



S.B. JAIN INSTITUTE OF TECHNOLOGY MANAGEMENT & RESEARCH, NAGPUR

Practical 03

Aim: Automate student marksheets generation, system information display, Fibonacci and prime number generation, and file management operations using shell scripts to enhance computational efficiency and user interaction.

Name:Dhanshri Gaurkhede

USN:CM24013

Semester / Year:

Academic Session:

Date of Performance:

Date of Submission:

❖ **Aim:** Automate student marksheet generation, system information display, Fibonacci and prime number generation, and file management operations using shell scripts to enhance computational efficiency and user interaction.

❖ **Tasks to be done in this Practical.**

- a) Write a shell script to generate mark- sheet of a student. Take 3 subjects, calculate and display total marks, percentage and Class obtained by the student.
- b) Write a menu driven shell script which will print the following menu and execute the given task.
 - Display calendar of current month.
 - Display today's date and time.
 - Display usernames those are currently logged in the system.
 - Display your terminal number
- c) Write a shell script which will generate first n Fibonacci numbers like: 1, 1, 2, 3, 5, 13
- d) Write a shell script which will accept a number b and display first n prime numbers as output.
- e) Write menu driven program for file handling activity
 - Creation of file.
 - Write content in the file.
 - Upend file content.
 - Delete file content

❖ **Objectives:**

1. Automate marksheet generation with total, percentage, and class classification.
2. Develop menu-driven scripts for system information and file operations.
3. Generate Fibonacci and prime numbers for user-defined inputs.

❖ **Requirements:**

✓ **Hardware Requirements:**

- Processor: Minimum 1 GHz
- RAM: 512 MB or higher
- Storage: 100 MB free space

✓ **Software Requirements:**

- Operating System: Linux/Unix-based
- Shell: Bash 4.0 or higher
- Text Editor: Nano, Vim, or any preferred editor



❖ **Theory:**

Shell scripting is a powerful way to automate repetitive tasks and manage system operations efficiently. It allows users to write programs using shell commands and scripting constructs. Shell scripts are interpreted line-by-line by a shell interpreter, making them ideal for administrative tasks, file management, and system automation. This practical encompasses a variety of real-world scenarios that demonstrate the utility of shell scripting for computing tasks and resource management.

1. Marksheets Generation

This script takes input marks for three subjects, calculates the total marks, percentage, and determines the class of the student based on predefined conditions. Conditional statements (if-else) are used to classify the performance into distinction, first class, second class, or fail. This exercise emphasizes the use of arithmetic operations and decision-making constructs.

Key concepts include:

- Reading user input using read
- Arithmetic operations with \$((expression))
- Conditional statements for decision-making

2. Menu-Driven Script for System Information

Menu-driven scripts enhance user interaction by presenting a list of options for performing different tasks. In this practical, options are provided to display the calendar of the current month, the current date and time, logged-in users, and the terminal number. The script utilizes looping constructs (while) and case statements for structured flow control.

Commands used:

- cal for displaying the calendar
- date for showing current date and time
- who to list logged-in users
- tty to identify the terminal



3. Fibonacci Number Generation

Fibonacci numbers are a sequence where each term is the sum of the two preceding ones. The script uses iterative constructs (for loop) to generate n terms based on user input. This practical illustrates the use of loop control and variable swapping to generate series data efficiently.

4. Prime Number Display

This script accepts an integer n and outputs the first n prime numbers. A nested loop checks divisibility to determine if a number is prime. The practical demonstrates logic building for number-theoretic operations using loops and conditionals.

