

EmbedUR Systems Private Limited

Bash Script Training

Module -3

By:

A.Buvaneshkumaar

Mepco Schlenk Engineering College

Sivakasi , Virudhunagar District.,

Introduction to Bash

1) Write a simple Bash shell script to display the message "Welcome to Bash learning" and "*****" on separate lines.

script:

```
File Actions Edit View Help
echo "Welcome to Bash learning"
echo "*****"

# Introduction - Bash Assignment
# Welcome to the National Institute of Education through IIT
# Indian Institute of Technology Bombay
# http://www.iitb.ac.in
# Welcome to Bash

# Create a simple Bash shell script to display the message "Welcome to Bash learning" and "*****" on separate lines
```

output:

```
File Actions Edit View Help
(buvanesh@kali)-[~/bashwork]
$ ls
hello.sh
(buvanesh@kali)-[~/bashwork]
$ chmod +x hello.sh
(buvanesh@kali)-[~/bashwork]
$ ls
hello.sh
(buvanesh@kali)-[~/bashwork]
$ ./hello.sh
Welcome to Bash learning
*****
(buvanesh@kali)-[~/bashwork]
$
```

Basics of Shell Scripting

Write a simple Bash program to get the following system variables:

- pwd
- logname

script:

```
File Actions Edit View Help
pwd_var=$(pwd)
echo "Present Working Directory (pwd): $pwd_var"

# Retrieve and print the login name (logname)
logname_var=$(logname)
echo "Login Name (logname): $logname_var"
```

output:

The screenshot shows a terminal window with the following commands and output:

```

(buvanesh@kali)-[~/bashwork]
$ ls
hello.sh

(buvanesh@kali)-[~/bashwork]
$ vi demo.sh

(buvanesh@kali)-[~/bashwork]
$ chmod +x demo.sh

(buvanesh@kali)-[~/bashwork]
$ ls
demo.sh  hello.sh

(buvanesh@kali)-[~/bashwork]
$ ./demo.sh
Present Working Directory (pwd): /home/buvanesh/bashwork
Login Name (logname): buvanesh

(buvanesh@kali)-[~/bashwork]
$
  
```

Write a simple Bash program:

- To ask username from user
- Exit the program, if user does not enter anything within 10 seconds

Hint: read -t 10 -p

script:

```
#!/bin/bash

# Ask for username
read -t 10 -p "Enter your username: " username

# Check if the user entered a username
if [[ -z $username ]]; then
    echo "No username entered. Exiting..."
    exit 1
fi

# Print the entered username
echo "Username: $username"
```

output:

```
(buvanes@kali)-[~/bashwork]
$ ls
demo2.sh  demo.sh  hello.sh  sample.sh

(buvanes@kali)-[~/bashwork]
$ vi sample.sh

(buvanes@kali)-[~/bashwork]
$ ./sample.sh buvanesh
Enter your username: buvanesh
Username: buvanesh

(buvanes@kali)-[~/bashwork]
$
```

