LINUX

Bash Scripting

TEJ4M

What is a BASH Script?

```
skickar@Dell-3 Nuke % which bash
/bin/bash
skickar@Dell-3 Nuke % ls
expect.exp      trigger.sh
skickar@Dell-3 Nuke % nano bash.sh
skickar@Dell-3 Nuke % bash bash.sh
Hackers love to learn on Null Byte
skickar@Dell-3 Nuke % nano bash.sh
skickar@Dell-3 Nuke % bash bash.sh "Vermin Supreme" "candidate" "president"
I firmly belive that Vermin Supreme is the candidate for the office of president
skickar@Dell-3 Nuke % nano bash.sh
skickar@Dell-3 Nuke % bash bash.sh
skickar
skickar@Dell-3 Nuke % nano bash.sh
skickar@Dell-3 Nuke % bash bash.sh
What is your name?
Kody
Wow, Kody sounds like a punk
skickar@Dell-3 Nuke % nano bash.sh
skickar@Dell-3 Nuke % bash bash.sh
What is your name?
Kody
bash.sh: line 9: syntax error: unexpected end of file
skickar@Dell-3 Nuke % nano bash.sh
skickar@Dell-3 Nuke % bash bash.sh
What is your name?
Kody
```

Automate tasks with Bash scripts

It's a series of commands written in a file and all executed sequentially when the file itself is executed.

How to identify?

A bash script file has the extension .sh.

Format of the File

All bash scripts start with the shebang:

```
#! /bin/bash
```

This is the absolute path to the bash interpreter that runs our commands. This path can vary but for the most part it's the same.

File Execution Rights

You can only execute a bash script if the user has execution rights to it. The x property of a file denotes it as having this right:

```
-rwxrw-rw- 1 zaira zaira   25 Mar 23 17:39 test_script.sh
```

TEJ4M

Create a Bash Script

Create the file using touch. Find your bash path and set it in the shebang. Add the commands. Set its execution rights. Finally run the script.

```
touch hello_world.sh
```

```
which bash
```

```
zaira@Zaira:~$ which bash
/usr/bin/bash
```

```
#! /usr/bin/bash
echo "Hello World"
```

```
chmod u+x hello_world.sh
```

```
./hello_world.sh
```

```
bash hello_world.sh
```

TEJ4M

OUTPUT

```
zaira@Zaira:~$ ./hello_world.sh
Hello World
zaira@Zaira:~$
zaira@Zaira:~$ bash hello_world.sh
Hello World
```

VARIABLES

We can define variables, similarly to python by just creating and assigning a value. We access the value stored in it using the $ operator.

main.sh ×

```
#!/bin/bash
# A simple variable example
greeting=Hello
name=Tux
echo $greeting $name

```

Console Shell

```
> bash main.sh
Hello Tux
>
```

ARITHMETIC OPERATIONS

Enclose all arithmetic operations in double quotes. Note the use of $ to access the value.

main.sh ×

```
#!/bin/bash

var=$((3+9))
echo $var

```

Console Shell

```
> bash main.sh
12
>
```

DECIMAL PLACES

Pipe the calculation to the bash calculator with the scale command to define the number of decimal places to show. In this example we show the result to 2 decimal places.

```
zaira@Zaira:~$ echo "scale=2;22/7" | bc
3.14
```

INPUT

To get input from a user use the read command

main.sh ×

```
#!/bin/bash

echo "Enter a numner"
read a

echo "Enter a numner"
read b

var=$((a+b))
echo $var

```

Console Shell

```
> bash main.sh
Enter a numner
9
Enter a numner
9
18
>
```

COMPARING NUMBERS AND STRINGS

OPERATION	SYNTAX	EXPLANATION
Equality	num1 -eq num2	is num1 equal to num2
Greater than equal to	num1 -ge num2	is num1 greater than equal to num2
Greater than	num1 -gt num2	is num1 greater than num2
Less than equal to	num1 -le num2	is num1 less than equal to num2
Less than	num1 -lt num2	is num1 less than num2
Not Equal to	num1 -ne num2	is num1 not equal to num2

For String comparisons use:
= equals to ?
== equals to ?
!= not equals to
> greater than
< less than

DECISIONS

The format for making simple decisions is a follows (note the spacing between the brackets and the condition):

```
if [ conditions ]
    then
        commands
fi
```

```
if [[ condition ]]
then
    statement
elif [[ condition ]]; then
    statement
else
    do this by default
fi
```

SAMPLE DECISION

main.sh ×

```bash
#!/bin/bash

read x
read y

if [ $x -gt $y ]
then
echo X is greater than Y
elif [ $x -lt $y ]
then
echo X is less than Y
elif [ $x -eq $y ]
then
echo X is equal to Y
fi
```

Console Shell

```
> bash main.sh
0
10
X is less than Y
>
```

LOGICAL OPERATORS

AND -a
OR -o

Example:
$a is greater than 40 AND $b is less than 6

```
if [ $a -gt 40 -a $b -lt 6 ]
```

SAMPLE

main.sh ×

```
#!/bin/bash

read a
read b
read c

if [ $a == $b -a $b == $c -a $a
== $c ]
then
echo EQUILATERAL

elif [ $a == $b -o $b == $c -o $a
 == $c ]
then
echo ISOSCELES
else
echo SCALENE

fi
```

Console Shell

```
> bash main.sh
3
3
3
EQUILATERAL
>
```

TEJ4M

TEST CASE #2

main.sh ×

```
#!/bin/bash

read a
read b
read c

if [ $a == $b -a $b == $c -a $a == $c ]
then
echo EQUILATERAL

elif [ $a == $b -o $b == $c -o $a == $c ]
then
echo ISOSCELES
else
echo SCALENE

fi
```

Console Shell

```
> bash main.sh
2
2
3
ISOSCELES
>
```

TEJ4M

TEST CASE #3

main.sh ×

```
#!/bin/bash

read a
read b
read c

if [ $a == $b -a $b == $c -a $a == $c ]
then
echo EQUILATERAL

elif [ $a == $b -o $b == $c -o $a == $c ]
then
echo ISOSCELES
else
echo SCALENE

fi
```

Console Shell

```
> bash main.sh
4
3
9
SCALENE
>
```

TEJ4M

QUESTIONS AND EXERCISES

Create and test a bash script to ask for a user's name and then greet that user.

