|  |  |  |  |
| --- | --- | --- | --- |
|  | **Course Name:** Operating System Lab | **Experiment No. 4** | |
| **Course Code :** PCC-CS592 | **Branch: CSE** | **Semester: 5** |

## AIM:

* + **Details understanding about memory devices and file system.**

## Understanding of managing the system as per requirement.

**DISK PARTITIONING**

## View all Disk Partitions in Linux

The **–l**  argument stand for (listing all partitions) is used with **fdisk** command to view all available partitions on Linux.



Floppy (a:) /dev/fd0 Floppy (b:) /dev/fd1

1st Hard disk (master, IDE-0) /dev/hda Hard disk (slave, IDE-0) /dev/hdb

Hard disk (master, IDE-1) /dev/hdc, etc. 1st SCSI hard disk /dev/sda

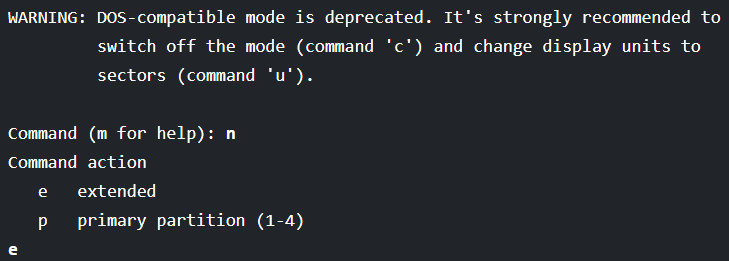
2nd SCSI hard disk /dev/sdb, etc.

To view all partitions of specific hard disk use the option ‗-l‗ with device name. For example, the following command will display all disk partitions of device /dev/sda.



## Create a New Partition in Linux

To create new partition for sda the option ‗-n‘ choosed.



While creating a new partition, it will ask you two options

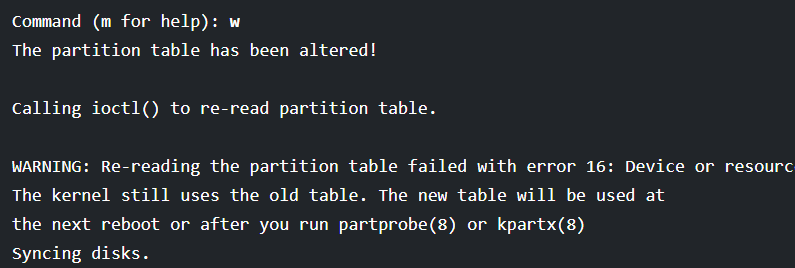
‗extended‗or ‗primary‗ partition creation.

Press ‗e‗ for extended partition and ‗p‗ for primary partition.

Then it will ask you to enter following two inputs.

First cylinder number of the partition to be create.

Last cylinder number of the partition to be created (Last cylinder, +cylinders or +size).



## Creating File system or format the partition

Type the following command in the terminal to format a partition. Here /dev/sda4 is my newly created partition.

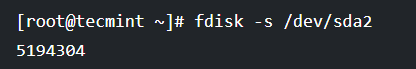
Partition type may be ext2, ext3, ext4 or many other



Partition type may be ext2, ext3, ext4 or many other

## Check size of the partition

After formatting new partition, check the size of that partition using flag s (displays size in blocks) with fdisk command.