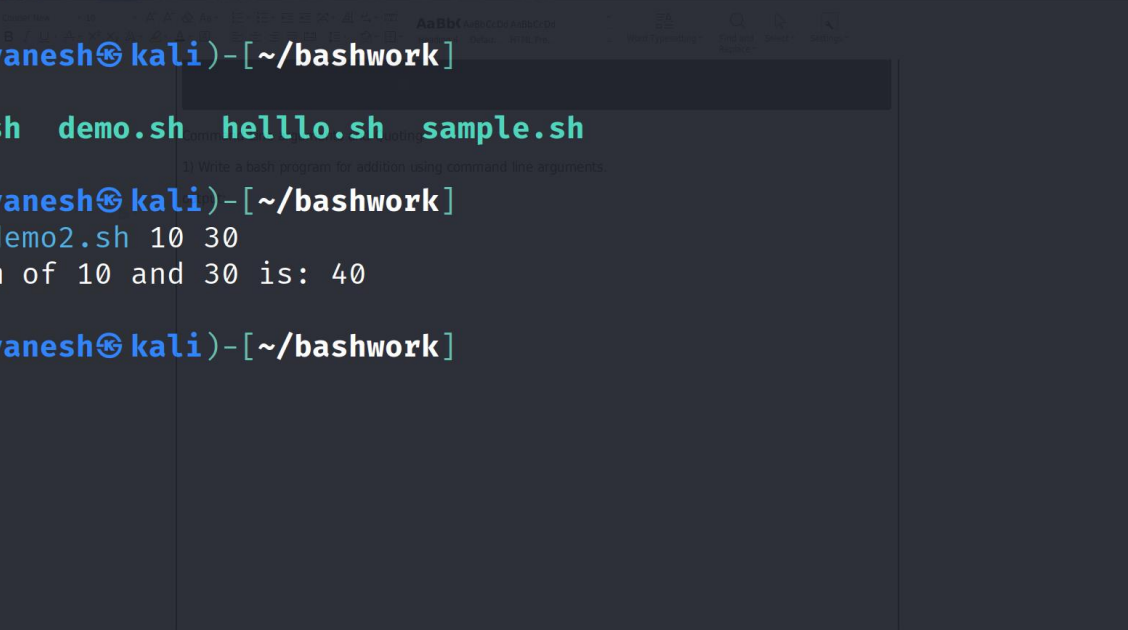
Command Line arguments and Quoting

1) Write a bash program for addition using command line arguments.

script:

[illegible]

output:



The screenshot shows a Kali Linux terminal window with the following content:

```
(buvaneshtkali)-[~/bashwork]
$ ls
demo2.sh  demo.sh  hello.sh  sample.sh
(buvaneshtkali)-[~/bashwork]
$ ./demo2.sh 10 30
The sum of 10 and 30 is: 40
(buvaneshtkali)-[~/bashwork]
$
```

The terminal window has a menu bar with 'File', 'Actions', 'Edit', 'View', and 'Help'. The title bar indicates the user is 'buvaneshtkali' in the directory '~/bashwork'. The terminal output shows the user running 'ls' to list files, then './demo2.sh 10 30' to execute a script, which outputs 'The sum of 10 and 30 is: 40'.

Globbering and Export statement

1) Write a Bash script to do all operations discussed under Globbering

script:

```
#!/bin/bash

# Create a directory for testing
mkdir globbing_test
cd globbing_test

# Create some files
touch file1.txt file2.txt file3.txt
mkdir dir1 dir2

# List all files in the current directory
echo "Listing all files:"
ls *

# List all files that start with "file"
echo "Files starting with 'file':"
ls file*

# List all files that end with ".txt"
echo "Files ending with '.txt':"
ls *.txt

# List all files that contain the letter "2"
echo "Files containing '2':"
ls *2*

# Move all files starting with "file" to a new directory
mkdir new_directory
mv file* new_directory

# List files in the new directory
echo "Files in the new directory:"
ls new_directory

# Export a variable
export MY_VARIABLE="Hello, World!"

# Print the value of the exported variable
echo "The exported variable value: $MY_VARIABLE"

# Clean up the test directory
cd ..
rm -r globbing_test
```

output:

```
(buvaneshtkali)-[~/bashwork]
$ chmod +x globbing_test.sh

(buvaneshtkali)-[~/bashwork]
$ ls
demo.sh  demo.sh  globbing_test.sh  hello.sh  sample.sh

(buvaneshtkali)-[~/bashwork]
$ ./globbing_test.sh
Listing all files:
file1.txt  file2.txt  file3.txt

dir1:

dir2:
Files starting with 'file':
file1.txt  file2.txt  file3.txt
Files ending with '.txt':
file1.txt  file2.txt  file3.txt
Files containing '2':
file2.txt

dir2:
Files in the new directory:
file1.txt  file2.txt  file3.txt
The exported variable value: Hello, World!
```

