**Shell-Script Assignment**

Q1. Create a script that, when run, will display the following environment variables to the console:

USER

HOME

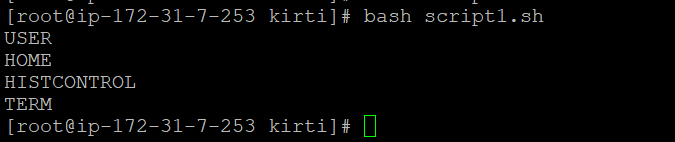
HISTCONTROL

TERM

**Code:**



**Output:**



Q2. Write a script that sets FOUR variables:

**MYUSERNAME**

**MYPASSWORD**

**STARTOFSCRIPT**

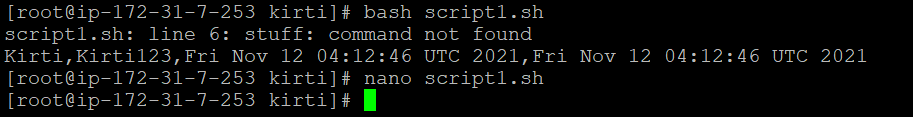
**ENDOFSCRIPT**

Populate the first two with some default value and use command redirection to set the third and fourth value to the date/time of when the script was started and completed. Within the script, be sure to display the values to the terminal when run.

**Code:**

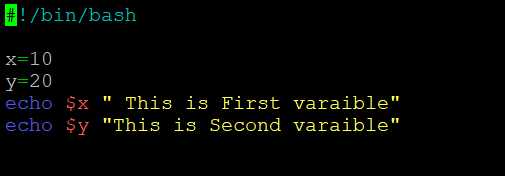


**Output:**

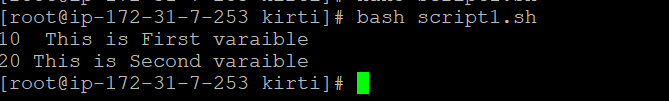


3. Develop a script that creates, sets and displays two variables to the terminal when run. Within this script, add comments to explain what the script is doing, what each variable is and, using inline comments, what each command is doing.

Code:



Output:



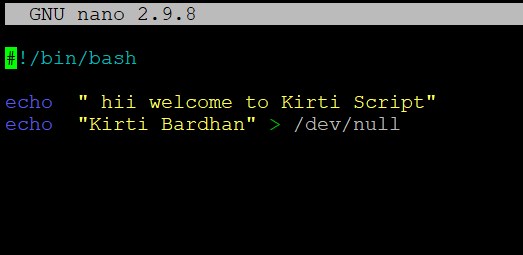
4. Create a simple script that does the following:

Echo a full sentence to the terminal

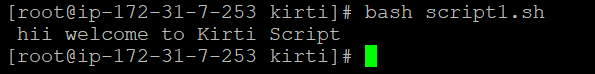
Echo a different full sentence, but redirect it to /dev/null

Run and display the results and make sure the statements appear where intended.

**Code:**



**Output:**



5. Write a script that runs three commands:

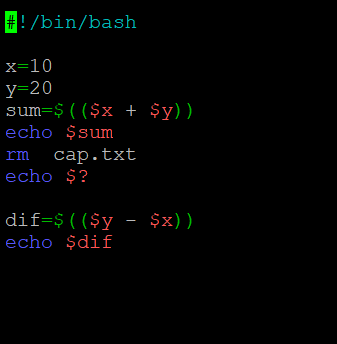
Evaluate an arithmetic expression

Attempt to remove a file that does not exist in the current directory

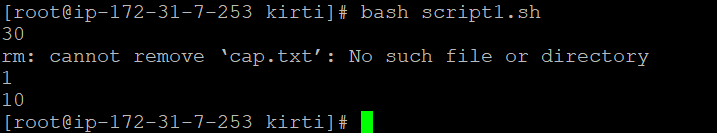
Evaluate another arithmetic expression

Immediately after each command, echo the exit status of that command to the terminal using the appropriate variable to show success and failure exit codes.

