Shell Scripting

Tasks

**Explain in your own words and examples, what is Shell Scripting for DevOps.**

Shell scripting is an interaction between the user and the operating system to perform certain actions. Like any other programming language, it has its own syntax and keywords to instruct the kernel to perform action.

Shell scripting is a common practise in DevOps role to automate almost most of the task, it is used for generating update, backup, cron jobs scripts. Shell scripting is a pre-requisite of getting into devops.

**What is #!/bin/bash? can we write #!/bin/sh as well?**

**#!/bin/bash** : is also called **shebang**  also called interpreter directive, which is written at the beginning of any shell script to specify the bash interpreter to use, if shebang line is not defined it uses the default interpreter of the kernel.

We can absolutely use **#!/bin/sh** as well. When we use **#!/bin/sh,** it specifies that the system's default Bourne Shell (or a compatible shell) should be used as the interpreter. This is more generic and ensures that the script will work on systems where Bash is not the default shell but where a Bourne Shell-compatible interpreter is available.

In many Unix-like systems, **/bin/sh** is a symbolic link to a specific shell, which is often a Bourne Shell-compatible shell. However, the behavior of /bin/sh can vary between different systems. Some systems link **/bin/sh** to a minimal POSIX-compatible shell, which may not support all the advanced features of Bash.

If you write your script to be compatible with **/bin/sh,** it should work on a wider range of Unix-like systems. However, you won't be able to use features specific to Bash that are not available in a standard Bourne Shell.

Write a Shell Script which prints I will complete #90DaysOofDevOps challenge.

*~ touch 90Days.sh*

*~ chmod 700 90Days.sh*

*~ vi 90Days.sh. # VIM editor open, press ESC+i*

*#! bin/bash*

*~ echo “I will complete the 90DaysOfDevOps challenge”. # press esc+:wq!*

*~./90Days.sh*

*~ I will complete the 90DaysOfDevOps challenge*

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

*ubuntu@ip-172-31-21-173:~$ vi user\_input.sh*

*ubuntu@ip-172-31-21-173:~$ chmod +x user\_input.sh*

*ubuntu@ip-172-31-21-173:~$ ./user\_input.sh*

*Please enter your name:kamran*

*My name is Kamran*

*ubuntu@ip-172-31-21-173:~$ cat user\_input.sh*

*#! bin/bash*

***read -p "Please enter your name:" NAME***

***echo "My name is $NAME"***

**Write an Example of If else in Shell Scripting by comparing 2 numbers.**

*ubuntu@ip-172-31-21-173:~$ vi compair.sh*

*ubuntu@ip-172-31-21-173:~$ chmod +x compair.sh*

*ubuntu@ip-172-31-21-173:~$ vi compair.sh*

*#! bin/bash*

*read -p "Enter the first number : " NUM1*

*read -p "Enter the second number : " NUM2*

*if [ "$NUM1" -gt "$NUM2" ]*

*then*

*echo "First number:$NUM1 is greater than second number:$NUM2"*

*else*

*echo "FirstFirst number:$NUM1 is smaller than second number:$NUM2"*

*fi*

*~*

*~*

*~*

*"compair.sh" 9L, 255B*

*ubuntu@ip-172-31-21-173:~$ ./compair.sh*

***Enter the first number : 4***

***Enter the second number : 3***

***First number:4 is greater than second number:3***