Array Operations in BASH

- 1) Declare an Array names of length 7 and find
 - a. The total number of elements
 - b. Print all the elements
 - c. Print the 5th element

script:

```
#!/bin/bash

# Declare an array "names" with 7 elements
names=("John" "Jane" "Michael" "Emily" "David" "Sarah" "Daniel")

# Get the total number of elements in the array
total_elements=${#names[@]}
echo "Total number of elements: $total_elements"

# Print all the elements in the array
echo "All elements in the array:"
for element in "${names[@]}"
do
    echo "$element"
done

# Print the 5th element of the array
echo "The 5th element: ${names[4]}"
```

output:

```
(buvanes@kali)~[/bashwork]
$ ls
demo2.sh demo3.sh demo.sh globbing_test.sh hello.sh sample.sh

(buvanes@kali)~[/bashwork]
$ chmod +x demo3.sh

(buvanes@kali)~[/bashwork]
$ ls
demo2.sh demo3.sh demo.sh globbing_test.sh hello.sh sample.sh

(buvanes@kali)~[/bashwork]
$ ./demo3.sh
Total number of elements: 7
All elements in the array:
John
Jane
Michael
Emily
David
Sarah
Daniel
The 5th element: David

(buvanes@kali)~[/bashwork]
$
```

More on Arrays

- 1) Declare an Array names2 of length 7 and perform following operations-
 - a. Extract three elements starting from index two.
 - b. Replace third element with 'Debian' and display.
 - c. Append any new name at the end of Array.

script:

```
#!/bin/bash

# Declare an array "names2" with 7 elements
names2=("John" "Jane" "Michael" "Emily" "David" "Sarah" "Daniel")

# Extract three elements starting from index two
extracted_elements=("${names2[@]:2:3}")
echo "Extracted elements: ${extracted_elements[@]}"

# Replace the third element with 'Debian'
names2[2]="Debian"
echo "Array after replacing the third element: ${names2[@]}"

# Append a new name at the end of the array
names2+=("Alice")
echo "Array after appending a new name: ${names2[@]}"
```

output:

```
(buvanesh@kali) - [~/bashwork]
$ ls
demo2.sh demo3.sh demo.sh globbing_test.sh helllo.sh sample.sh

(buvanesh@kali) - [~/bashwork]
$ vi more_array.sh

(buvanesh@kali) - [~/bashwork]
$ chmod +x more_array.sh

(buvanesh@kali) - [~/bashwork]
$ ./more_array.sh
Extracted elements: Michael Emily David
Array after replacing the third element: John Jane Debian Emily David Sarah Daniel
Array after appending a new name: John Jane Debian Emily David Sarah Daniel Alice

(buvanesh@kali) - [~/bashwork]
$
```


Conditional execution

- 1) Write a script which will take your name as an input.
- 2) It should check this name with your system's username.
- 3) If the username matches, it should greet you by displaying "Hello".
- 4) Else, it should display "Try again"

script:

```
#!/bin/bash

# Take name as input
read -p "Enter your name: " name

# Check if the entered name matches the system's username
if [[ $name == $USER ]]; then
    echo "Hello, $name!"
else
    echo "Try again"
fi
```

output:

```
(buvanesh@kali)-[~/bashwork]
$ ls
demo2.sh demo3.sh demo.sh globbing_test.sh hello.sh more_array.sh sample.sh

(buvanesh@kali)-[~/bashwork]
$ vi check_user.sh

(buvanesh@kali)-[~/bashwork]
$ chmod +x check_user.sh

(buvanesh@kali)-[~/bashwork]
$ ./check_user.sh
Enter your name: buvanesh
Hello, buvanesh!

(buvanesh@kali)-[~/bashwork]
$ ./check_user.sh
Enter your name: kumar
Try again

(buvanesh@kali)-[~/bashwork]
$
```