5. Menu-Driven File Management

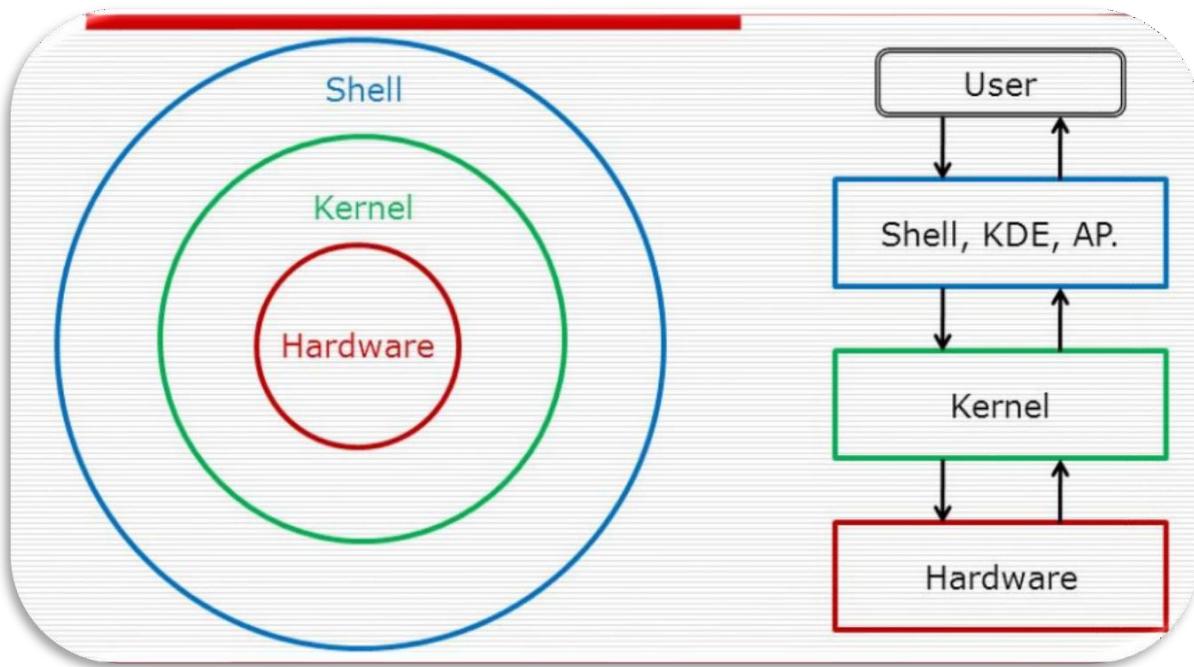
The file handling script enables users to create, write, append, and delete file content. The case construct manages different file operations.

Commands include:

- touch to create files
- cat for writing and appending content
- rm for deleting files

This exercise emphasizes text manipulation, input handling, and file control mechanisms in Unix-like environments.

