**INSTITUTE OF ENGINEERING & TECHNOLOGY**

**MOHANLAL SUKHADIA UNIVERSITY, UDAIPUR**

**(RAJ.)**

***DEPARTMENT OF COMPUTER SCIENCE ENGINEERING***

***B. Tech - IV SEMESTER***

****

**Session 2022-23**

**Linux Shell Programming Lab**

**LABORATORY MANUAL**

**BT4CS10-CP04**

**Prepared by:**

**Shivam Chouhan**

**Content**

|  |  |  |
| --- | --- | --- |
| S.No. | Topic | Page No. |
| 1. | Introduction to Linux Shell & Shell Scripting | 3 |
| 2. | Use of basic Unix Commands | 6 |
| 3. | Script to perform integer arithmetic operations | 9 |
| 4. | Script to display first ten natural numbers | 10 |
| 5. | Script to find factorial | 11 |
| 6. | Script to check whether number is prime or not | 12 |
| 7. | Script to find sum of digits of given number | 13 |
| 8. | Script to check Armstrong number | 14 |
| 9. | Script to calculate percentage | 15 |
| 10. | Script to check whether number is positive, negative or zero | 16 |
| 11. | Script to check leap year | 17 |
| 12. | Script to generate Fibonacci series | 18 |
| 13. | Script to swap two numbers | 19 |
| 14. | Script to generate pyramid using \* | 20 |
| 15. | Script to print greatest and smallest number | 21 |
| 16. | Introduction to arrays | 22 |
| 17. | Script to search a particular element in array | 28 |
| 18. | Script to read & print elements of array | 29 |
| 19. | Script to find sum of all array elements | 30 |

# Theory

# Introduction to Linux Shell and Shell Scripting

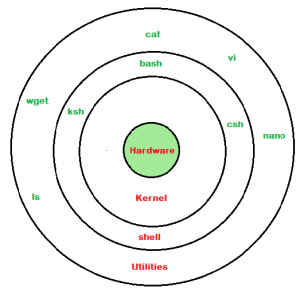
## **What is Kernel?**

The kernel is a computer program that is the core of a computer’s operating system, with complete control over everything in the system. It manages the following resources of the Linux system –

* File management
* Process management
* I/O management
* Memory management
* Device management etc.

## **What is Shell?**

A shell is a special user program that provides an interface for the user to use operating system services. Shell accepts human-readable commands from users and converts them into something which the kernel can understand. It is a command language interpreter that executes commands read from input devices such as keyboards or from files. The shell gets started when the user logs in or starts the terminal.



Shell is broadly classified into two categories –

* Command Line Shell
* Graphical shell

### **Command Line Shell**

Shell can be accessed by users using a command line interface. A special program called Terminal in Linux/macOS, or Command Prompt in Windows OS is provided to type in the human-readable commands such as “cat”, “ls” etc. and then it is being executed. The result is then displayed on the terminal to the user.

### **Graphical Shells**

Graphical shells provide means for manipulating programs based on the graphical user interface (GUI), by allowing for operations such as opening, closing, moving, and resizing windows, as well as switching focus between windows.

There are several shells are available for Linux systems like –

* **BASH (Bourne Again SHell)** – It is the most widely used shell in Linux systems. It is used as default login shell in Linux systems and in macOS. It can also be installed on Windows OS.
* **CSH (C SHell)** – The C shell’s syntax and its usage are very similar to the C programming language.
* **KSH (Korn SHell)** – The Korn Shell was also the base for the POSIX Shell standard specifications etc.

Each shell does the same job but understands different commands and provides different built-in functions.

## What is a terminal?

A program which is responsible for providing an interface to a user so that he/she can access the shell. It basically allows users to enter commands and see the output of those commands in a text-based interface. Large scripts that are written to automate and perform complex tasks are executed in the terminal.

## ****Shell Scripting****

Usually, shells are interactive, which means they accept commands as input from users and execute them. However, sometimes we want to execute a bunch of commands routinely, so we have to type in all commands each time in the terminal.

As a shell can also take commands as input from file, we can write these commands in a file and can execute them in shell to avoid this repetitive work. These files are called **Shell Scripts**or**Shell Programs**. Shell scripts are similar to the batch file in MS-DOS. Each shell script is saved with**`.sh`** file extension e.g., **myscript.sh.**

A shell script has syntax just like any other programming language like Python, C/C++ etc.

