Shell Scripting

What is Kernel?

The kernel is a computer program that is the core of a computer's operating system, with complete control over everything in the system.

What is Shell?

A shell is a special user program that provides an interface for the user to use operating system services. Shell accepts human-readable commands from users and converts them into something which the kernel can understand. It is a command language interpreter that executes commands read from input devices such as keyboards or from files. The shell gets started when the user logs in or starts the terminal.

What is Linux Shell Scripting for DevOps?

Shell Scripting is a way to automate tasks in a linux operating system. It is a sequence of code or text that contains commands as a input from users and execute them. Shell Scripting is a program to write a series of commands for the shell to execute.A shell script is usually created for command sequences in which a user has a need to use repeatedly in order to save time.

What is #!/bin/bash?

#!/bin/bash is the first line of the script and is called **Shebang**. Shebang tells the shell to execute it via bash shell. Shebang is simply an absolute path to the bash interpreter.

Can we write #!/bin/sh as well?

Yes, we can use #!/bin/sh to specify the path to the Bourne Shell (sh) interpreter.

The Bourne Shell is another commonly used shell in Unix-based operating systems.

Write a Shell Script that prints "I will complete #90DaysofDeVops challenge"

```
#! /bin/bash

echo " Its DevOps Bootcamp Course"
```

Write a Shell Script to take user input, input from arguments, and print the variables.

```
#! /bin/bash
echo “Enter your name”
read name
echo “Your name is $name”
```

Write an Example of If else in Shell Scripting by comparing two numbers

```
#! /bin/bash
echo “Enter first number”
read a
echo “Enter second number”
read b
if [ $a –gt $b ]
then
        echo “ a is greater than b”
else
        echo “ b is greater than a”
fi
```

Using While Loop:

Create a bash file with the name, **'while_example.sh',** to know the use of **while** loop. In the example, **while** loop will iterate for **5** times. The value of **count** variable will increment by **1** in each step. When the value of **count** variable will 5 then the **while** loop will terminate.

```
#!/bin/bash
valid=true
count=1
while [ $valid ]
do
 echo $count
if [ $count -eq 5 ];
then
break
fi
((count++))
done
```

```
ubuntu@ubuntu-VirtualBox:~/code$ bash while_example.sh
1
2
3
4
5
ubuntu@ubuntu-VirtualBox:~/code$
```

Using if statement with AND logic:

Different types of logical conditions can be used in if statement with two or more conditions. How you can define multiple conditions in if statement using **AND** logic is shown in the following example. **'&&'** is used to apply **AND** logic of **if** statement. Create a file named **'if_with_AND.sh'** to check the following code. Here, the value of **username** and **password** variables will be taken from the user and compared with '**admin**' and '**secret**'. If both values match then the output will be "**valid user**", otherwise the output will be "**invalid user**".

```
#! /bin/bash

echo “Enter Username”
read username
echo “Enter Password”
read password

if [[  ( $username == “admin” && $password == “admin123” ) ]]
echo “Valid User”
else
echo “invalid user”
fi
```

```
ubuntu@ubuntu-VirtualBox:~/code$ bash if_with_AND.sh
Enter username
admin
Enter password
1234
invalid user
ubuntu@ubuntu-VirtualBox:~/code$ bash if_with_AND.sh
Enter username
admin
Enter password
secret
valid user
ubuntu@ubuntu-VirtualBox:~/code$
```