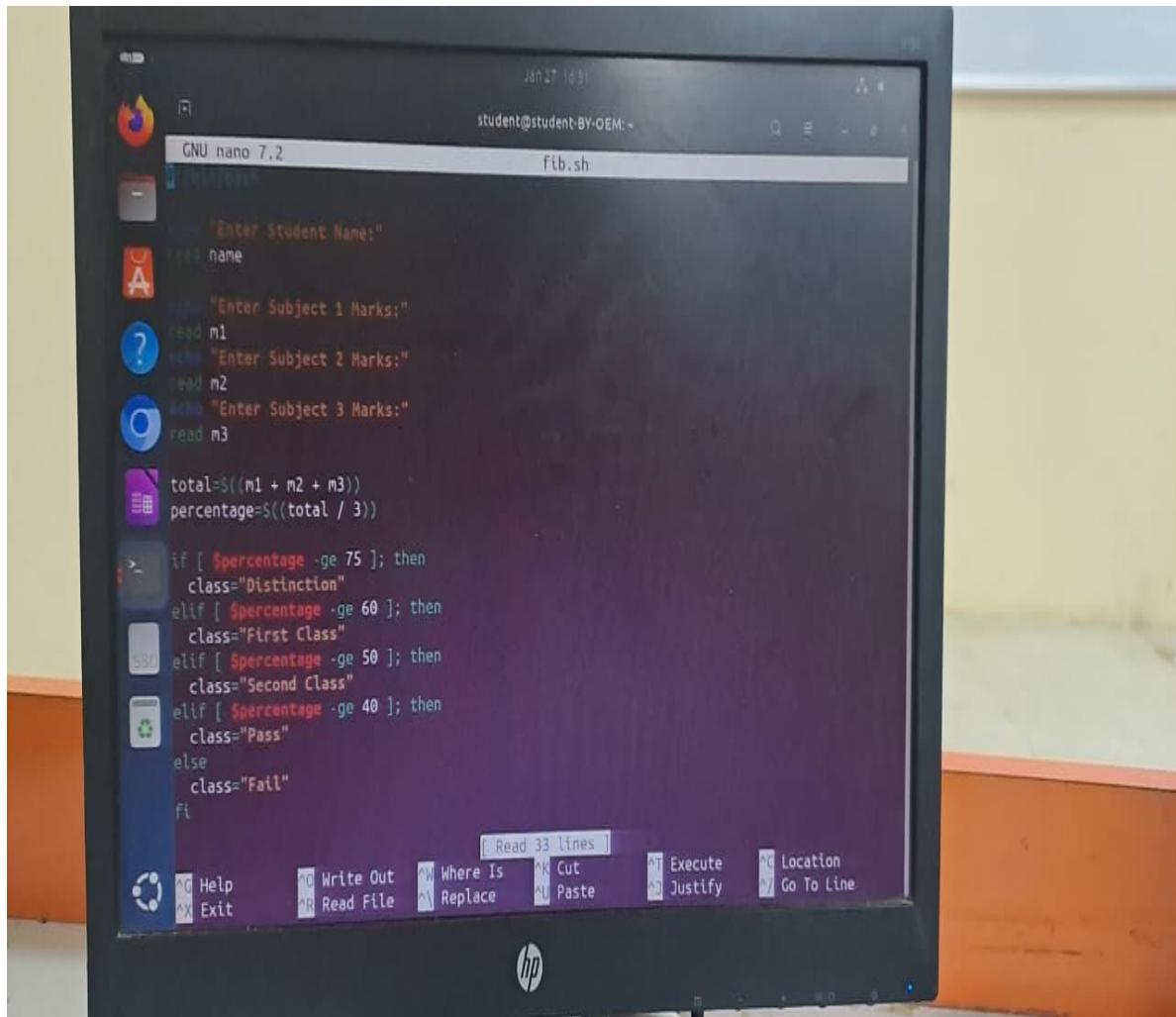
Diagrammatical View of Shell



❖ CODES

1. Write a shell script to generate mark- sheet of a student. Take 3 subjects, calculate and display total marks, percentage and Class obtained by the student.

Output 1:

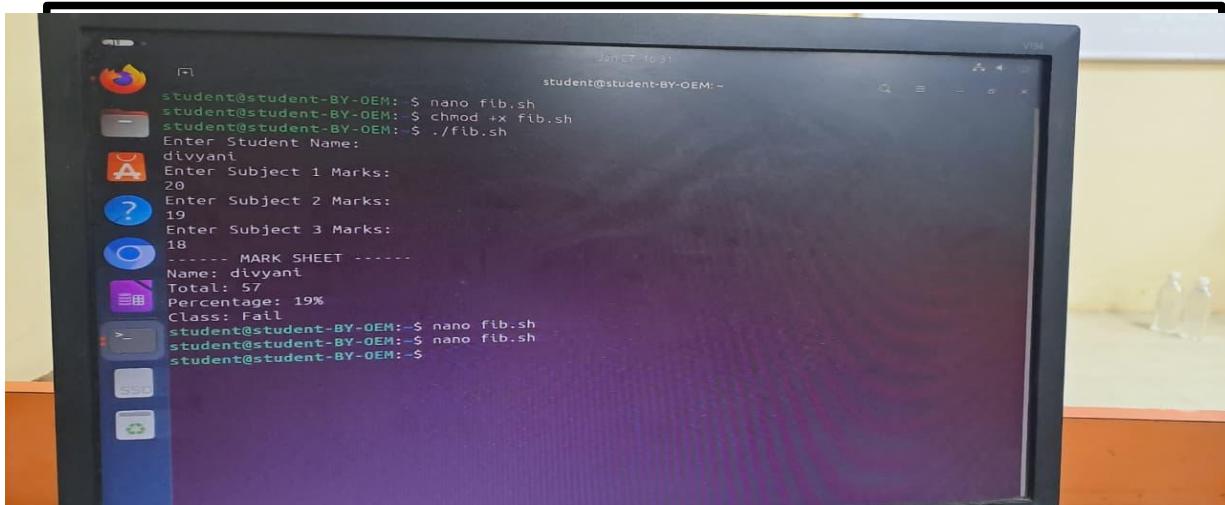


```
student@student-BY-OEM:~$ nano fib.sh
student@student-BY-OEM:~$ cat fib.sh
#!/bin/bash
echo "Enter Student Name:"
read name
echo "Enter Subject 1 Marks:"
read m1
echo "Enter Subject 2 Marks:"
read m2
echo "Enter Subject 3 Marks:"
read m3

total=$((m1 + m2 + m3))
percentage=$((total / 3))

if [ $percentage -ge 75 ]; then
    class="Distinction"
elif [ $percentage -ge 60 ]; then
    class="First Class"
elif [ $percentage -ge 50 ]; then
    class="Second Class"
elif [ $percentage -ge 40 ]; then
    class="Pass"
else
    class="Fail"
fi

echo "Total Marks: $total"
echo "Percentage: $percentage"
echo "Class: $class"
```



