NAME : SAI CHARITHA

ROLL NO : 422128

SEC : A

Shell Scripts :

1)b

```bash
#!/bin/bash


# Define list of machines
machines=("172.50.11.106" "172.50.9.213" "127.0.0.1")  # Add your machine names or IPs here


# SSH username
username="your_username"


# Function to get memory usage
get_memory_usage() {
    ssh "$username@$1" "free -m | awk 'NR==2{printf \"Memory Usage: %.2f%%\\n\", \$3*100/\$2}'"
}


# Function to get CPU usage
get_cpu_usage() {
    ssh "$username@$1" "top -bn1 | grep 'Cpu(s)' | sed 's/.*, *\([0-9.]*\)%* id.*/\1/' | awk '{print \"CPU Usage: \"100 - \$1\"%\"}'"
}


# Iterate over machines
for machine in "${machines[@]}"; do
    echo "Machine: $machine"
    get_memory_usage "$machine"
    get_cpu_usage "$machine"
    echo "--------------------------------------"
done
```

2)more

```bash
#!/bin/bash

# Check if a filename is provided as argument
if [ $# -eq 0 ]; then
    echo "Usage: $0 states.txt"
    exit 1
fi

# Check if the file exists
if [ ! -f "$1" ]; then
    echo "File '$1' does not exist."
    exit 1
fi

# Display the content of the file using the more command
more "$1"
```

3)nice

```bash
#!/bin/bash

# Check if a command is provided as argument
if [ $# -eq 0 ]; then
    echo "Usage: $0 <command>"
    exit 1
fi

# Run the provided command with a nice value of 10
nice -n 10 "$@"
```

4)nl

```bash
#!/bin/bash
```

```bash
# Check if a filename is provided as argument
if [ $# -eq 0 ]; then
    echo "Usage: $0 capital.txt"
    exit 1
fi


# Check if the file exists
if [ ! -f "$1" ]; then
    echo "File '$1' does not exist."
    exit 1
fi


# Display the content of the file with line numbers using the nl command
nl "$1"
```

5)pr

```bash
#!/bin/bash


# Check if a filename is provided as argument
if [ $# -eq 0 ]; then
    echo "Usage: $0 states.txt>"
    exit 1
fi


# Check if the file exists
if [ ! -f "$1" ]; then
    echo "File '$1' does not exist."
    exit 1
fi


# Display the content of the file with pagination using the pr command
pr "$1"
```

6)psswd

```bash
#!/bin/bash

# Prompt user for username
read -p "Enter username: " username

# Check if the username is provided
if [ -z "$username" ]; then
    echo "Username is required."
    exit 1
fi

# Check if the user exists
if ! id "$username" &>/dev/null; then
    echo "User '$username' does not exist."
    exit 1
fi

# Prompt user for a new password
read -s -p "Enter new password for $username: " password
echo

# Prompt user to confirm the new password
read -s -p "Confirm new password: " password_confirm
echo

# Check if passwords match
if [ "$password" != "$password_confirm" ]; then
    echo "Passwords do not match. Please try again."
    exit 1
fi
```

```bash
# Change the user's password using the passwd command
echo "$password" | passwd --stdin "$username"


# Check if the password change was successful
if [ $? -eq 0 ]; then
    echo "Password for user '$username' has been successfully updated."
else
    echo "Failed to update password for user '$username'."
    exit 1
fi
```

7)rcp

```bash
#!/bin/bash


# Check if source and destination files are provided
if [ $# -lt 2 ]; then
    echo "Usage: $0 <source_file> <destination_host>:<destination_file>"
    exit 1
fi


# Extract source and destination information
source_file="$1"
destination_host="${2%%:*}"
destination_file="${2##*:}"


# Check if the source file exists
if [ ! -f "$source_file" ]; then
    echo "Source file '$source_file' does not exist."
    exit 1
fi
```

```bash
# Check if the destination host is provided
if [ -z "$destination_host" ]; then
    echo "Destination host is required."
    exit 1
fi


# Check if the destination file is provided
if [ -z "$destination_file" ]; then
    echo "Destination file is required."
    exit 1
fi


# Copy the file using rcp
rcp "$source_file" "$destination_host":"$destination_file"


# Check if the copy operation was successful
if [ $? -eq 0 ]; then
    echo "File '$source_file' copied successfully to '$destination_host:$destination_file'."
else
    echo "Failed to copy file '$source_file' to '$destination_host:$destination_file'."
    exit 1
fi
```