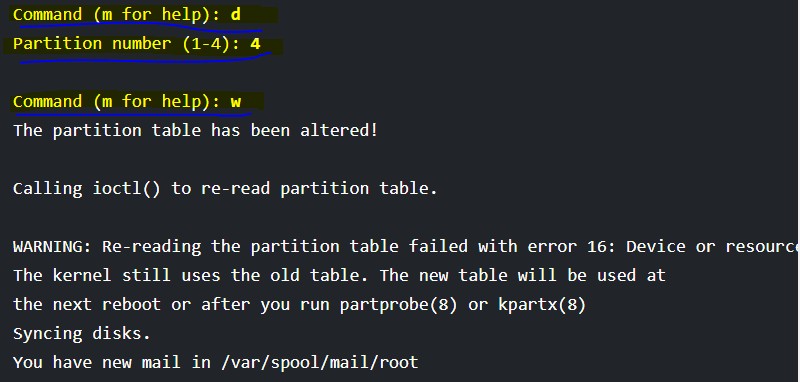


## Delete the partition

If you would like to delete a specific partition from the specific hard disk such as /dev/sda. we must be in fdisk command mode to do this.



Next, enter ‗d‗ to delete any given partition name from the system.

it will prompt to enter partition number that we want to delete from /dev/sda hard disk. Suppose we enter number ‘4’ here, then it will delete partition number ‘4’

**SWAP SPACE**

*Swap space* in Linux is used when the amount of physical memory (RAM) is full. If the system needs more memory resources and the RAM is full, inactive pages in memory are moved to the swap space. While swap space can help machines with a small amount of RAM, it should not be considered a replacement for more RAM. Swap space is located on hard drives, which have a slower access time than physical memory.

## Recommended System Swap Space



**Check swap space size**

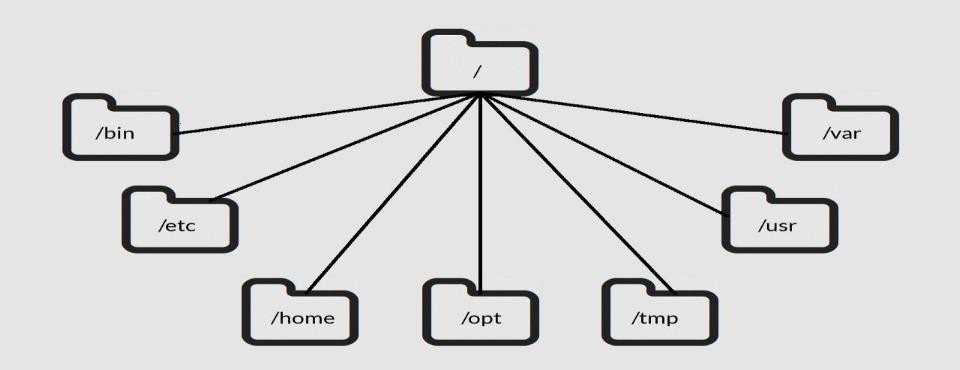
Use free-h to check swap space.

## Increase or decrease swap space

Use mkswap and swapon command to increase swap space. Use swapoff command to remove swap space.

**File Hierarchy Structure**

The Linux File Hierarchy Structure or the Filesystem Hierarchy Standard (FHS) defines the directory structure and directory contents in Unix-like operating systems.It is maintained by the Linux Foundation.



## Mounting a File System

One of the great things about Linux is that it has the ability to access data stored on many different file systems, even if these file systems are from other operating systems.

Example: windows (NTFS, FAT, FAT16, or FAT32) Unix (ext,ext2.ext3.nfs,vfat,smb)

Check the mount point by df –h command.

Step 1: Making a directory

#mkdir /mnt/mdir

Step 2: The next step would be to mount the file system to that folder or mount oint. # mount -t vfat /dev/sdb /mnt/mdir

## Unmount a filesystem

umount /mnt/mdir

Use df –h to check it is unmounted or not.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Course Name:** Operating System Lab | **Experiment No. 5** | |
| **Course Code :** PCC-CS592 | **Branch: CSE** | **Semester: 5** |

## AIM:

* + **Learn to make automated scripts.**

## Learn to control administration task sch as user management, system monitoring,scheduling backup etc.

**Shell:**

A shell is simply software the gives a user interface to various operating system functions and services. Shells provide a way for us to communicate with the operating system.

The prompt, **$**, which is called the **command prompt**, is issued by the shell. While the prompt is displayed, we can type a command.