2. Write a menu driven shell script which will print the following menu and execute the given task.

- Display calendar of current month.
- Display today's date and time.
- Display usernames those are currently logged in the system.
- Display your terminal number

A screenshot of a Linux desktop environment showing a terminal window. The terminal window title is 'calender.sh'. The terminal content shows a shell script with a menu. The script uses 'echo' to print the menu options (1. Display calendar of current month, 2. Display today's date and time, 3. Display logged in users, 4. Display terminal number) and a case statement to handle user input. The script reads input from the user using 'read ch' and then executes the corresponding command based on the choice. The terminal prompt is 'student@student-BY-OEM: ~'.

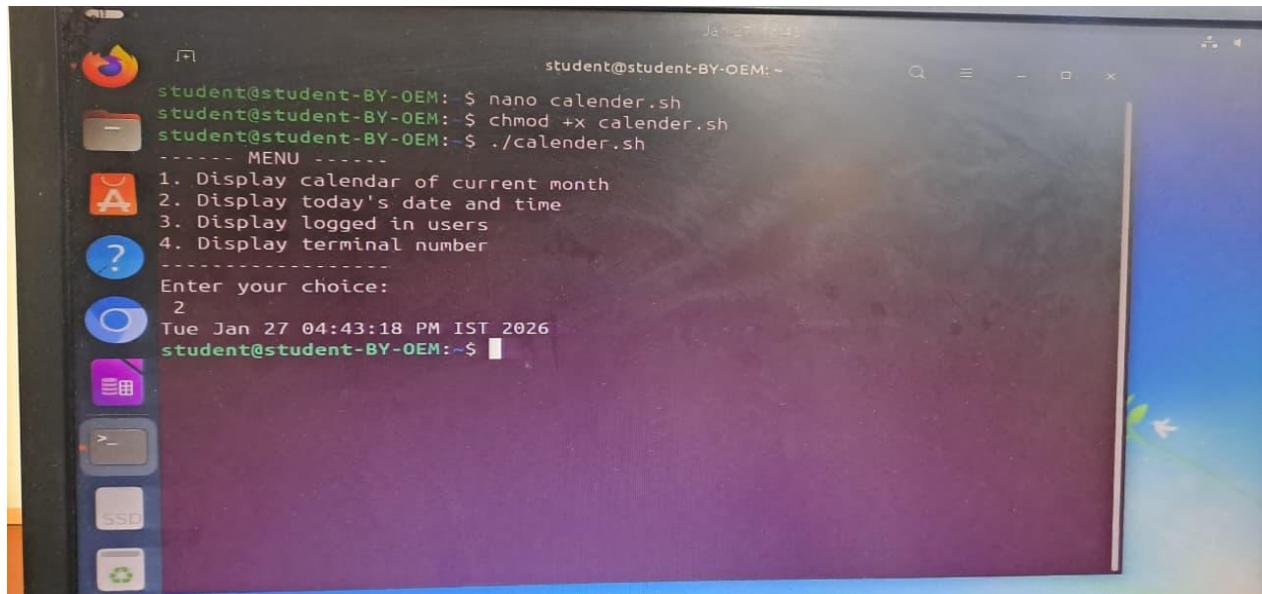
```
GNU nano 7.2
#!/bin/bash

echo "----- MENU -----"
echo "1. Display calendar of current month"
echo "2. Display today's date and time"
echo "3. Display logged in users"
echo "4. Display terminal number"
echo "-----"

echo "Enter your choice:"
read ch

case $ch in
    1) cal ;;
    2) date ;;
    3) who ;;
    4) tty ;;
    *) echo "Invalid Choice" ;;
esac
```