8)rlogin

```bash
#!/bin/bash


# Check if a hostname is provided as argument
if [ $# -eq 0 ]; then
    echo "Usage: $0 student"
    exit 1
fi
```

```bash
# Check if the hostname is provided
if [ -z "$1" ]; then
    echo "Hostname is required."
    exit 1
fi

# Prompt user for username
read -p "Enter username for $1: " username

# Check if the username is provided
if [ -z "$username" ]; then
    echo "Username is required."
    exit 1
fi

# Attempt to login using rlogin
rlogin "$1" -l "$username"
```
9)rsh
```bash
#!/bin/bash

# Check if a hostname and command are provided
if [ $# -lt 2 ]; then
    echo "Usage: $0 <hostname> <command>"
    exit 1
fi

# Extract hostname and command
hostname="$1"
shift
command="$@"
```

```bash
# Check if the hostname is provided
if [ -z "$hostname" ]; then
    echo "Hostname is required."
    exit 1
fi


# Check if the command is provided
if [ -z "$command" ]; then
    echo "Command is required."
    exit 1
fi


# Check if the 'rsh' command is available
if ! command -v rsh &>/dev/null; then
    echo "The 'rsh' command is not available."
    exit 1
fi


# Execute the command on the remote system using rsh
rsh "$hostname" "$command"
```

10)talk

```bash
#!/bin/bash


# Check if a username is provided
if [ $# -eq 0 ]; then
    echo "Usage: $0 student"
    exit 1
fi


# Check if the username is provided
if [ -z "$1" ]; then
```

```bash
    echo "Username is required."

    exit 1

fi


# Check if the 'talk' command is available

if ! command -v talk &>/dev/null; then

    echo "The 'talk' command is not available."

    exit 1

fi


# Start the talk session with the specified user

talk "$1"
```

11)telnet

```bash
#!/bin/bash


# Check if host and port are provided

if [ $# -lt 2 ]; then

    echo "Usage: $0 <host> <port>"

    exit 1

fi


# Extract host and port

host="$1"

port="$2"


# Check if host and port are provided

if [ -z "$host" ] || [ -z "$port" ]; then

    echo "Both host and port are required."

    exit 1

fi
```

```bash
# Check if telnet command is available
if ! command -v telnet &> /dev/null; then
    echo "telnet command is not available."
    exit 1
fi

# Run telnet command
telnet "$host" "$port"
```

12)tput

```bash
#!/bin/bash

# Clear the screen
tput clear

# Get the number of columns and rows of the terminal
cols=$(tput cols)
rows=$(tput lines)
echo "Terminal size: $cols columns x $rows rows"

# Set terminal text color to red
tput setaf 1
echo "This is red text"

# Set terminal text color to green
tput setaf 2
echo "This is green text"

# Set terminal text color to default
tput sgr0

# Set terminal background color to yellow
```

```bash
tput setab 3
echo "This has a yellow background"

# Reset terminal background color
tput sgr0

# Move cursor to specific position
tput cup 10 20
echo "Cursor moved to row 10, column 20"
```

13)tty

```bash
#!/bin/bash

# Get the terminal filename
terminal=$(tty)

# Print the terminal filename
echo "Terminal filename: $terminal"

# Get the terminal type
terminal_type=$(tty -s && echo "$TERM" || echo "not a tty")
echo "Terminal type: $terminal_type"
```

14)uname

```bash
#!/bin/bash

# Get system name
system_name=$(uname -s)
echo "System name: $system_name"

# Get node (machine) name
node_name=$(uname -n)
echo "Node name: $node_name"
```

```bash
# Get kernel release
kernel_release=$(uname -r)
echo "Kernel release: $kernel_release"

# Get kernel version
kernel_version=$(uname -v)
echo "Kernel version: $kernel_version"

# Get machine hardware name
machine=$(uname -m)
echo "Machine hardware: $machine"

# Get processor type
processor=$(uname -p)
echo "Processor type: $processor"

# Get operating system name
operating_system=$(uname -o)
echo "Operating system: $operating_system"
```

15)wc

```bash
#!/bin/bash

# Check if a filename is provided
if [ $# -eq 0 ]; then
    echo "Usage: $0 states.txt"
    exit 1
fi

# Check if the file exists
if [ ! -f "$1" ]; then
```

```
    echo "File '$1' does not exist."

    exit 1

fi


# Get line count

lines=$(wc -l < "$1")

echo "Number of lines: $lines"


# Get word count

words=$(wc -w < "$1")

echo "Number of words: $words"


# Get byte count

bytes=$(wc -c < "$1")

echo "Number of bytes: $bytes"
```