**SheBang (#!) in Shell Scripting**

The **(#)** and **(!)** signs together are called **SheBang(#!)** in **Shell Scripting**. When the script is run with the **SheBang(#!)** in its first line, it instructs the interpreter to execute the script line-by-line. Programmer need to write the **SheBang(#!)** along with the **bash path directive: /bin/bash** that denotes the execution of a specified type(**bash**) of the scripts.

**The syntax for SheBang (#!) in Shell Scripting is given below:**

|  |
| --- |
| #!/bin/bash |

### How to Write and Execute a Bash Script in Linux

#### Step 1: Create a Shell Script in Linux

➊ At first, open a terminal window by pressing **CTRL**+**ALT**+**T**.

➋ Then, create a **bin** folder in **home directory** by typing the following command.

|  |
| --- |
| mkdir bin |

➌ After that, create a bash script file inside the **bin directory** with the command below.

|  |
| --- |
| vi bin/hello\_world.sh |

❹ Now, write the following script in **hello\_world.sh** file.

|  |
| --- |
| #!/bin/bash echo "Hello World" |

❺ To save and exit from the script, press ESC+SHIFT+**CTRL** and type WQrespectively.

❻ Now, Type the following to make the script executable for the current user in your system.

|  |
| --- |
| chmod u+rwx bin/hello\_world.sh |

#### Step 2: Running the “Hello World” Bash Script in Linux

|  |
| --- |
| bash hello\_world.sh |

## Input and output commands:

* + **read:**-For input values use **read <variabe>**.
  + **echo:**-For output anything use **echo “string”**

or to display the value of variable use **$** before the variable.

* + Using command line argument input can be possible.

|  |
| --- |
| **Variable & Description** |
| **$0:** The filename of the current script. |
| **$1 $2 $3 ….. :** These variables correspond to the arguments with which a script was invoked. The first argument is $1, the second argument is $2, and so on. |
| **$#:** The number of arguments supplied to a script. |
| **$\*:** All the arguments are double quoted. If a script receives two arguments, $\* is equivalent to  $1 $2. |
| **$@:** All the arguments are individually double quoted. If a script receives two arguments, $@ is equivalent to $1 $2. |

#### Example 1: Read, Store and Display User Input using Bash Script

**Code:**

|  |
| --- |
| #!/bin/bash echo "Enter a number:" read num echo "The number is: $num" |

**Output:**

|  |
| --- |
| Enter a number:12 The number is: 12 |

#### Example 2: Read User Input with Prompt Message using Bash Script

**Code:**

|  |
| --- |
| #!/bin/bash read -p "Enter a number:" num echo "The number is: $num" |

**Output:**

|  |
| --- |
| Enter the number:12  The number is: 12 |

#### Example 3: Concatenating Multiple Variables

**Code:**

|  |
| --- |
| #!/bin/bash # Declaration of variables name='My name is Sujit.' age='My age is 12.' # Concatenation info="${name} ${age}" echo "Result: $info" |

**Output:**

|  |
| --- |
| Result: My name is Sujit. My age is 12. |

#### Example 4: Generating a Random Number between Two Given Numbers

**Code:**

|  |
| --- |
| #!/bin/bash read -p "Enter minimum range:" min read -p "Enter maximum range:" max r\_num=$(( $RANDOM % ($max - $min + 1) + $min )) echo "Random Number: $r\_num" |

**Output:**

|  |
| --- |
| Enter minimum range:10 Enter maximum range:35 Random Number: 24 |

#### Example 5: Check if a Number is an Even or Odd

**Code:**

|  |
| --- |
| #!/bin/bash read -p "Enter a number:" num if [ $((num%2)) == 0 ] then  echo "The number is even" else  echo "The number is odd" fi |

