT: Here is an example of a bash `while` loop:

```
my_choice="yes"

while [ "$my_choice" == "yes" ]
do
```

```
    echo "You chose yes"
    echo "Enter your next choice:"
    read my_choice
done
echo "after loop: $my_choice"
```

Write a `bash` shell script `stubborn` or `stubborn.sh` that:

- tries to set a variable `given_file` to the first command line argument
- IF the resulting variable `given_file` does NOT contain the name of a file that exists in the current directory, your script should:
  - complain that the first command-line argument is NOT a file in the current directory
  - display a message saying here are options they can choose from, and show the current contents of the current working directory
  - while `given_file` does not contain the name of a file that exists in the current working directory, it asks the user to enter the name of a file in the current directory and reads in what they enter into variable `given_file`
- (So, when it gets here, variable `given_file` should have the name of a file that exists in the current directory!)
- It should call `ls` with options `-ld` for the value within variable `given_file`. (This will cause no harm if it contains a regular file, but if it contains a directory it will give the information for that directory itself rater than its contents.)
- If the file in `given_file` is a directory file, it prints a message saying that it is a directory
  - Otherwise, it gives the result of the `wc` command for the file in `given_file`.
- (If you would like to do more with variable `given_file`'s value before ending your script, please feel free to do so!)

Submit your resulting bash script `stubborn` or `stubborn.sh` .