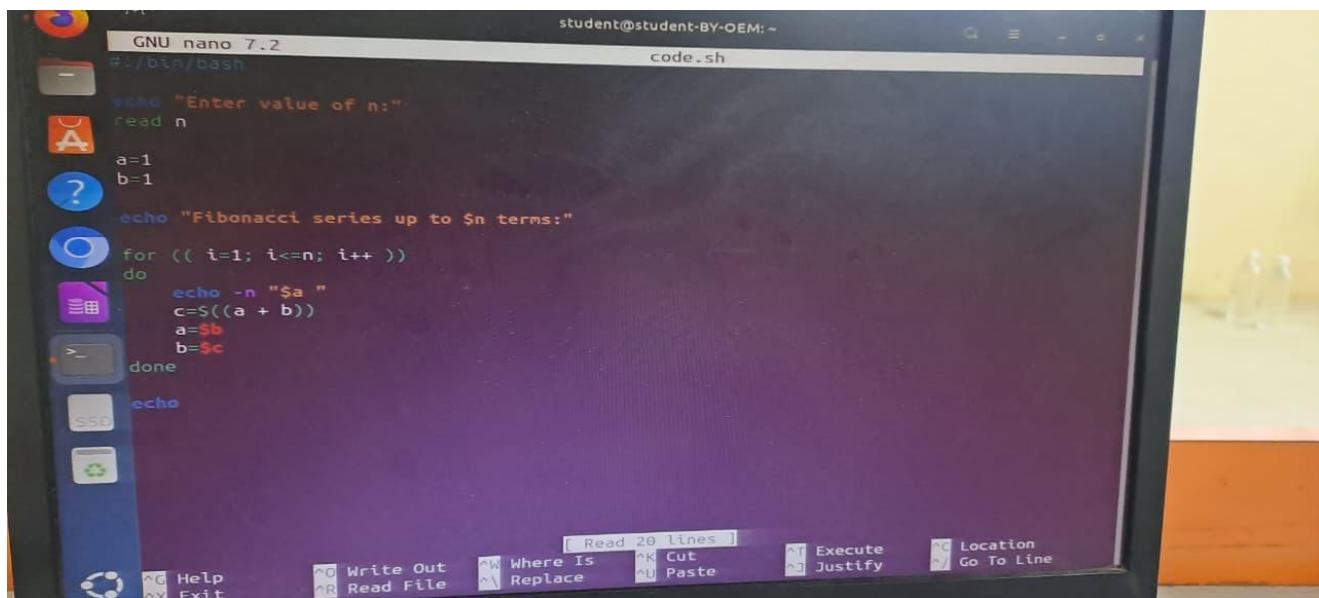
Output 2:



A screenshot of a Linux desktop environment showing a terminal window. The terminal title is "student@student-BY-OEM: ~". The command entered is "nano calender.sh", followed by "chmod +x calender.sh", and finally "./calender.sh". The script displays a menu with four options: 1. Display calendar of current month, 2. Display today's date and time, 3. Display logged in users, and 4. Display terminal number. The user enters choice 2, which outputs the date and time: "Tue Jan 27 04:43:18 PM IST 2026".

```
student@student-BY-OEM: $ nano calender.sh
student@student-BY-OEM: $ chmod +x calender.sh
student@student-BY-OEM: ~$ ./calender.sh
----- MENU -----
1. Display calendar of current month
2. Display today's date and time
3. Display logged in users
4. Display terminal number
Enter your choice:
2
Tue Jan 27 04:43:18 PM IST 2026
student@student-BY-OEM: ~$
```