**Output:**

|  |
| --- |
| Enter a number:25 The number is odd |

#### Example 6: Perform an Arithmetic Operation Based on User Input

**Code:**

|  |
| --- |
| !/bin/bash read -p "Enter a number:" num1 read -p "Enter a smaller number:" num2 read -p "Enter an operand:" op if [ $op == + ] then  echo "$num1 + $num2 = $((num1+num2))" elif [ $o == - ] then  echo "$num1 - $num2 = $((num1-num2))" elif [ $op == \* ] then  echo "$num1 \* $num2 = $((num1\*num2))" elif [ $op == / ] then  echo "$num1 / $num2 = $((num1/num2))" else  echo "Operator not listed" fi |

**Output:**

|  |
| --- |
| Enter a number:34 Enter a smaller number:14 Enter an operand:+ 34 + 14 = 48 |

#### Example 7: Check if a Given Number is Positive or Negative

**Code:**

|  |
| --- |
| #!/bin/bash read -p "Enter a number:" num if [ $num -gt 0 ] then  echo "The number is Positive!" elif [ $num -lt 0 ] then  echo "The number is Negative!" else  echo "The number is Zero!!" fi |

**Output:**

|  |
| --- |
| Enter a number:12 The number is Positive! |

#### Example 8: Take Password Input

**Code:**

|  |
| --- |
| #!/bin/bash read -sp "Enter your password: " pass echo -e "\nYour password is: $pass" |

**Output:**

|  |
| --- |
| Enter your password:  Your password is: linuxsimply |

#### Example 9: Take Timed Input

**Code:**

|  |
| --- |
| #!/bin/bash read -t 5 -p "Enter your name within 5 seconds: " name |

**Output:**

|  |
| --- |
| Enter your name within 5 seconds: aec |

#### Example 10: Find the Length of a String

**Code:**

|  |
| --- |
| #!/bin/bash  str="My name is Tom!" len=${#str} echo "The length of the string is: $len" |

**Output:**

|  |
| --- |
| The length of the string is: 15 |

#### Example 11: Check if Two Strings are Equal

**Code:**

|  |
| --- |
| #!/bin/bash  string1="hello" string2="world"  if [ "$string1" == "$string2" ]; then  echo "The strings are equal." else  echo "The strings are not equal." fi |

**Output:**

|  |
| --- |
| The strings are not equal. |

#### Example 17: Convert All Uppercase Letters in a String to Lowercase

**Code:**

|  |
| --- |
| #!/bin/bash read -p "Enter a string: " str echo "Converted String:" $str | tr '[:upper:]' '[:lower:]' |

**Output:**

|  |
| --- |
| Enter a string: ABCDefgh converted string: abcdefgh |

#### Example 12: Remove All Whitespace from a String

**Code:**

|  |
| --- |
| #!/bin/bash  str=" Hello from Linuxsimply ! ! " str=${str// /} echo "The resultant string: $str" |

**Output:**

|  |
| --- |
| The resultant string: HellofromLinuxsimply!! |

#### Example 13: Reverse a String

**Code:**

|  |
| --- |
| #!/bin/bash  str="Linuxsimply" str=$(echo "$str" | rev) echo "The reversed string: $str" |

**Output:**

|  |
| --- |
| The reversed string: ylpmisxuniL |

#### Example 14: Replace a Word in a Sentence

**Code:**

|  |
| --- |
| #!/bin/bash read -p "Enter a sentence: " str1 read -p "Enter the word to be replaced: " str2 read -p "Enter the new word: " str3 echo "Modified sentence: ${str1/$str2/$str3}" |

**Output:**

|  |
| --- |
| Enter a sentence: I love Linux Enter the word to be replaced: Linux Enter the new word: Linuxsimply Modified sentence: I love Linuxsimply |

#### Example 15: Print Even Numbers from 1 to 10

**Code:**

|  |
| --- |
| #!/bin/bash for (( i=1; i<=10; i++ )) do  if [ $((i%2)) == 0 ]  then  echo $i  fi done |

