**BASH SCRIPTING**

Bash is a scripting language which allows users to automate, execute commands and manage system operations. To create a shell script the extension should be **.sh**

* Commands for shell scripting

1. **Touch**: - to create a file
2. **Vi/vim**: - To open the file if already created. If it is not created already then it will create a file and opens it
3. **Ls**: - To list the files
4. **Ls -ltr**: - to list the files with time stamp
5. **Man**: - Describes the functionality of a command
6. **Cat**: - It is used to print the content of the file without opening the file
7. **Sh filename (or) ./filename: -** It is used to execute the file. Which gives the output of the file
8. **Chmod:** - It is useful to grant the permissions of the users. **Ch 🡪 change**
9. **History: -** It is used to get the list of commands that we have used so far
10. **pwd:** - It is used to know in which directory we are at. Stands for **present working directoty**
11. **mkdir: -** It is used to create a directory**(folder).** Stands for **make directory**
12. **cd: -** It is used to go into the directory**(folder).** Stands for change directory
13. **cd .**. :- It is used to get back from the directory
14. **cp file1.txt file2.txt:** Copies contents from file1 to file2 → Use -r to copy directories
15. **mv file.txt ~/Desktop/:** Moves file to Desktop → Also used to rename files**: mv old.txt new.txt**
16. **rm:** - used to delete a file
17. **rm -rf: -** It is used to delete a directory
18. **nproc:** - It is used to list the cpu’s on the machine
19. **free -g:** - It is used to get the ram details
20. **top:** - used to get the list of processes that are running on the machine. Used to analyze the node status
21. **df -h:** - used to get the machine detail. Disc space
22. **set -x**: - debug mode in the shell script. It will print the output along with the command given
23. **set -e**: - it exits the script if there is an error. Without this command even if there is an error in the 1st line the script will execute, if we use this it will fail to execute the script at line 1 itself.
24. **Set -o pipefail**: - it is used along with **set -e**. because without using this, if there is an error in a line with “|” at the end of the line it wont stop the execution.
25. **ps -ef**: - used to get the processes details running in the machine
26. **| grep “ ”**: - **used to search for specific patterns of text**
27. **| :-** it is a pipe command. Used to send the output of the first command to the second command
28. **Wc -l: -** used to get the word/letter count
29. **Date: -** used to get today’s date and time
30. **Awk -f: -** used to get the single column information.

**EX:- ps -ef |grep amazon | awk -f “ ” ‘{print $2}’ 🡪 it prints the second column**

1. **Curl: -** it will retrieve the information from the internet. **Curl https://...........**
2. **Wget: -** it will retrieve the information from the internet and store in a file. If we want to see the content we need to open the file, but curl directly shows the output without storing it in any files
3. **Sudo su -: -** used to go to root user. **Sudo 🡪 substitute user do, su 🡪 switch user**
4. **Find: -** used to find the location of a file. **EX: - sudo find / -name pam**
5. **head -n 5 file.txt:** Shows first 5 lines
6. **tail -n 10 logs.txt:** Shows last 10 lines of a file
7. **uptime: -**Tells how long your machine’s been running
8. **Trap: -** it is used to trap the manual signals
9. **Kill: -** used to kill a process. **Ex: - kill -9 $process**
10. **Crontab -e: -** it is used to open the crontab to schedule a cronjob with timing
11. **Crontab -l: -** used to get the list of current cronjob’s without opening the file
12. **Crontab -r: -** used to remove the cronjob which is scheduled

* Shell scripting must start with **#!/bin/bash** 🡪 **(#!/) it is called as shebang**
* **Bash/dash/sh/ksh** 🡪 **these are called executables**
* **Previously sh will automatically redirect to bash using linking concept, but now sh is redirecting to dash as default**
* **Echo: - It is used to print something**
* **To write something in a file** first we need to go to insert mode, we can achieve that by **(click on esc then click on i)**
* **To save the file** we need to **(click on esc and enter :wq!)**
* **To exit from the file without saving** we need to **(click on esc and enter :q!)**
* **:wq! 🡪 to save the file and exit**
* **:q! 🡪 exit from the file without saving**
* **Chmod** has 3 categories