3. Write a shell script which will generate first n Fibonacci numbers like:
1, 1, 2, 3, 5, 13



A screenshot of a terminal window showing the nano 7.2 editor. The file is named "code.sh". The script code generates a Fibonacci series up to n terms. It starts by prompting for the value of n, initializes a=1 and b=1, then enters a loop where it prints the current value of a, calculates the next value c as the sum of a and b, and updates a and b for the next iteration. The loop continues until i is greater than or equal to n.

```
student@student-BY-OEM: ~
GNU nano 7.2
#!/bin/bash

read -p "Enter value of n:" n

a=1
b=1

echo "Fibonacci series up to $n terms:"
for (( i=1; i<=n; i++ ))
do
    echo -n "$a "
    c=$((a + b))
    a=$b
    b=$c
done
echo
```

Output 3:

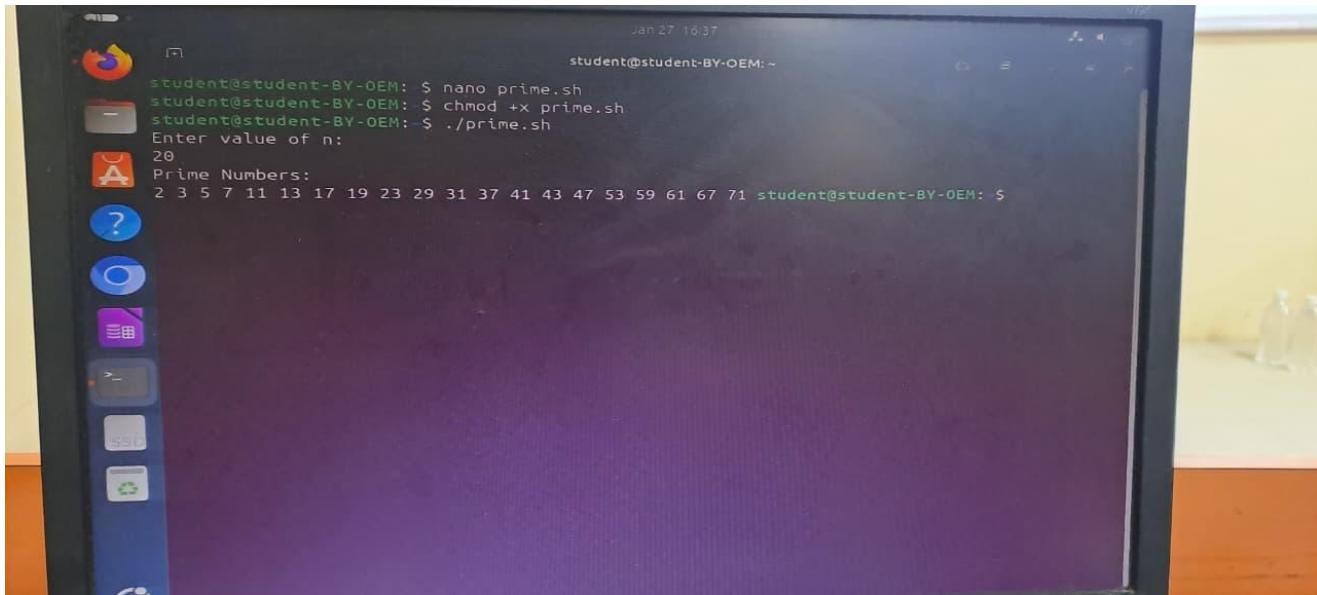
```
student@student-BY-OEM: $ nano code.sh
student@student-BY-OEM: $ chmod +x code.sh
Command 'chomd' not found, did you mean:
  command 'chmod' from deb coreutils (9.4-3ubuntu6.1)
Try: sudo apt install <deb name>
student@student-BY-OEM: $ chmod +x code.sh
student@student-BY-OEM: $ ./code.sh
Enter value of n:
4
Fibonacci series up to 4 terms:
1 1 2 3
student@student-BY-OEM: $
```