**Code:**



**Output:**



6. Write a script that evaluates and displays the following arithmetic operations:

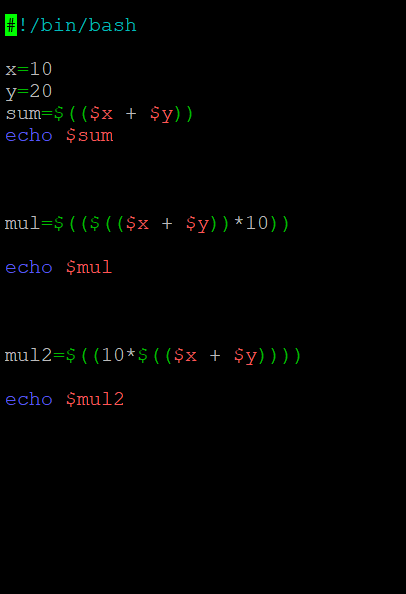
Add two numbers

Add two numbers and multiply by a third, do not group anything

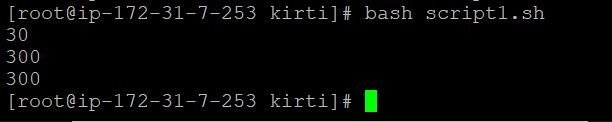
Add two numbers, grouped (changing order of precedence) and multiply by a third

Keep in mind special characters and/or escape characters as needed.

**Code:**



**Output:**



8. Create a script that interacts with the user. You will want to prompt the user to enter the following information (which you will capture and place into the following variables):

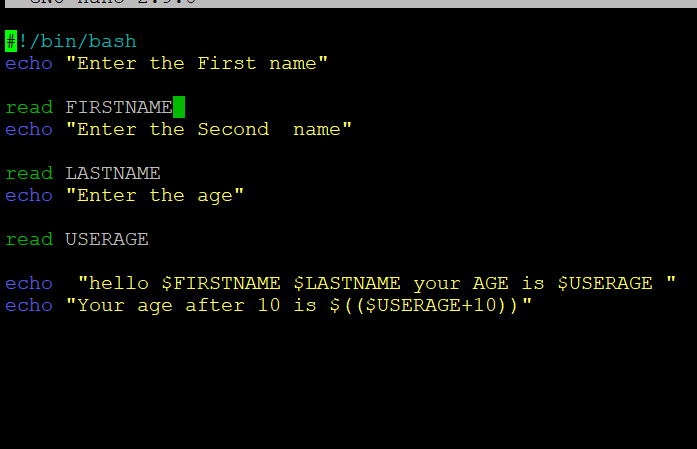
FIRSTNAME

LASTNAME

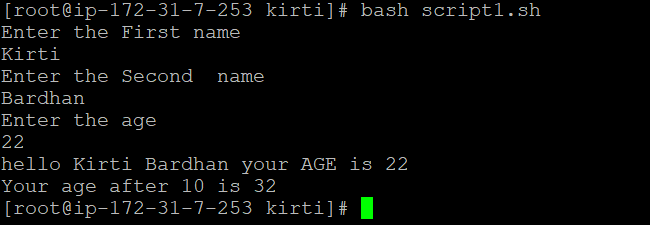
USERAGE

Greet the user with their name and current age displayed and then calculate and display their age in 10 years.

**Code:**

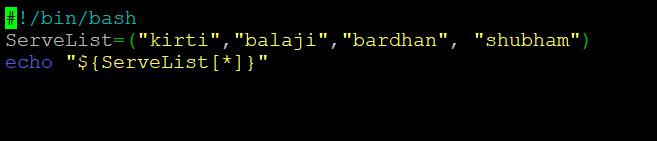


**Output:**

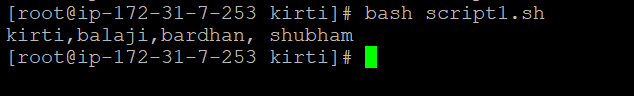


9. Write a script intended to iterate through an array called SERVERLIST containing at least four values (server names). Display all four values to the terminal when run.

**Code:**

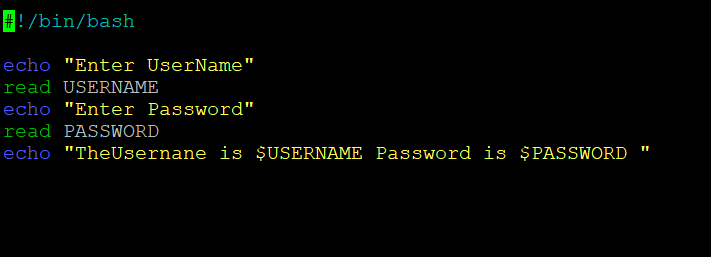


**Output:**

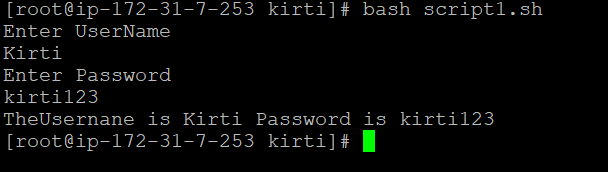


10. Create a script that, when run, will accept two command line values as arguments. These arguments should be a username and password and assigned to two variables in the script named appropriately. Finally, echo those values out to the terminal when run to indicate they were read and assigned as expected.

**Code:**



**Output:**



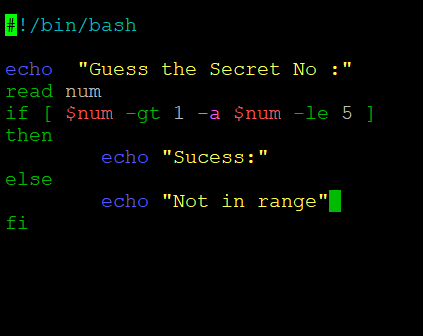
11. Create a script that interacts with the user. Prompt them to guess a secret number between 1 and 5.

Read the guess from the terminal and assign it to a variable.

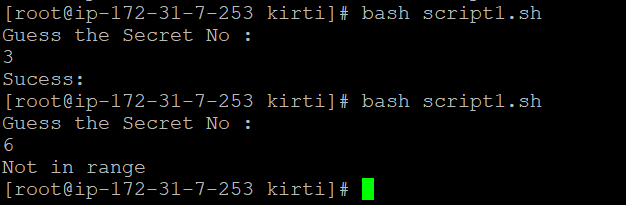
Using the 'if' statement from the course, test that value to determine if they guessed the right number (you choose the correct value).

If they do, display a success message, otherwise do nothing.

**Code:**

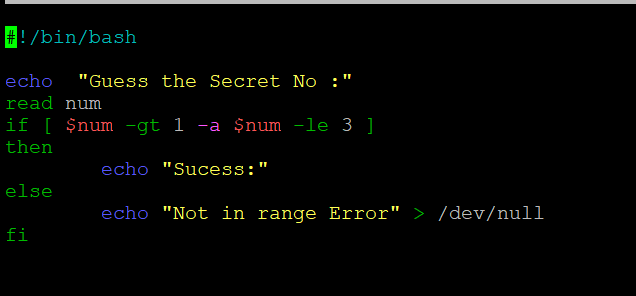


**Output:**

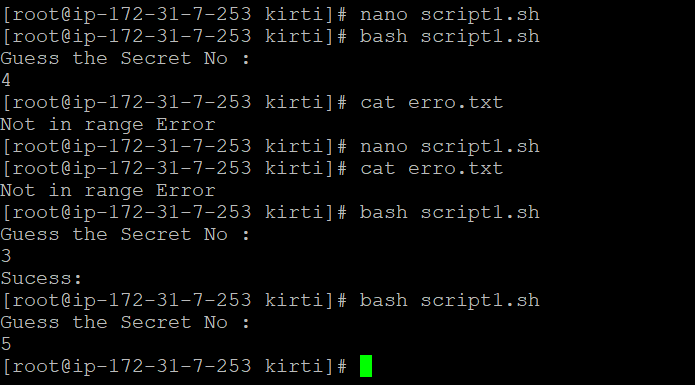


12. Write a script that will prompt the user to enter a number between 1 and 3. Capture that number in a variable and then test that variable. Be sure to display the variable and it's value as appropriate within an if/then/else statement where the final statement will display if they did not enter a number between 1 and 3 telling them they failed to follow instructions. Redirect errors from each of the tests to /dev/null (to prevent script errors from showing if text is entered for example).

