



## **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:**

**USN:**

**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 marksheets 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**

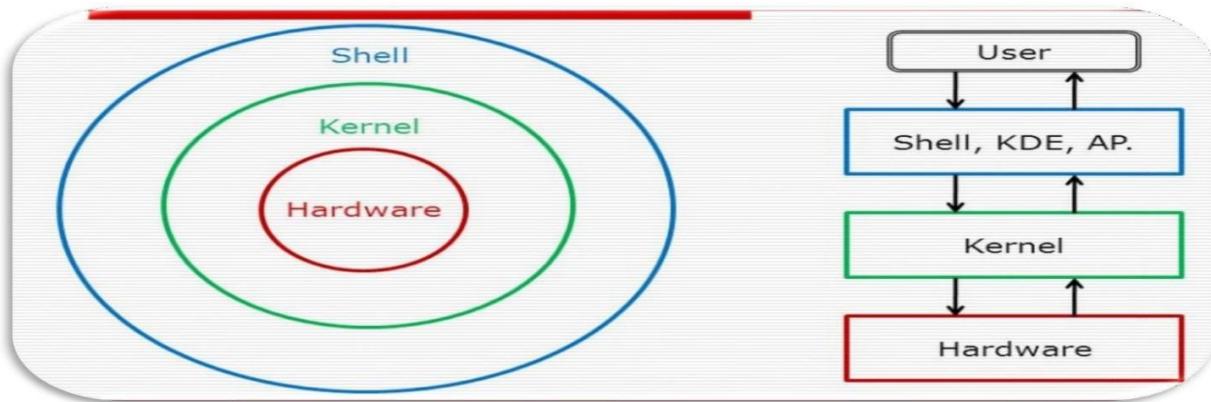
The file handling script enables users to create, write, append, and delete file content. The case construct manages different file operations.

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.

```

MINGW64:/c/Users/Admin
Student@DESKTOP-JJ8CFNB MINGW64 ~
$ echo "Enter marks for Subject 1:"
read m1
echo "Enter marks for Subject 2:"
read m2
echo "Enter marks for Subject 3:"
read m3
# Calculate total and percentage
total=$((m1 + m2 + m3))
percentage=$((total * 100 / 300))
echo "Student Name : $name"
echo "Total Marks : $total / 300"
echo "Percentage : $percentage%"
if [ $percentage -ge 75 ]
then
    echo "Class Obtained: Distinction"
elif [ $percentage -ge 60 ]
then
    echo "Class Obtained: First Class"
elif [ $percentage -ge 50 ]
then
    echo "Class Obtained: Second Class"
elif [ $percentage -ge 35 ]
then
    echo "Class Obtained: Pass"
else
    echo "Class Obtained: Fail"
fi
Enter marks for Subject 1:
40
Enter marks for Subject 2:
40
Enter marks for Subject 3:
40
Student Name : winget
Total Marks : 120 / 300
Percentage : 40%
Class Obtained: Pass
Student@DESKTOP-JJ8CFNB MINGW64 ~
$ |

```



ENG 11:02 AM IN 1/31/2026

**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

```
MINGW64/c/Users/Admin
Student@DESKTOP-JJ8CFNB MINGW64 ~
$ while true
do
echo -n "Enter your choice: "
read choice

case $choice in
  1)
    date +"%B %Y"
    ;;
  2)
    date
    ;;
  3)
    whoami
    ;;
  4)
    tty
    ;;
  5)
    echo "Exiting program..."
    exit
    ;;
  *)
    echo "Invalid choice. Please try again."
    ;;
esac

echo
done
Enter your choice: 1
January 2026

Enter your choice: 2
Sat Jan 31 11:05:43 IST 2026

Enter your choice: 3
Student

Enter your choice: 4
/dev/pty0

Enter your choice:
```