4. Write a shell script which will accept a number b and display first n prime numbers as output.

```
student@student-BY-OEM: ~
GNU nano 7.2
prime.sh

#!/bin/bash
# Enter value of n:
read n
count=0
num=2
echo "Prime Numbers:"
while [ $count -lt $n ]
do
    flag=1
    for ((i=2;i<=num/2;i++))
    do
        if [ $((num%i)) -eq 0 ]; then
            flag=0
            break
        fi
    done
    if [ $flag -eq 1 ]; then
        echo -n "$num "
        count=$((count+1))
    fi
done
```

Output 4:

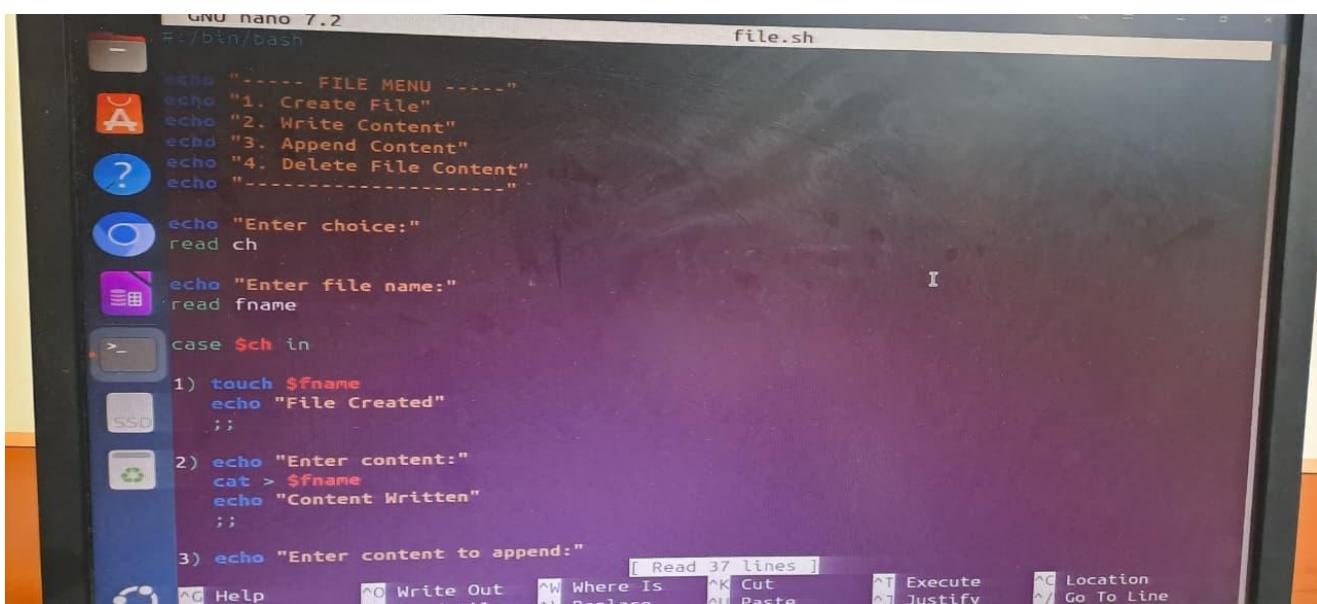


A screenshot of a Linux desktop environment. A terminal window titled "student@student-BY-OEM: ~" is open, showing the output of a shell script named "prime.sh". The script prompts for a value of n (20), lists prime numbers from 2 to 71, and ends with a final "\$". The desktop background is purple, and the taskbar shows various icons.

```
student@student-BY-OEM: $ nano prime.sh
student@student-BY-OEM: $ chmod +x prime.sh
student@student-BY-OEM: $ ./prime.sh
Enter value of n:
20
Prime Numbers:
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 student@student-BY-OEM: $
```

5. Write menu driven program for file handling activity

- Creation of file.
- Write content in the file.
- Upend file content.
- Delete file content,