**Code:**

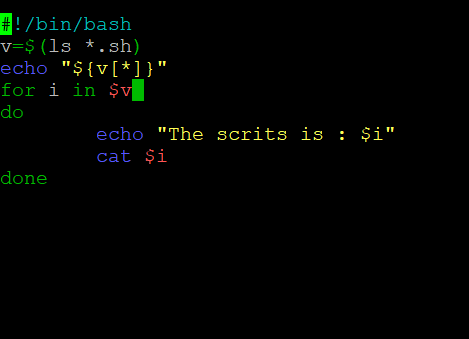


**Output:**

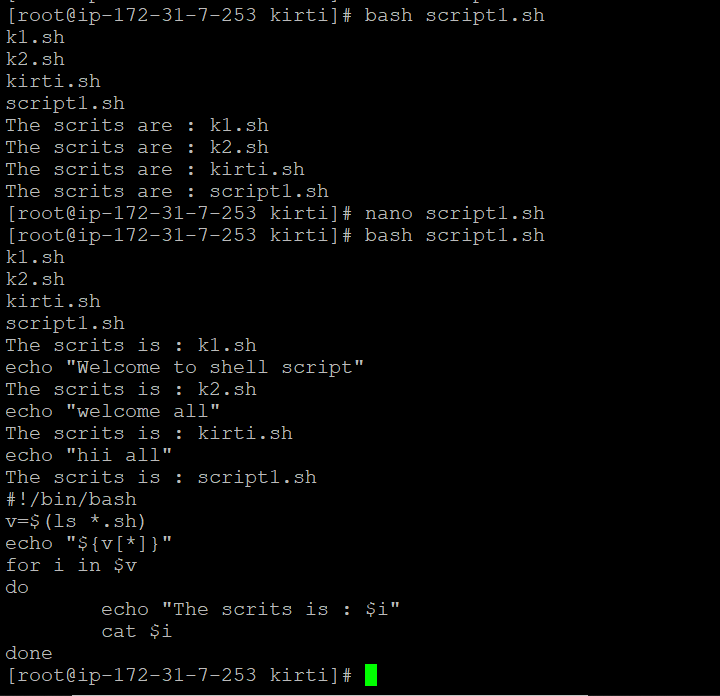


13. Write a script that assigns a variable that contains a list of all shell scripts (\*.sh) in the current directory (command redirection). Using the 'for'statement from the course, iterate through that list of files and display the filename of each and cat out the contents to the terminal.

**Code:**

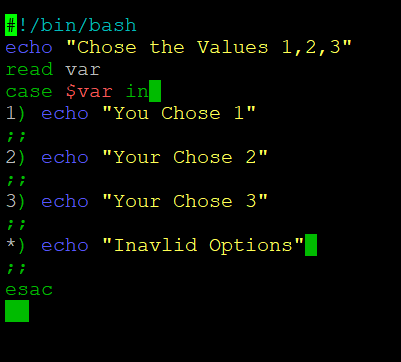


**Output:**

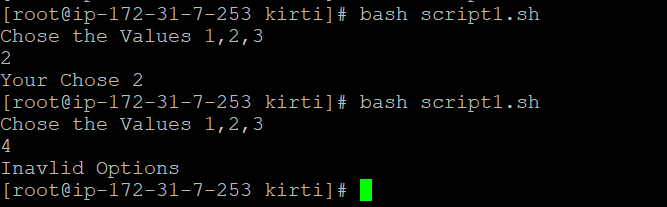


14. Develop a simple three item menu to display on the terminal. Your script, upon display of the menu, should prompt the user to choose one of the three available options. Using the 'case' statement from the course, display three unique messages depending on which number they chose, with a catch all message letting them know if they went outside the bounds of instructions.