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

```
MINGW64/c/Users/satis/OneDrive/Desktop
satis@SAMYAK MINGW64 ~/OneDrive/Desktop (master)
$ echo -n "Enter the value of n: "
read n

a=1
b=1

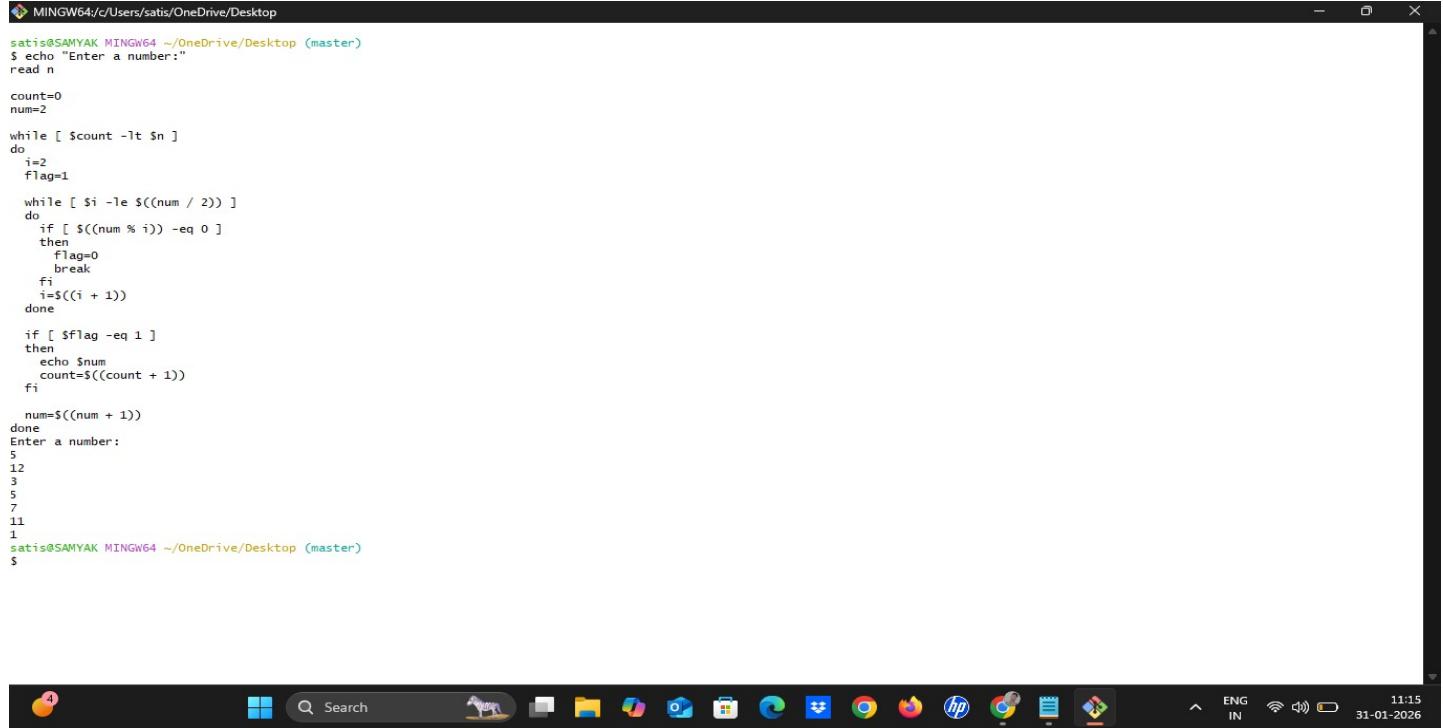
echo "Fibonacci Series:"
for (( i=1; i<=n; i++ ))
do
  echo -n "$a "
  fn=$((a + b))
  a=$b
  b=$fn
done

echo
Enter the value of n: 7
Fibonacci Series:
1 1 2 3 5 8 13

satis@SAMYAK MINGW64 ~/OneDrive/Desktop (master)
$
```