QUESTIONS AND EXERCISES

Create a bash script to ask someone for a number and then display the square of that number.

QUESTIONS AND EXERCISES

Create a bash script to ask a user for the diameter of a cylinder and then displays the volume of that cylinder (don't worry about accuracy with decimals).

QUESTIONS AND EXERCISES

Create a bash script to ask a user for the name of a file to create and then creates that file and then displays the contents of the current directory to show the file created.

QUESTIONS AND EXERCISES

Create a bash script to ask someone for their age and respond as follows:

-if they are less than 13 tell them that they're a kid

-if they are between 13 and 19 they are a teenager

-if they are over 19 they are an adult unless over 64 then they're an elder

QUESTIONS AND EXERCISES

Using this ask someone to enter the names of two cities and tell them if they entered two different cities or the same one.

QUESTIONS AND EXERCISES

Create a bash script to ask for the names of three files. After getting these names create the 3 files. Add 3 lines of random data to each file and then display the contents of each file one after another. Note the sleep command adds in a delay between lines of execution i.e. sleep 1 waits 1 second and so use this to delay displays of each file.

QUESTIONS AND EXERCISES

Create a bash script to ask for a number and then tell the user if its positive or negative. If positive add the numbers to a positives.txt file otherwise add it to a negatives.txt file.

QUESTIONS AND EXERCISES

```bash
#!/bin/bash
# Shebang line: Indicates the path to the shell

# Prompt the user to enter a number
echo "Input a number:"
read n

# Check if the number is greater than 100
if [ "$n" -gt 100 ]; then
    echo "The number is greater than 100."
else
    echo "The number is not greater than 100."
fi
```

Explain what the following bash script does. Assume someone enters 120, write out how the output would look.:

QUESTIONS AND EXERCISES

```
#!/bin/bash
# Shebang line: Indicates the p


# Check if the file "test.txt"
if [ -f "test.txt" ]; then
    echo "File exists"
else
    echo "File does not exist"
fi
```

Run, test and explain this code.

QUESTIONS AND EXERCISES

Create a program that asks the user for 3 numbers. Tell them if the first two numbers match and the last number is equal to 2, otherwise tell then that the first two numbers don't match and that the last number is not 2. Test with inputs 1,1,2 and 2,3,4. Try testing with 1,1,3 and 1,2,2.

```bash
#!/bin/bash

# Question 1: Ask for user's name and greet them
read -p "Enter your name: " name
echo "Hello, $name! Welcome."

# Question 2: Ask for a number and display its square
read -p "Enter a number: " num
square=$((num * num))
echo "The square of $num is $square"

# Question 3: Ask for diameter and calculate volume of cylinder (approximate)
read -p "Enter diameter of cylinder: " d
read -p "Enter height of cylinder: " h
r=$((d / 2))
volume=$((3 * r * r * h))
echo "The approximate volume of the cylinder is $volume"

# Question 4: Create a file and display directory contents
read -p "Enter filename to create: " fname
touch "$fname"
echo "File '$fname' created."
echo "Current directory contents:"
ls

# Question 5: Age classification
read -p "Enter your age: " age
if [ $age -lt 13 ]; then
  echo "You're a kid."
elif [ $age -ge 13 ] && [ $age -le 19 ]; then
  echo "You're a teenager."\elif [ $age -gt 64 ]; then
  echo "You're an elder."
else
  echo "You're an adult."
fi

# Question 6: Compare two cities
read -p "Enter the first city: " city1
read -p "Enter the second city: " city2
if [ "$city1" = "$city2" ]; then
else
  echo "You entered two different cities."
fi

# Question 7: Create 3 files, write and display data
read -p "Enter name for file1: " f1
read -p "Enter name for file2: " f2
read -p "Enter name for file3: " f3

touch $f1 $f2 $f3
echo "Line 1" > $f1
echo "Line 2" >> $f1
echo "Line 3" >> $f1

echo "Line A" > $f2
echo "Line B" >> $f2
echo "Line C" >> $f2

echo "X1" > $f3
echo "X2" >> $f3
echo "X3" >> $f3

sleep 1
cat $f1
sleep 1
cat $f2
sleep 1
cat $f3

# Question 8: Add number to positives.txt or negatives.txt
read -p "Enter a number: " n
if [ $n -ge 0 ]; then
  echo $n >> positives.txt
  echo "$n is positive."
else
  echo $n >> negatives.txt
  echo "$n is negative."
fi

# Question 9: Explain the script assuming input is 120
# Example script:
# read -p "Enter number: " n
# if [ $n -gt 100 ]; then echo "High"; fi
# OUTPUT: High (because 120 > 100)

# Question 10: Run and explain (assume simple if-else conditions and outputs, not shown here)

# Question 11: Compare numbers
read -p "Enter number 1: " n1
read -p "Enter number 2: " n2
read -p "Enter number 3: " n3
if [ $n1 -eq $n2 ] && [ $n3 -eq 2 ]; then
  echo "First two match and third is 2."
else
  echo "Mismatch or third is not 2."
fi
```