**Code:**

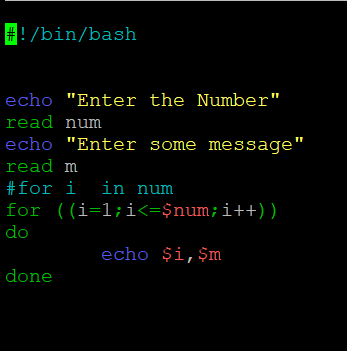


**Output:**



15. Create a script that prompts the user for a number. That number is to be used to display a simple message to the terminal X number of times (where X is the number captured from the user input). The message should include an indication of the number for each count the message is displayed.

**Code:**



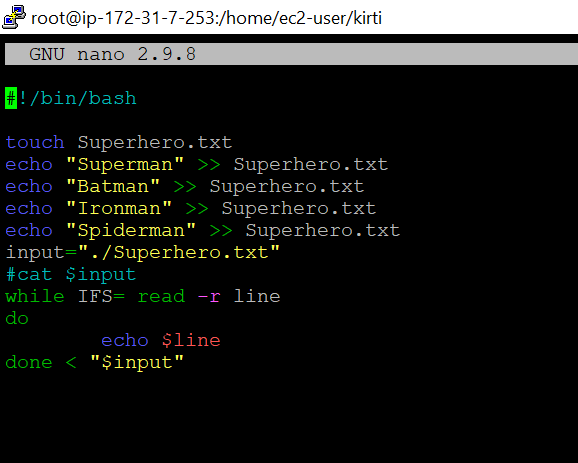
**Output:**



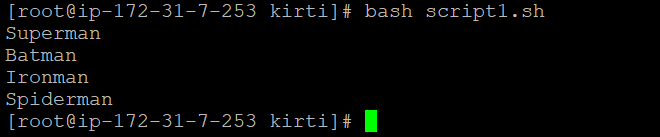
16. Create a simple text file containing a list of super heros (or some names if preferred), use at least four values, one per line in the file.

Write a bash shell script that then reads that file and displays it line by line on the terminal window.

**Code:**

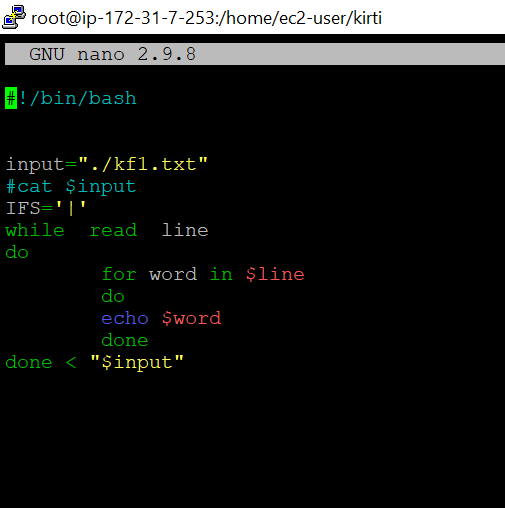


**Output:**

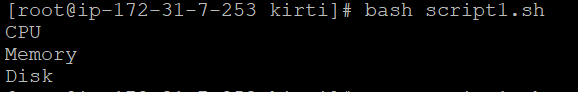


18. Create a simple text file at the command prompt. This file should contain three values - CPU, Memory and Disk space for an imaginary system, all on one line and delimited with a '|' character. Write a script to read that file and prompt the user for the delimiter value. Use that delimiter along with the IFS variable and read those delimited values into three appropriately named variables. Finally, display them with labels, one each per line.

**Code:**



**Output:**



20. Create a script that accepts a command line parameter intended to be the name of a directory.

With the script, attempt to change to the indicated directory,

be sure to redirect errors to /dev/null on the command as we will be providing our own messaging.

Test the results of the command to change directories.

If it was successful, indicate success and display the contents of the directory.

If it was not successful, indicate we cannot change directories and exit to the terminal with a custom exit code (less than 200).

**Code:**



**Output:**