**Output:**

|  |
| --- |
| 2 4 6 8 10 |

#### Example 16: Calculate the Sum of Digits of a Given Number

|  |
| --- |
| #!/bin/bash read -p "Enter a number: " num sum=0 while [ $num -gt 0 ] do  dig=$((num%10))  sum=$((sum+dig))  num=$((num/10)) done echo "The sum of digits of the given number: $sum" |

**Output:**

|  |
| --- |
| Enter a number: 1567 The sum of digits of the given number: 19 |

#### Example 17: Calculate the Factorial of a Number

**Code:**

|  |
| --- |
| #!/bin/bash read -p "Enter a number: " num temp=1 for (( i=1; i<=$num; i++ )) do  temp=$((temp\*i)) done echo "The factorial of $num is: $temp" |

**Output:**

|  |
| --- |
| Enter a number: 6 The factorial of 6 is: 720 |

#### Example 18: Calculate the Sum of the First “n” Numbers

**Code:**

|  |
| --- |
| #!/bin/bash read -p "Enter a number: " num sum=0 for (( i=1; i<=$num; i++ )) do  sum=$((sum + i)) done echo "Sum of first $num numbers: $sum" |

**Output:**

|  |
| --- |
| Enter a number: 100 Sum of first 100 numbers: 5050 |
|  |

# 

# *19. Programs for printing patterns:*

|  |
| --- |
| **#!/bin/bash**  **# print pyramid**  **p=7;**  **for((m=1; m<=p; m++))**  **do**  **for((a=m; a<=p; a++))**  **do**  **echo -ne " ";**  **done**  **for((n=1; n<=m; n++))**  **do**  **echo -ne "#";**  **done**  **for((i=1; i<m; i++))**  **do**  **echo -ne "#";**  **done**  **# New line**  **echo;**  **done** |

**Output:**

#

###

#####

#######

#########

###########

#############

# *20.Fibonacci Series*

|  |
| --- |
| #!/bin/bash  #Program for Fibonacci Series    N=6   a=0   b=1   echo "The Fibonacci series is : "    for (( i=0; i<N; i++ ))  do      echo -n "$a "      fn=$((a + b))      a=$b      b=$fn  done  # End of for loop |

Output:

Fibonacci Series is:

0

1

1

2

3

5

8

#### Example 21: Find the Smallest and Largest Elements in an Array

**Code:**

|  |
| --- |
| #!/bin/bash  array=(45 25 14 26 28 19 56)  smallest=${array[0]}  largest=${array[0]}  for num in "${array[@]}"; do  if (( num < smallest )); then  smallest=$num  elif (( num > largest )); then  largest=$num  fi  done  echo "Smallest element: $smallest"  echo "Largest element: $largest" |

**Output:**

|  |
| --- |
| Given array: 45 25 14 26 28 19 56 The smallest element: 14 The largest: 56 |

#### Example 22: Sort an Array of Integers in Ascending Order

**Code:**

|  |
| --- |
| #!/bin/bash arr=(24 27 84 11 99) echo "Given array: ${arr[\*]}" arr=($(echo "${arr[\*]}" | tr ' ' '\n' | sort -n | tr '\n' ' ')) echo "Sorted array: ${arr[\*]}" |

**Output:**

|  |
| --- |
| Given array: 24 27 84 11 99 Sorted array: 11 24 27 84 99 |

#### Example 23: Calculate the Average of an Array of Numbers

**Code:**

|  |
| --- |
| #!/bin/bash echo "Enter an array of numbers (separated by space):" read -a arr sum=0 for i in "${arr[@]}" do  sum=$((sum+i)) done avg=$((sum/${#arr[@]})) echo "Average of the array elements: $avg" |

**Output:**

|  |
| --- |
| Enter an array of numbers (separated by space): 23 45 11 99 100 Average of the array elements: 55 |