1. What are the permissions for a root user **(you)**
2. Which group has access **(your group)**
3. What are your permissions **(all users)**

**Linux uses chmod as 4, 2, 1 manner which says**

**4 as read**

**2 as write**

**1 as execute**

* **If we give chmod 777 then you, your group and all users has permission for read, write and execute the file**
* **If we give chmod 774 then you and your group has permission to read, write and execute but for all users it will be only read permission**
* **# 🡪** it is used to write the comments in shell script
* **Date | echo “Today’s date”** 🡪 this will give the output as today’s date. Because date is a system default command and it sends the output to stdin. But pipe (|) doesn’t have the ability to receive the info from stdin, it only takes from stdout.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**If else in shell scripting**

If [ condition ]

Then

Statement

Elif [ condition ]

Then

statement

Else

Statement

Fi

**For loop in shell scripting**

For condition

Do

Script

Done

**Aws cli reference** 🡪 used to get the commands for getting the aws resources details

**Aws s3 ls** 🡪 gives the list of s3 buckets in the aws.

**Aws ec2 describe-instances** 🡪 gives the list of ec2 instances in the aws

**Aws lambda list-functions** 🡪 gives the list of lambda functions

**Aws iam list-users** 🡪 gives the list of iam users

**Jq 🡪** it is used to get the single parameter from the whole list,

**ex: - ./filename.sh | jq ‘.list[].list[].id’ (or) we can directly write this inside the file itself.**

**JQ is json parser. Similarly we have yq which is yaml parser**

**Interview Questions**

1. **List some of the most commonly used shell commands?**

* Ls, cd, pwd, mkdir, rm, cp, mv, cat, grep, find, top, ps, kill, chmod, curl, wget

1. **Write a shell script to list all processes?**

* #!/bin/bash Ps -ef

1. **Write a script to print only errors from a remote log?**

* #!/bin/bash

Curl https://........ | grep “error”

1. **Write a shell script to print numbers divided by 3 & 5 and not 15?**

* #!/bin/bash

echo "printing from 1 to 100 which are divisible by 3 & 5 and not 15"

for ((i = 1; i<100; i++))

do

if (( (i % 3 ==0 || i % 5 ==0) && i % 15 !=0))

then

echo $i

fi

done

1. **Write a script to print a number of “S” In Mississippi?**

* #!/bin/bash

x=mississippi

grep -o "s" <<<"$x" | wc -l

1. **How will you debug the shell script?**

* Set -x

1. **What is crontab in linux? Can you provide an example of usage?**

* The cron daemon runs in the background and looks at a user’s crontab file to see if there are any tasks scheduled. If there are, it executes them at the specified time.

**\* \* \* \* \* /path/to/command**

**| | | | |**

**| | | | └─── Day of week (0-6; Sunday = 0)**

**| | | └───── Month (1-12)**

**| | └─────── Day of month (1-31)**

**| └───────── Hour (0-23)**

**└─────────── Minute (0-59)**

Let’s say we want to run a shell script **every day at 7:30 AM**. We need to add this to our crontab:

**30 7 \* \* \* /home/kondareddy/scripts/backup.sh**

1. **How to open a file in read only mode?**

* less filename.txt **(or)** vi -R filename.txt

1. **What is the difference between soft link and hard link?**

* [Difference Between Hard Link and Soft Link](https://www.tutorialspoint.com/difference-between-hard-link-and-soft-link)

1. **What Is the difference between the break and the continue statements?**

* Break 🡪 breaking the execution

Continue 🡪 skip the present condition and continue to the next

1. **What are some disadvantages of shell scripting?**

* Poor error handling

Limited data structures

Portability issues

Security risks

Not ideal for complex logic

Debugging can be painful

Speed limitations

1. **What are the different types of loops and when to use?**
2. **Is bash dynamical or statically typed?**

* It is dynamically typed

1. **Explain about a network troubleshooting utility?**

* **Traceroute** **🡪** it is used to know how many hops we are having to reach an endpoint

1. **How will you sort list of names in a file?**

* Using the sort command we can able to list the names in a file

1. **How will you manage logs of a system that generates huge log files every day?**

* The logrotate tool is the go-to solution on most Linux systems. It automatically rotates, compresses, removes, and mails log files based on rules we set.