A screenshot of a terminal window titled "file.sh" in the "nano 7.2" editor. The script displays a menu with options for creating files, writing content, appending content, and deleting file content. It also prompts for a choice and a file name, and handles cases for each option. The terminal shows the script's code and some command-line interface options at the bottom.

```
GNU nano 7.2
#!/bin/bash

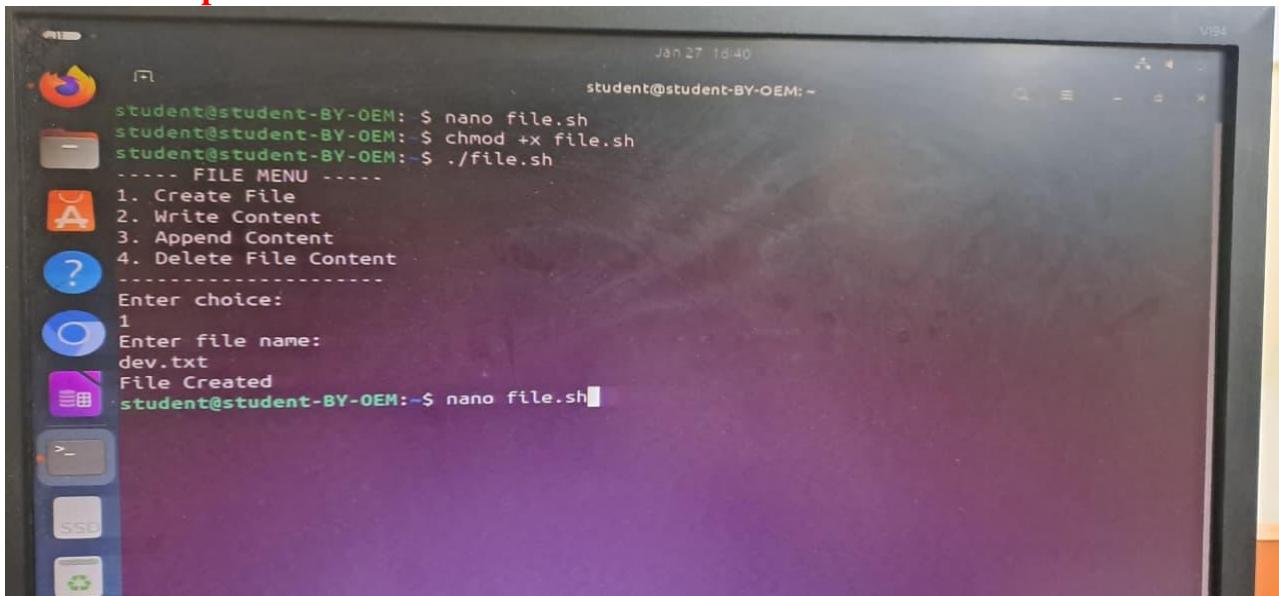
echo "----- FILE MENU -----"
echo "1. Create File"
echo "2. Write Content"
echo "3. Append Content"
echo "4. Delete File Content"
echo "-----"

echo "Enter choice:"
read ch

echo "Enter file name:"
read fname

case $ch in
    1) touch $fname
        echo "File Created"
        ;;
    2) echo "Enter content:"
        cat > $fname
        echo "Content Written"
        ;;
    3) echo "Enter content to append:"
```

Output 5:



```
student@student-BY-OEM:~$ nano file.sh
student@student-BY-OEM:~$ chmod +x file.sh
student@student-BY-OEM:~$ ./file.sh
----- FILE MENU -----
1. Create File
2. Write Content
3. Append Content
4. Delete File Content
-----
Enter choice:
1
Enter file name:
dev.txt
File Created
student@student-BY-OEM:~$ nano file.sh
```

❖ **Conclusion:** In this practical, we conclude that shell scripting efficiently automates tasks like marksheet generation, system information display, number computations, and file management, enhancing system operations and user interaction through command-line utilities.

❖ **Discussion Questions:**

1. **What is the purpose of using shell scripting in this practical?**
2. **Which command is used to display the current date and time?**
3. **How does the script calculate the Fibonacci sequence?**
4. **Which command is used to create a file in the file management script?**
5. **How does the prime number script determine if a number is prime?.**

❖ **References:**

https://www.tutorialspoint.com/unix/shell_scripting.html
<https://www.javatpoint.com/shell-scripting-tutorial>

Date: ____ / ____ /2026

Signature

Course Coordinator
B.Tech CSE(AIML)