#### Example 24: Find the Length of an Array

**Code:**

|  |
| --- |
| #!/bin/bash arr=(24 27 84 11 99) echo "Given array: ${arr[\*]}" len=${#arr[@]} echo "The length of the array: $len" |

**Output:**

|  |
| --- |
| Given array: 24 27 84 11 99 The length of the array: 5 |

#### Example 25: Check if a String is a Palindrome

**Code:**

|  |
| --- |
| #!/bin/bash Palindrome () {  s=$1  if [ "$(echo $s | rev)" == "$str" ]  then  echo "The string is a Palindrome"  else  echo "The string is not a palindrome"  fi } read -p "Enter a string: " str Palindrome "$str" |

**Output:**

|  |
| --- |
| Enter a string: wow The string is a Palindrome |

#### Example 26: Check if a Number is Prime

**Code:**

|  |
| --- |
| #!/bin/bash Prime () {  num=$1  if [ $num -lt 2 ]  then  echo "The number $num is Not Prime"  return  fi  for (( i=2; i<=$num/2; i++ ))  do  if [ $((num%i)) -eq 0 ]  then  echo "The number $num is Not Prime"  return  fi  done  echo "The number $num is Prime" } read -p "Enter a number: " num  Prime "$num" |

**Output:**

|  |
| --- |
| Enter a number: 2 The number 2 is Prime |

#### Example 27: Grading System

**Code:**

|  |
| --- |
| Grade() {  score=$1  if (( $score >= 80 )); then  grade="A+"  elif (( $score >= 70 )); then  grade="A"  elif (( $score >= 60 )); then  grade="B"  elif (( $score >= 50 )); then  grade="C"  elif (( $score >= 40 )); then  grade="D"  else  grade="F"  fi  echo "The grade for mark $s is $grade" } read -p "Enter a score between 1-100:" s Grade $s |

**Output:**

|  |
| --- |
| Enter a score between 1-100:76 "The grade for mark 76 is A" |

#### 28.Create a New File and Write Text Inside

**Code:**

|  |
| --- |
| #!/bin/bash read -p "Enter the file name: " file echo "Enter text to write:" read text echo "$text" > "$file" echo "-----------------------------------" echo "The File $file is created!" |

**Output:**

|  |
| --- |
| Enter the file name: text\_file1.txt Enter text to write: In English, there are three articles: a, an, and the. The File text\_file1.txt is created! |

#### 29.Delete a Given File If It Exists

**Code:**

|  |
| --- |
| #!/bin/bash read -p "Enter the file name for deletion: " file if [ -f $file ] then  rm $file  echo "The file $file deleted successfully!" else  echo "Error! The file $file does not exist." fi |

**Output:**

|  |
| --- |
| Enter the file name for deletion: article1.txt The file article1.txt deleted successfully! |

#### 30.Renames a File from Script

**Code:**

|  |
| --- |
| #!/bin/bash read -p "Enter the file name: " file read -p "Enter new file name: " new\_file if [ -f $file ] then  mv "$file" "$new\_file"  echo "The file $file has been renamed as $new\_file!" else  echo "Error! The file $file does not exist." fi |

**Output:**

|  |
| --- |
| Enter the file name: p.txt Enter new file name: d.txt The file poem.txt has been renamed as d.txt! |

#### 31.Sets the Permissions of a Directory for the Owner

**Code:**

|  |
| --- |
| #!/bin/bash read -p "Enter the directory name: " dir if [ -d $dir ]; then  chmod u+rwx $dir  echo "Directory permissions have been updated!" else  echo "Error! The directory $dir does not exist." fi |

**Output:**

|  |
| --- |
| Enter the directory name: Documents Directory permissions have been updated! |

#### 32.Change the Overall Permissions of a File

**Code:**

|  |
| --- |
| #!/bin/bash read -p "Enter the file name: " file read -p "Enter new permissions in Absolute Mode: " permissions if [ -f $file ]; then  sudo chmod $permissions $file  echo "Permissions for the file $file has been changed!" else  echo "Error! The file $file does not exist." fi |