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



```

MINGW64:/c/Users/satis/OneDrive/Desktop
satis@SAMYAK MINGW64 ~/OneDrive/Desktop (master)
$ echo "Enter a number:"
read n

count=0
num=2

while [ $count -lt $n ]
do
    i=2
    flag=1

    while [ $i -le ${num / 2} ]
    do
        if [ $((num % i)) -eq 0 ]
        then
            flag=0
            break
        fi
        i=$((i + 1))
    done

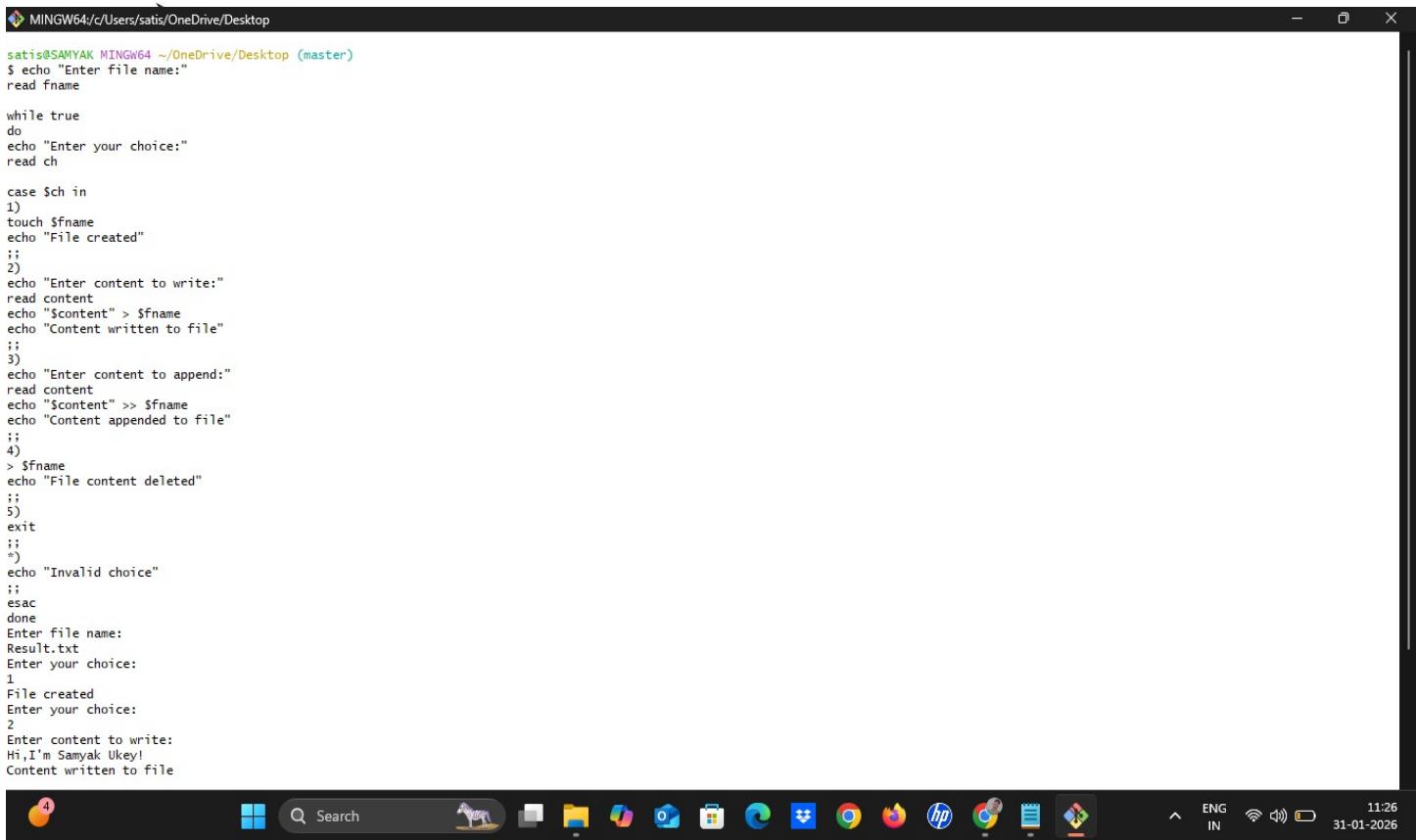
    if [ $flag -eq 1 ]
    then
        echo $num
        count=$((count + 1))
    fi

    num=$((num + 1))
done
Enter a number:
5
12
3
5
7
11
1
satis@SAMYAK MINGW64 ~/OneDrive/Desktop (master)
$ 
```

The screenshot shows a Windows desktop environment with a terminal window open in the MinGW64 shell. The script prompts the user to enter a number (5 in this case) and then prints the first 5 prime numbers: 2, 3, 5, 7, and 11. The taskbar at the bottom shows various application icons.