output :

```
-rw-rw-r-- 1 student student  53 Apr  4 10:46 capital.txt
-rwxrwxr-x 1 student student 298 Apr  4 11:38 more.sh
-rwxrwxr-x 1 student student 194 Apr  4 11:46 nice.sh
-rwxrwxr-x 1 student student 313 Apr  4 11:41 nl.sh
-rw-rw-r-- 1 student student  53 Apr  4 10:28 states.txt
student@ai-HP-ProDesk-600-G4-MT:~/Desktop/422128$ chmod +x passwd.sh
chmod: cannot access 'passwd.sh': No such file or directory
student@ai-HP-ProDesk-600-G4-MT:~/Desktop/422128$ chmod +x psswd.sh
student@ai-HP-ProDesk-600-G4-MT:~/Desktop/422128$ ./psswd.sh
Enter username: charitha
User 'charitha' does not exist.
student@ai-HP-ProDesk-600-G4-MT:~/Desktop/422128$ ./psswd.sh
Enter username: student
Enter new password for student:
Confirm new password:
passwd: unrecognized option '--stdin'
Usage: passwd [options] [LOGIN]

Options:
  -a, --all                     report password status on all accounts
  -d, --delete                  delete the password for the named account
  -e, --expire                  force expire the password for the named account
  -h, --help                    display this help message and exit
  -k, --keep-tokens             change password only if expired
  -i, --inactive INACTIVE       set password inactive after expiration
                                to INACTIVE
  -l, --lock                    lock the password of the named account
  -n, --mindays MIN_DAYS        set minimum number of days before password
                                change to MIN_DAYS
  -q, --quiet                   quiet mode
  -r, --repository REPOSITORY   change password in REPOSITORY repository
  -R, --root CHROOT_DIR         directory to chroot into
  -S, --status                  report password status on the named account
  -u, --unlock                  unlock the password of the named account
  -w, --warndays WARN_DAYS      set expiration warning days to WARN_DAYS
  -x, --maxdays MAX_DAYS        set maximum number of days before password
                                change to MAX_DAYS

Failed to update password for user 'student'.
student@ai-HP-ProDesk-600-G4-MT:~/Desktop/422128$ ./psswd.sh
Enter username: student
Enter new password for student:
Confirm new password:
passwd: unrecognized option '--stdin'
```

```
Failed to update password for user 'student'.
student@ai-HP-ProDesk-600-G4-MT:~/Desktop/422128$ chmod +x pr.sh
student@ai-HP-ProDesk-600-G4-MT:~/Desktop/422128$ ./pr.sh a.txt
student@ai-HP-ProDesk-600-G4-MT:~/Desktop/422128$ ./pr.sh
Usage: ./pr.sh states.txt>
student@ai-HP-ProDesk-600-G4-MT:~/Desktop/422128$ pr states.txt


2024-04-04 10:28                     states.txt                     Page 1


Andhra Pradesh
Telangana
TamilNadu
Maharashtra
Jammu
```

student@ai-HP-ProDesk-600-G4-MT: ~/Desktop/422128

student@ai-HP-ProDesk-600-G4-MT: ~/Desktop/422... ×   student@ai-HP-ProDesk-600-G4-MT: ~/Desktop/422...   student@ai-HP-ProDesk-600-G4-MT: ~/Desktop/422... ×

```
student@ai-HP-ProDesk-600-G4-MT:~/Desktop/422128$ chmod +x rlogin.sh
student@ai-HP-ProDesk-600-G4-MT:~/Desktop/422128$ ./rlogin.sh
Usage: ./rlogin.sh student
student@ai-HP-ProDesk-600-G4-MT:~/Desktop/422128$ rlogin states.txt -l student
rlogin: Could not resolve hostname states.txt: Name or service not known
student@ai-HP-ProDesk-600-G4-MT:~/Desktop/422128$ rlogin student -l student
rlogin: Could not resolve hostname student: Temporary failure in name resolution
student@ai-HP-ProDesk-600-G4-MT:~/Desktop/422128$ []
```

student@ai-HP-ProDesk-600-G4-MT: ~/Desktop/422128

student@ai-HP-ProDesk-600-G4-MT: ~/Desktop/422... ×   student@ai-HP-ProDesk-600-G4-MT: ~/Desktop/422...   student@ai-HP-ProDesk-600-G4-MT: ~/Desktop/422... ×

```
Terminal size: 151 columns x 43 rows
This is red text
This is green text
This has a yellow background




                    Cursor moved to row 10, column 20
student@ai-HP-ProDesk-600-G4-MT:~/Desktop/422128$ ./tty.sh
Terminal filename: /dev/pts/1
Terminal type: xterm-256color
student@ai-HP-ProDesk-600-G4-MT:~/Desktop/422128$ ./uname.sh
System name: Linux
Node name: ai-HP-ProDesk-600-G4-MT
Kernel release: 5.15.0-87-generic
Kernel version: #97~20.04.1-Ubuntu SMP Thu Oct 5 08:25:28 UTC 2023
Machine hardware: x86_64
Processor type: x86_64
Operating system: GNU/Linux
student@ai-HP-ProDesk-600-G4-MT:~/Desktop/422128$ ./wc.sh states.txt
Number of lines: 5
Number of words: 6
Number of bytes: 53
student@ai-HP-ProDesk-600-G4-MT:~/Desktop/422128$ who
student  :0           2024-04-04 10:14 (:0)
student@ai-HP-ProDesk-600-G4-MT:~/Desktop/422128$ []
```