**Output:**

|  |
| --- |
| Enter the file name: d.txt Enter new permissions in Absolute Mode: 777 [sudo] password for aec:  Permissions for the file daffodils.txt has been changed! |

#### 33.Process Management: Check if a Process is Running

**Code:**

|  |
| --- |
| #!/bin/bash read -p "Enter process name: " process if pgrep $process > /dev/null then  echo "Process is running." else  echo "Process is not running." fi |

**Output:**

|  |
| --- |
| Enter process name: bash Process is running. |

#### 34.Check the Number of Logged-in Users

**Code:**

|  |
| --- |
| #!/bin/bash users=$(who | wc -l) echo "Number of currently logged-in users: $users" |

**Output:**

|  |
| --- |
| Number of currently logged-in users: 2 |

#### 35.Check the Operating System Information

**Code:**

|  |
| --- |
| #!/bin/bash  os\_name=$(uname -s) os\_release=$(uname -r) os\_version=$(cat /etc/\*-release | grep VERSION\_ID | cut -d '"' -f 2) os\_arch=$(uname -m)  echo "OS Name: $os\_name" echo "OS Release: $os\_release" echo "OS Version: $os\_version" echo "OS Architecture: $os\_arch" |

**Output:**

|  |
| --- |
| OS Name: Linux OS Release: 5.19.0-38-generic OS Version: 22.04 OS Architecture: x86\_64 |

#### 36.Check the System’s Memory Usage

**Code:**

|  |
| --- |
| #!/bin/bash mem=$(free -m | awk 'NR==2{printf "%.2f%%", $3\*100/$2}') echo "Current Memory Usage: $mem" |

**Output:**

|  |
| --- |
| Current Memory Usage: 72.48% |

#### 37.Check the System’s Network Information

**Code:**

|  |
| --- |
| #!/bin/bash echo " System's network information:-" ip=$(hostname -I) echo "IP Address: $ip" gw=$(ip route | awk '/default/ { print $3 }') echo "Gateway: $gw" dns=$(grep "nameserver" /etc/resolv.conf | awk '{print $2}') echo "DNS Server: $dns" |

**Output:**

|  |
| --- |
| System's network information:- IP Address: 192.168.81.109  Gateway: 192.168.80.1 DNS Server: 127.0.0.53 |

#### 38. Automated Backup

**Code:**

|  |
| --- |
| #!/bin/bash read -p "Enter path of the directory to backup: " source\_dir read -p "Enter destination path for backup: " backup\_dir date=$(date +%Y-%m-%d) backup\_file="backup-$date.tar.gz" # Create backup directory if it doesn't exist if [ ! -d "$backup\_dir" ]; then  mkdir -p "$backup\_dir" fi # Create backup archive tar -czf "$backup\_dir/$backup\_file" "$source\_dir" echo "Completed Creating backup at: $backup\_dir." |

**Output:**

|  |
| --- |
| Enter path of the directory to backup: /home/aec/Documents Enter destination path for backup: /home/aec/Desktop Completed Creating backup at: /home/aec/Desktop. |

#### 39.Create a New User and Add to Sudo Group

**Code:**

|  |
| --- |
| #!/bin/bash  read -p "Enter username: " username read -p "Enter password: " password  useradd -m -s /bin/bash -p $(openssl passwd -1 $password) $username if [ $? -eq 0 ]; then usermod -a -G sudo $username  mkdir /home/$username/mydir chown -R $username:$username /home/$username/mydir usermod -d /home/$username/mydir $username  echo "$username ALL=(ALL) NOPASSWD:ALL" >> /etc/sudoers  echo "User $username created successfully!" echo "User $username added to sudo group!" else echo "Error while creating user!" fi |

**Output:**

|  |
| --- |
| Enter username:aec Enter password: aec123 User aec created successfully! User aec added to sudo group! |