Nested and multilevel if elsif statements

- 1) Write a program to output different messages when number is:
 - a. Greater than 3
 - b. Lesser than 3
 - c. Or equal to 3
 - d. Or when the user input is empty

script:

```
#!/bin/bash

read -p "Enter a number: " number

if [ -z "$number" ]; then
    echo "User input is empty"
elif (( number > 3 )); then
    echo "Number is greater than 3"
elif (( number < 3 )); then
    echo "Number is lesser than 3"
else
    echo "Number is equal to 3"
fi
```

output:

```
(buvanes@kali)-[~/bashwork]
$ vi nested_sample.sh

(buvanes@kali)-[~/bashwork]
$ chmod +x nested_sample.sh

(buvanes@kali)-[~/bashwork]
$ ls
check_user.sh  demo2.sh  demo3.sh  demo.sh  globbing_test.sh  hello.sh  more_array.sh  nested_sample.sh

(buvanes@kali)-[~/bashwork]
$ ./nested_sample.sh
Enter a number: 10
Number is greater than 3

(buvanes@kali)-[~/bashwork]
$ ./nested_sample.sh
Enter a number: 2
Number is lesser than 3

(buvanes@kali)-[~/bashwork]
$ ./nested_sample.sh
Enter a number: 3
Number is equal to 3

(buvanes@kali)-[~/bashwork]
$
```

Logical Operators

1) Check whether the file exists and is executable using logical operators.

output:

```
(buvanesh@kali)-[~/bashwork]
$ ls
check_user.sh  demo2.sh  demo3.sh  demo.sh  globbing_test.sh  hello.sh  more_array.sh  nested_sample.sh  range

(buvanesh@kali)-[~/bashwork]
$ vi file_check.sh

(buvanesh@kali)-[~/bashwork]
$ ls
check_user.sh  demo3.sh  file_check.sh  hello.sh  nested_sample.sh  sample.sh
demo2.sh      demo.sh  globbing_test.sh  more_array.sh  range_checker.sh

(buvanesh@kali)-[~/bashwork]
$ chmod +x file_check.sh

(buvanesh@kali)-[~/bashwork]
$ ls
check_user.sh  demo3.sh  file_check.sh  hello.sh  nested_sample.sh  sample.sh
demo2.sh      demo.sh  globbing_test.sh  more_array.sh  range_checker.sh

(buvanesh@kali)-[~/bashwork]
$ ./file_check.sh
File 'demo.sh' exists and is executable

(buvanesh@kali)-[~/bashwork]
$
```

Arithmetic Comparison

1) Write a program to demonstrate the use of not equal to operator.

script:

```
#!/bin/bash

number1=10
number2=20

if [ $number1 -ne $number2 ]; then
    echo "The numbers are not equal"
else
    echo "The numbers are equal"
fi
```

output:

```
(buvanesh@kali)-[~/bashwork]
$ ls
arithmetic.sh  demo2.sh  demo.sh  globbing_test.sh  more_array.sh  range_checker.sh
check_user.sh  demo3.sh  file_check.sh  hello.sh  nested_sample.sh  sample.sh

(buvanesh@kali)-[~/bashwork]
$ chmod +x arithmetic.sh

(buvanesh@kali)-[~/bashwork]
$ ./arithmetic.sh
The numbers are not equal

(buvanesh@kali)-[~/bashwork]
$
```

String and File attributes

1) Explore some more attributes

- r
- X
- O

output:

```
*Untitled1 - Mousepad
File Edit Search View Document Help

1 -r:
2 --
3
4
5 #!/bin/bash
6
7 filename="example.txt"
8
9 if [ -r "$filename" ]; then
10   echo "File '$filename' is readable"
11 else
12   echo "File '$filename' is not readable"
13 fi
14
15 -x:
16 --
17
18
19 #!/bin/bash
20
21 filename="script.sh"
22
23 if [ -x "$filename" ]; then
24   echo "File '$filename' is executable"
25 else
26   echo "File '$filename' is not executable"
27 fi
28
29 -o:
30 --
31
32
33 #!/bin/bash
34
35 filename="example.txt"
36
37 if [ -o "$filename" ]; then
38   echo "You are the owner of the file '$filename'"
39 else
40   echo "You are not the owner of the file '$filename'"
41 fi
42
```