**Shell Scripting**

**Introduction to Bash Scripting**

Bash scripting is writing scripts using the Bash shell to automate tasks on Unix-based systems. A script is a series of commands executed by the shell.

**Types:**

sh

**Bash – higher version for sh**

Ksh – IBM – AIX

Csh – c program shell

Tcsh – turbo c shell

### Sample Script

A sample script is a basic example that demonstrates the structure and syntax of a Bash script.

vi sample.sh

#!/bin/bash

echo "Hello\_world"

sh sample.sh

./sample.sh

bash -x sample.sh

### Variables & System Variables

Variables in Bash store data that can be used and manipulated. System variables are predefined variables set by the system.

vi var.sh

#!/bin/bash

student=abu

echo $student

echo "$student is a clas topper"

echo "$student plays cricket"

chmod +x var.sh

sh var.sh

**System variables:**

#env

**Command Line Arguments**

Command line arguments are inputs provided to the script when it is executed, allowing the script to use these inputs.

vi var.sh

*#!/bin/bash*

*student=abu*

*student2=abi*

*echo $student*

*echo "$student is a clas topper"*

*echo "$student plays cricket"*

*echo $student2*

*echo "$2 is a rock star"*

*echo "$student2 plays guitar"*

*echo $1*

*echo "The user $2 is a boxer"*

./var.sh prem miketyson

**Loops**

Loops in Bash scripting allow repetitive execution of commands until a certain condition is met.

**While loop**

While loops execute until the condition is true

**Syntax:**

counter

while args

do

statements

done

Example:

vi while.sh

#!/bin/bash

n=1

while [ $n -le 10 ]

do

echo "$n"

n=$(( n+1 ))

sleep 1

done

./while.sh

**Operators:**

Arithmetic Operators

Relational Operators

Boolean Operators

String Operators == , !=

Arithmetic Operators:

+, -, \*, /, %

#!/bin/bash

val1=10

val2=40

expr $val1 + $val2

Relational Operators (compare values)

gt (greater than)

lt (less than)

ge (greater than and equal)

le (less then and equal)

eq (equals)

ne (not equals)

**Boolean Operators:**

AND

if [ $v1 -lt $v2 -a $v3 -gt $v2 ]

OR

if [ $v1 -lt $v2 -o $v3 -gt $v2 ]

NOT

if ![ $v1 -lt $v2 -a $v3 -gt $v2 ]

**For loop**

To loop repeated task in Linux we can use for loop, and it will exit the script if the inputs are iterated in the loop

Syntax:

for var in list

do

statement

done

simple script:

#!/bin/bash

for i in apple orange banana

do

echo "The fruit is $i"

done

**Real time example:**

cat ip.txt

www.google.com

www.facebook.com

www.flipkart.com

[www.teaetsdfgcv.com](http://www.teaetsdfgcv.com)

vi for.sh

#!/bin/bash

for ip in $(cat ip.txt)

do

ping -c3 $ip

done

./for.sh

**Decision Making**

Decision making in Bash scripting involves using conditional statements to perform different actions based on certain conditions.

if

elif

else

sample example:  
#!/bin/bash

user=sundar

if grep $user /etc/passwd; then

echo "The user is available"

else

echo "The use is not available"

fi

**realtime example:**

#!/bin/bash

for ip in $(cat ip.txt)

do

ping -c3 $ip > /dev/null 2>&1

if [ $? -eq 0 ]

then

echo "$ip server is reachable"

elif [ $? -eq 1 ]

then

echo "$ip server is NOT stable"

else

echo "$ip is not reachable"

fi

done

**Monitoring Scripts**

Monitoring scripts are used to check the status of systems, services, or resources and perform actions based on their status.

#!/bin/bash

threshold=10

usage=$(df / | grep / | awk '{print $5}' | sed 's/%//g')

if [ $usage -gt $threshold ];then

echo "Disk is above the threshold"

else

echo "Disk is good"

fi

**SSH Key Exchange**

SSH key exchange is a method of securely transferring public keys between systems to enable password-less authentication.

1. Enable root login

vi /etc/ssh/sshd\_config

PasswordAuthentication yes

PermitRootLogin yes

systemctl restart sshd

1. set a password for root

passwd root

1. try to connect from remote server it will ask for password

ssh [root@172.31.17.86](mailto:root@172.31.17.86)

1. enable the password less authentication

ssh-keygen

cd .ssh

ssh-copy-id -i id\_rsa.pub root@172.31.17.86

now check the password less authentication:

ssh [root@172.31.17.86](mailto:root@172.31.17.86)

now do a real time example with script:

#!/bin/bash

for ip in 172.31.17.86

do

ping -c3 $ip > /dev/null 2>&1

if [ $? -eq 0 ]

then

echo "$ip server is reachable"

**scp fruits root@172.31.17.86:/tmp**

else

echo "$ip is not reachable"

fi

done

xxxxxxxxxxxxxxxxxxxx

Session 7:

User input :

#!/bin/bash

echo "Enter the name: "

read name

echo "Enter your age: "

read age

./userinput.sh

Enter the name:

ravi

Enter your age:

25

Xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

**Get a secret password:**

#!/bin/bash

read -p "username: " user\_var

read -sp "password: " pass\_var

echo "Entered username: $user\_var"

echo "Entered password: $pass\_var"

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

**Writing the shell script to host the website**

#!/bin/bash

yum update

yum install httpd -y

cp index.html /var/www/html

systemctl start httpd

systemctl enable httpd

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

**Functions:**

Syntax:

function\_name () {

statement …

statement …

statement …

}

**To call this:**

function\_name

Function example:

#!/bin/bash

fun1 () {

echo "Hello Ford"

free -m

df -h

echo "This is a brandnew machine"

}

fun2 () {

echo "Hello maruthi"

echo "This is Australia"

date

}

read -p "Enter the datacenter: " building

if [ $building == 'india' ] || [ $building == 'us' ] || [ $building == 'uk' ]; then

fun1

elif [ $building == 'australia' ]; then

fun2

else

echo "Enter a valid building"

fi

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx