Department Of Cybersecurity

Linux Programming

Assignment 8

Name: Mohd Sahil Ansari USN: ENG24CY0134

Roll No:22 Class: 3A

1. What is **a user-defined function** in shell scripting? Explain with an example. (**CO4**)

   Ans: A user-defined function in shell scripting implies the set of reusable block of code carrying out a specific task.

   It helps in making script more organised and maintainable.

   Syntax: function_name(){

   #function body

   Commands

   }

```
#!/bin/bash
# User-defined function to greet a user
def_user() {
    echo "Hello, $1! welcome to assignment 8."
}

# Calling the function
def_user "Respected Faculty"
whoami
```

input

```
Hello, Respected Faculty! welcome to assignment 8.
runner6
```

2. Write a bash script with a function that **multiply two integer numbers**. (**CO4**)

Ans:

```
#!/bin/bash
#multiplication of two numbers
echo "Multiplication of two numbers"
multi(){
    local n1=$1
    local n2=$2
    local result=$((n1 * n2))
    echo "The product of $n1 and $n2 is: $result"
    return $result
}

echo "Enter first number:"
read first
echo "Enter second number:"
read second
multi $first $second
whoami
```

input

```
Multiplication of two numbers
Enter first number:
15
Enter second number:
4
The product of 15 and 4 is: 60
runner91
```

3. Explain how **arrays (1D, 2D, and 3D)** are declared in bash scripting. (**CO4**)

Ans:

```
#Array declaration in bash
#!/bin/bash
# 1D array
animals=("cats" "squirrel" "sparrow")
echo "1D Array: ${animals[@]}"

# 2D array
declare -A mat
mat[0,0]="L"
mat[0,1]="O"
mat[1,0]="G"
mat[1,1]="S"
echo "2D Array element [1,1]: ${mat[1,1]}"

# 3d
declare -A cuboid
cuboid[0,0,0]="4"
cuboid[0,0,1]="5"
cuboid[1,1,1]="6"
echo "3 D element [0,0,1]: ${cuboid[0,0,1]}"
```

input

```
1D Array: cats squirrel sparrow
2D Array element [1,1]: S
3 D element [0,0,1]: 5
```

4. Write a shell script to display elements of an array. (**CO4**)

Ans:

```
#!/bin/bash
#!/bin/bash
# Script to display array elements

# Declare and initialize array
fruits=("Apple" "Banana" "Cherry" "Date" "Elderberry")

echo "     Displaying Array Elements     "
echo

# Display all elements at once
echo "Method 1 - All elements: ${fruits[@]}"

# Display elements with index using for loop
echo "Elements with index:"
for i in "${!fruits[@]}"; do
    echo "Index $i: ${fruits[i]}"
done
echo

# Display array length
echo "Total number of elements: ${#fruits[@]}"

# Display specific element
echo "First element: ${fruits}"
echo "Last element: ${fruits[-1]}"
```

5. What is the purpose of **cron** in Linux? (**CO4**)

   Ans: Cron is time oriented job scheduler in Linux/Unix systems that gives the user flexibility to run scripts automatically within specified time-constraints.

   Purpose:

   a. Automates repetitive tasks.

   b. Runs resource-intensive tasks during off-peak hours.

   c. Schedule regular maintanence tasks like log rotation , backups etc.

   d. Schedule batch processing jobs.

6. Write a **cron** job to run a backup script every day at midnight. (**CO4**)

Ans: Step 1: Create the backup script

Step 2: Make script executable

Step 3: Add cron job

Step 4: Verify cron job

7. How do you schedule a one-time job using at command? (**CO4**)

Ans: 'at' command is used to schedule a specific job to run at a specific time.

Syntax(in bash): at [time][date]

at> cmd1

at>cmd2

at> Ctrl+D(to exit)

example: at 10:00 AM Dec 25 (Scheduling for specific date)

at 15:30 2023-12-25

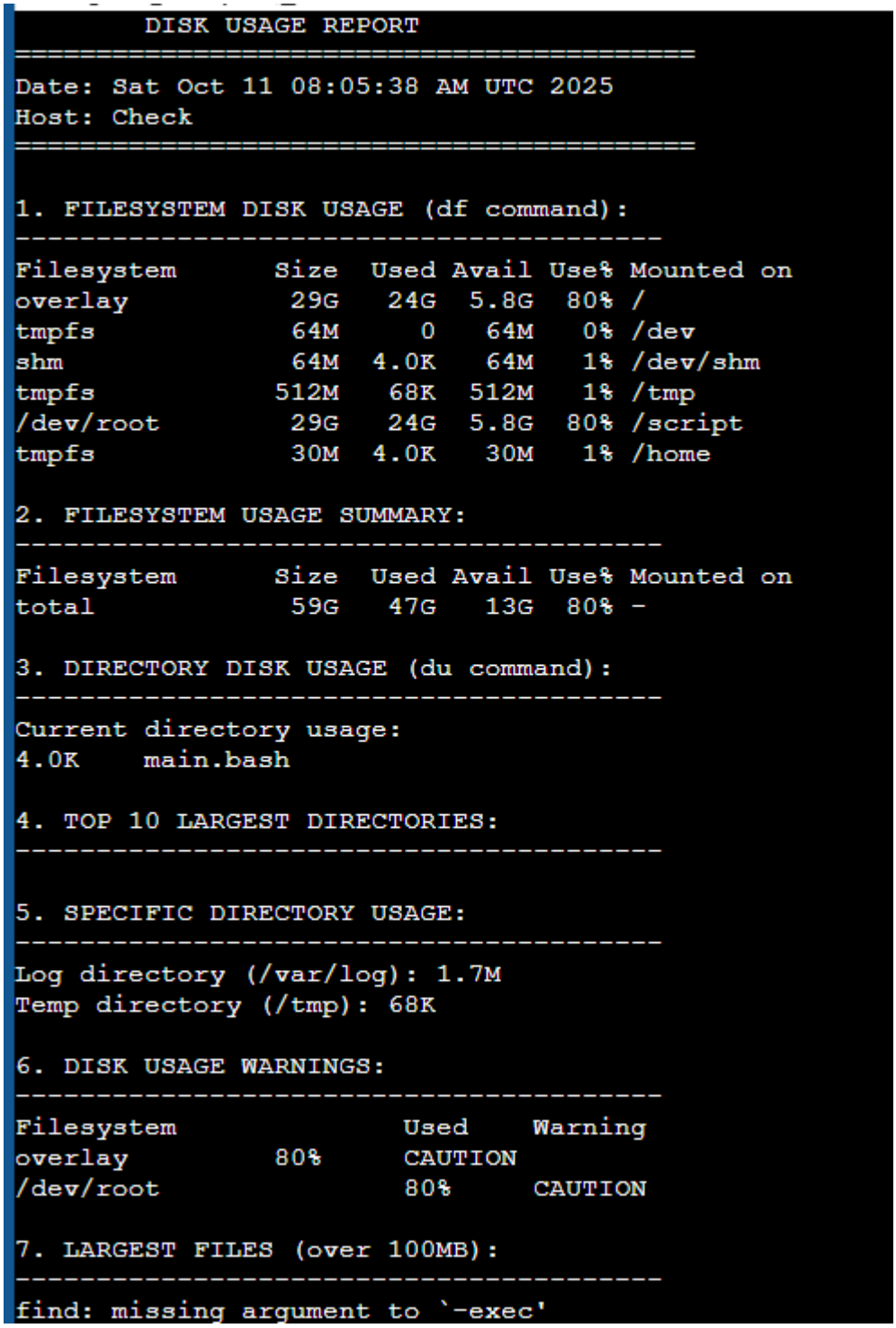
at 9:00 PM next Friday

8. Write a script to display disk usage using **df** and **du**. (**CO4**)

Ans:

```bash
#!/bin/bash
# Script to display disk usage using df and du

echo "========================================"
echo "        DISK USAGE REPORT"
echo "========================================"
echo "Date: $(date)"
echo "Host: $(hostname)"
echo "========================================"
echo

# Display filesystem disk usage using df
echo "1. FILESYSTEM DISK USAGE (df command):"
echo "----------------------------------------"
df -h  # Human readable format
echo

echo "2. FILESYSTEM USAGE SUMMARY:"
echo "----------------------------------------"
df -h --total | grep -E '(Filesystem|total)'
echo

# Display directory disk usage using du
echo "3. DIRECTORY DISK USAGE (du command):"
echo "----------------------------------------"
echo "Current directory usage:"
du -sh * 2>/dev/null | sort -hr
echo

echo "4. TOP 10 LARGEST DIRECTORIES:"
echo "----------------------------------------"
du -sh */ 2>/dev/null | sort -hr | head -10
echo
```

```
# Check specific directories
echo "5. SPECIFIC DIRECTORY USAGE:"
echo "-----------------------------------------"
if [ -d "/var/log" ]; then
    echo "Log directory (/var/log): $(du -sh /var/log 2>/dev/null
fi
if [ -d "/tmp" ]; then
    echo "Temp directory (/tmp): $(du -sh /tmp 2>/dev/null | cut -
fi
if [ -d "$HOME" ]; then
    echo "Home directory: $(du -sh $HOME 2>/dev/null | cut -f1)"
fi
echo

# Disk usage warnings
echo "6. DISK USAGE WARNINGS:"
echo "-----------------------------------------"
df -h | awk '
BEGIN {print "Filesystem\t\tUsed\tWarning"}
NR>1 {
    use = $5
    gsub(/%/, "", use)
    if (use > 90) print $1 "\t\t" $5 "\tCRITICAL"
    else if (use > 80) print $1 "\t\t" $5 "\tWARNING"
    else if (use > 70) print $1 "\t\t" $5 "\tCAUTION"
}'
echo

# Find large files
echo "7. LARGEST FILES (over 100MB):"
echo "-----------------------------------------"
find / -type f -size +100M -exec ls -lh {} ; 2>/dev/null | head -5
echo

echo "======== END OF DISK USAGE REPORT ========"
```



```
        DISK USAGE REPORT
=========================================
Date: Sat Oct 11 08:05:38 AM UTC 2025
Host: Check
=========================================

1. FILESYSTEM DISK USAGE (df command):
-----------------------------------------
Filesystem      Size  Used Avail Use% Mounted on
overlay          29G   24G  5.8G  80% /
tmpfs            64M     0   64M   0% /dev
shm              64M  4.0K   64M   1% /dev/shm
tmpfs           512M   68K  512M   1% /tmp
/dev/root        29G   24G  5.8G  80% /script
tmpfs            30M  4.0K   30M   1% /home

2. FILESYSTEM USAGE SUMMARY:
-----------------------------------------
Filesystem      Size  Used Avail Use% Mounted on
total            59G   47G   13G  80% -

3. DIRECTORY DISK USAGE (du command):
-----------------------------------------
Current directory usage:
4.0K    main.bash

4. TOP 10 LARGEST DIRECTORIES:
-----------------------------------------

5. SPECIFIC DIRECTORY USAGE:
-----------------------------------------
Log directory (/var/log): 1.7M
Temp directory (/tmp): 68K

6. DISK USAGE WARNINGS:
-----------------------------------------
Filesystem              Used    Warning
overlay         80%     CAUTION
/dev/root               80%     CAUTION

7. LARGEST FILES (over 100MB):
-----------------------------------------
find: missing argument to `-exec'
```

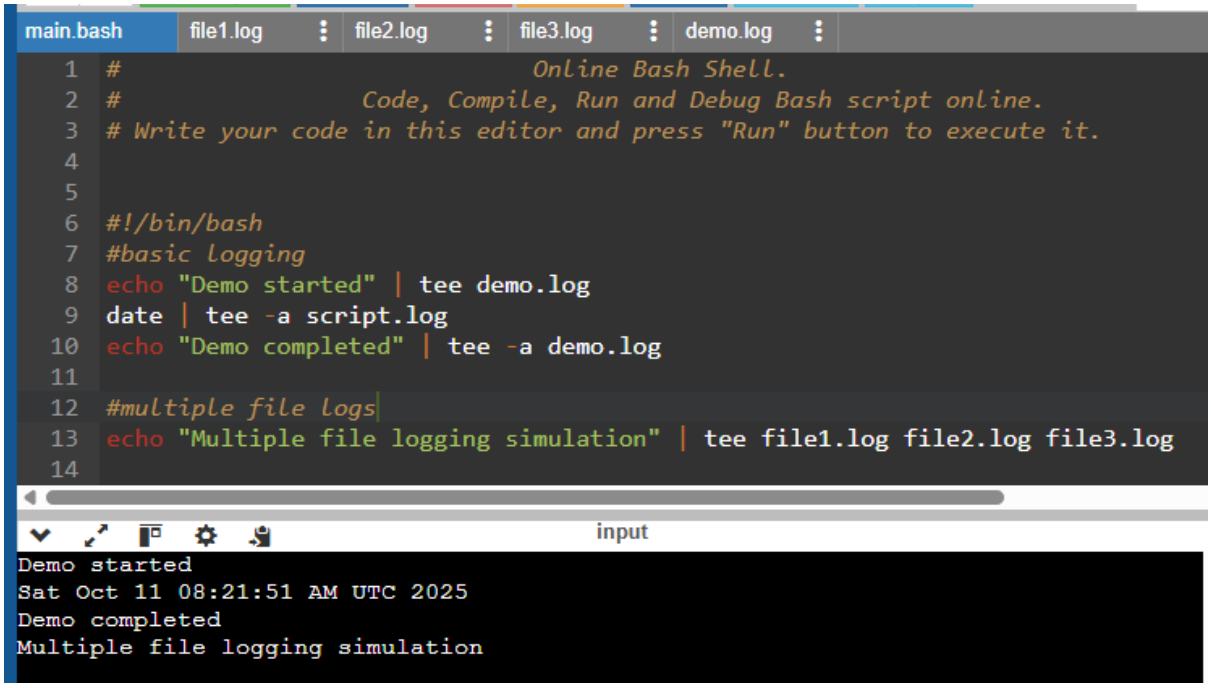
9. How can you log the output of a script using the **tee** command? (**CO4**)

Ans: The 'tee' command reads from standard input and writes both to standard output and files simultaneously, making it mist apt for logging script output.

Syntax: command | tee [options] filename

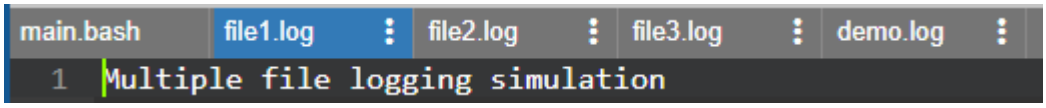
Options: -a: Append to file intead of overwriting

-i: Ignore interrupt signals



All the file1.log, file2.log, file3.log consists of

"Multiple file logging Simulation"



10. Explain with an example **how shell scripting can automate system** administration tasks. (**CO4**)

Ans:

Shell scripting automates repetitive admin tasks such as backups, user management, or monitoring by running predefined commands automatically.

Example: A script to check system memory and email an alert if usage exceeds a threshold:

```
mem_usage=$(free | grep Mem | awk '{print $3/$2 * 100.0}')
if (( ${mem_usage%.*} > 80 )); then
  echo "Memory usage is above 80%" | mail -s "Alert"
admin@example.com
fi
```