**5. Write menu driven program for file handling activity**

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



```

MINGW64:/c/Users/satis/OneDrive/Desktop
satis@SAMYAK MINGW64 ~/OneDrive/Desktop (master)
$ echo "Enter file name:"
read fname

while true
do
echo "Enter your choice:"
read ch

case $ch in
1)
touch $fname
echo "File created"
;;
2)
echo "Enter content to write:"
read content
echo "$content" > $fname
echo "Content written to file"
;;
3)
echo "Enter content to append:"
read content
echo "$content" >> $fname
echo "Content appended to file"
;;
4)
> $fname
echo "File content deleted"
;;
5)
exit
;;
*)
echo "Invalid choice"
;;
esac
done
Enter file name:
Result.txt
Enter your choice:
1
File created
Enter your choice:
2
Enter content to write:
Hi,I'm Samyak Ukey!
Content written to file
$ 
```

The screenshot shows a Windows desktop environment with a terminal window open in the MinGW64 shell. The script creates a file named 'Result.txt', writes the string 'Hi,I'm Samyak Ukey!' to it, and then appends the same string to the end of the file. It then deletes the file and exits. The taskbar at the bottom shows various application icons.

```

MINGW64/c/Users/satis/OneDrive/Desktop
case $ch in
1)
touch $fname
echo "File created"
;;
2)
echo "Enter content to write:"
read content
echo "$content" > $fname
echo "Content written to file"
;;
3)
echo "Enter content to append:"
read content
echo "$content" >> $fname
echo "Content appended to file"
;;
4)
> $fname
echo "File content deleted"
;;
5)
exit
;;
*)
echo "Invalid choice"
;;
esac
done
Enter file name:
Result.txt
Enter your choice:
1
File created
Enter your choice:
2
Enter content to write:
Hi, I'm Samyak Ukey!
Content written to file
Enter your choice:
3
Enter content to append:
I'm working on Git Bash.
Content appended to file
Enter your choice:
4
File content deleted
Enter your choice:

```

❖ **Conclusion:** In this practical, we conclude that shell scripting efficiently automates tasks like marksheets 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?**

**Ans:** Shell scripting in this practical automates tasks like marksheets generation, Fibonacci sequence calculation, system information display, and file management. It improves efficiency by reducing manual interventions and allows easy automation of routine tasks using simple commands and logic.

2. **Which command is used to display the current date and time?**

**Ans:** The date command is used to display the current date and time. It provides a formatted output, showing the system's current time, date, and related information depending on the user's locale settings.

3. **How does the script calculate the Fibonacci sequence?**

**Ans:** The Fibonacci sequence script uses a loop to iteratively calculate each number by summing the previous two numbers in the sequence, starting from 0 and 1. This continues until the specified number of terms is generated, making use of simple arithmetic operations.

4. **Which command is used to create a file in the file management script? Ans:** The touch command is used to create an empty file or update the timestamp of an existing file. This is a simple and effective way to initiate a file before performing further operations like writing or appending content.

5. **How does the prime number script determine if a number is prime? Ans:** The prime number script checks for divisibility by iterating through potential divisors up to the square root of the number. If no divisor other than 1 and the number itself is found, the number is classified as prime.

❖ **References:** [https://www.tutorialspoint.com/unix/shell\\_scripting.html](https://www.tutorialspoint.com/unix/shell_scripting.html)  
<https://www.javatpoint.com/shell-scripting-tutorial>

*Operating System Lab (N-PCCCD401P)*

Date: \_\_\_\_\_ / \_\_\_\_\_ /2026

\_\_\_\_\_

**Signature**

Course

Coordinator

B.Tech CSE(DS)