CE 350: LABWORK 1 2011-2012 SPRING

1. # reference: http://tldp.org/LDP/abs/html/sha-bang.html

The #! line in a shell script will be the first thing the command interpreter (**sh** or **bash**) sees. Since this line begins with a #, it will be correctly interpreted as a comment when the command interpreter finally executes the script. The line has already served its purpose - calling the command interpreter.

If, in fact, the script includes an *extra* #! line, then **bash** will interpret it as a comment.

```
#!/bin/bash

echo "Part 1 of script."
a=1

#!/bin/bash
# This does *not* launch a new script.

echo "Part 2 of script."
echo $a  # Value of $a stays at 1.
```

2. # reference: http://tldp.org/LDP/abs/html/sha-bang.html

```
#!/bin/rm
# Self-deleting script.

# Nothing much seems to happen when you run this... except that the file
# disappears.

WHATEVER=85

echo "This line will never print (betcha!)."

exit $WHATEVER  # Doesn't matter. The script will not exit here.
                # Try an echo $? after script termination.
                # You'll get a 0, not a 85.
```

TO DO 1: Also, try starting a `README` file with a **`#!/bin/more`**, and making it executable.

3. # reference: http://tldp.org/LDP/abs/html/sha-bang.html

A script may begin with a #!/bin/env bash *sha-bang* line. This may be useful on UNIX machines where *bash* is not located in `/bin`

TO DO 1: Try to modify the script to be done in 2nd question as to include **`#!/bin/env more`**

TO DO 2: Try to modify the script given in 1st question to be run by different shells.

4. It is important to know that, once the script starts running it creates a child bash process. Check the following example:

```
#!/bin/bash

echo "print working directory: `pwd`"
echo "now changing directory to .."

cd ..

echo "print working directory: `pwd`"
echo "once the script ends, check your pwd!"
```

---

5. You can save a command output to a variable using backquotes around the command: `command`
The following example saves the output to a variable, and uses it again.

```
#!/bin/bash
echo "Please enter your name!"
echo "WAIT! I have changed my mind!"
echo "I actually now it! Is it `whoami`?"
echo "Let me save it!"
isim=`whoami`
echo "your name is now saved in \$isim: $isim"
```

TO DO 1: In the last line what does **`\$`** do? What is the difference between `\$isim` and `$isim` when used with an echo line?

---

6. Write a script which lists the contents of the working directory and its parent directory.