LINUX PROGRAMMING

ASSIGNMENT-8

NAME: V S SAIDATTA USN: ENG24CY0175 ROLLNO:28

1. What is a user-defined function in shell scripting? Explain with an example

A.

A user-defined function groups commands under a name so you can call them multiple times with different inputs.

```
set -euo pipefail
greet() #define
{
        local name="${1:-user}"
        echo "Hello, $name"
}
greet "DATTA"      # call
greet              # uses default
```

2. Write a bash script with a function that multiply two integer numbers.

A.

```
#!/bin/bash


multiply()
{
    local num1=$1
    local num2=$2
    local result=$((num1 * num2))
    echo "Result: $result"
}
if [[ $# -ne 2 ]]; then
   echo "Usage: $0 <integer1> <integer2>"
   exit 1
```

fi
multiply "$1" "$2"

3. Explain how arrays (1D, 2D, and 3D) are declared in bash scripting.

A.

When compared to languages such as Python or C, arrays in Bash are relatively simple. There is native support for only 1D arrays; there is no direct syntax for 2D or 3D arrays, but you can work around that with associative arrays or if you name your keys judiciously.

- **1D Arrays**

  ```
  nums = (10 20 30)
  echo "${nums[1]}"        # 20
  nums+=("40")             # append
  echo "${#nums[@]}"       # length
  ```

- **2D emulation**: flatten keys row:col in an associative array.
  declare -A m

  ```
  m["0:0"] = 1;
  m["0:1"] = 2
  m["1:0"] = 3;
  m["1:1"] = 4
  echo "${m[1:1]}"       # 4
  ```

- **3D emulation**: extend key x:y:z.
  declare -A t

  ```
  t["0:0:0"] ="A"
  t["0:1:2"] ="B"
  ```

4. Write a shell script to display elements of an array.

A.

```
#!/bin/bash
# Define a 1D array
fruits = ("apple" "banana" "cherry" "date" "elderberry")
```

```
echo "Elements of the array are:"
for fruit in "${fruits[@]}"; do
        echo "$fruit"
Done
```

5. What is the purpose of cron in Linux?

A.

in Linux, cron is a time-driven job scheduler that allows scripts or commands to be run automatically at specified times and dates.Allows for automatically running tasks in the background at user-defined times and intervals (daily, hourly, weekly, etc.).

- Running **backups** every night
- **Rotating logs**
- Sending **scheduled emails**

6. Write a cron job to run a backup script every day at midnight?

A.

Edit your crontab:

crontab -e

Add:

0 0 * * * /usr/local/bin/backup.sh >> /var/log/backup.log 2>&1

7. How do you schedule a one-time job using at command?

A.

Ensure the daemon is running (atd service). Then:

```
echo "/usr/local/bin/backup.sh >> /var/log/backup.log 2>&1" | at 23:45
at 02:15 AM tomorrow -f /home/user/run_report.sh
atq
atrm <job_id>
```

8. Write a script to display disk usage using df and du.

A.

```
#!/bin/bash
echo "====== Disk Space Usage Summary (df) ======"
df -h
echo
echo "====== Current Directory Usage (du) ======"
du -sh ./* 2>/dev/null
```

9. How can you log the output of a script using the tee command?

A.

To save and show the output:

```
#!/bin/bash
echo "System Info:" | tee system_log.txt
date | tee -a system_log.txt
df -h | tee -a system_log.txt
```

10. Explain with an example how shell scripting can automate system administration tasks.

A.

Example backup script:

```
#!/bin/bash
SOURCE="/home/user/documents"
DESTINATION="/home/user/backup"
DATE=$(date +%Y-%m-%d)
mkdir -p "$DESTINATION/$DATE"
cp -r "$SOURCE" "$DESTINATION/$DATE"
echo "Backup completed on $DATE."
```

Now, to automate the backup with cron it would like:

```
0 2 * * * /home/user/backup_script.sh
```