A shell script comprises the following elements –

* Shell Keywords – if, else, break etc.
* Shell commands – cd, ls, echo, pwd, touch etc.
* Functions
* Control flow – if..then..else, case and shell loops etc.

### **Why do we need shell scripts?**

There are many reasons to write shell scripts:

* To avoid repetitive work and automation
* System admins use shell scripting for routine backups.
* System monitoring
* Adding new functionality to the shell etc.

### **Some Advantages of shell scripts**

* The command and syntax are exactly the same as those directly entered in the command line, so programmers do not need to switch to entirely different syntax
* Writing shell scripts are much quicker
* Quick start
* Interactive debugging etc.

### **Some Disadvantages of shell scripts**

* Prone to costly errors, a single mistake can change the command which might be harmful.
* Slow execution speed
* Design flaws within the language syntax or implementation
* Not well suited for large and complex task
* Provide minimal data structure unlike other scripting languages. etc.

# EXPERIMENT NO. 1

**Use of Basic Unix Shell Commands: ls, mkdir, rmdir, cd, cat, banner, touch, file, wc, sort, cut, grep, dd, dfspace, du, ulimit.**

1. **Ls : listing files**

**Syntax** - $ls

Foo letter2

Foobar letter3

**Example** - $ls l\* (display all files beginning with the string l)

**Output –** letter1 letter2 letter3

**The (\*) is “wildcard” character.**

**2. mkdir :** Make Directories(use this command to create a directory)

**Syntax –**mkdir[directory name]

**Example –**mkdir easy

**3.Rmdir :**remove directories.(use this command to remove directory)

**Syntax -** $rmdir [directory name]

**Example -** $rmdir easy

**Options:-**rm –r directory name will remove all files even if directory is not empty

rmdir –r will remove directory& any present directory that are empty

rmdir –s will suppress standard error message caused by – p

**4.Cd :**change directories

This command is used to change the current working/present directory.

**Syntax - $cd [dir name]**

**Example - $cd my**

**5.Cat :**create file, display content of file, concatenate files

You can create a file without a text editor by using the cat command and “>”

(Redirect output) symbol. To create a file using cat command, type

**Syntax -** $cat>file name

**Example -** $cat>colors

Red,green,blue……………………………<CTRL\_D>

To view a file we use cat in different way:

$cat colors

**Output –**Red,green,blue

**6. banner** command in linux is used to print the ASCII character string in large letter to standrad output.

**Syntax:**

banner text

**7.**The ***touch*** command is a standard command used in UNIX/Linux operating system which is used to create, change and modify timestamps of a file. Basically, there are two different commands to create a file in the Linux system which is as follows:

**touch command:** It is used to create a file without any content. The file created using touch command is empty. This command can be used when the user doesn’t have data to store at the time of file creation.

**8.Du:** used disk space.

The du stands for disk usage. this command is used to show the amount of disk space consumed by one or more files or directory (or directory trees).

**Syntax -** $du

**122 .**/backup

**425 .**/fa/backup

**2281** ./fa

**5** ./check

**18** ./dbf

**9.Ulimit :**user status

The Ulimit stand for user limit. This command shows the user space**.**

**Example -** $ulimit

**Output –** 2097151

**10. wc:**Print byte, word, and line counts, count the no. of bytes, whitespace-

separated words, and newlines in each given FILE

**Syntax -** wc [*options*]... [*File*]...

**Example -** $cat colors

**Output –** 2 line 4 words 27 character colors

**Options**

-c --bytes Print only the byte counts.

--chars

-w --words Print only the word counts.

-l --lines Print only the newline counts

**11.grep :**search for a string or word in a file

**Syntax** - $grep option filename

**Example -** $grep red colors

**Output –** red

**12. Dfspace:** free disk space int MB.

The Dfspace command present in/etc directory

**Example -**  $/etc/dfspace

**Output -** : Disk space: 6.30 MB available (4.59%)

Total Disk space: 6.30 MB of 135.74 MB available (4.59%)

**13. Sort :**sorting a file

**Syntax -** $sort [file name]

**Example -** $sort file1

**Output -** BcaMcaOo

**15. The cut** command in UNIX is a command for cutting out the sections from each line of files and writing the result to standard output. It can be used to cut parts of a line by **byte position, character and field**. Basically the cut command slices a line and extracts the text. It is necessary to specify option with command otherwise it gives error. If more than one file name is provided then data from each file is **not precedes** by its file name.

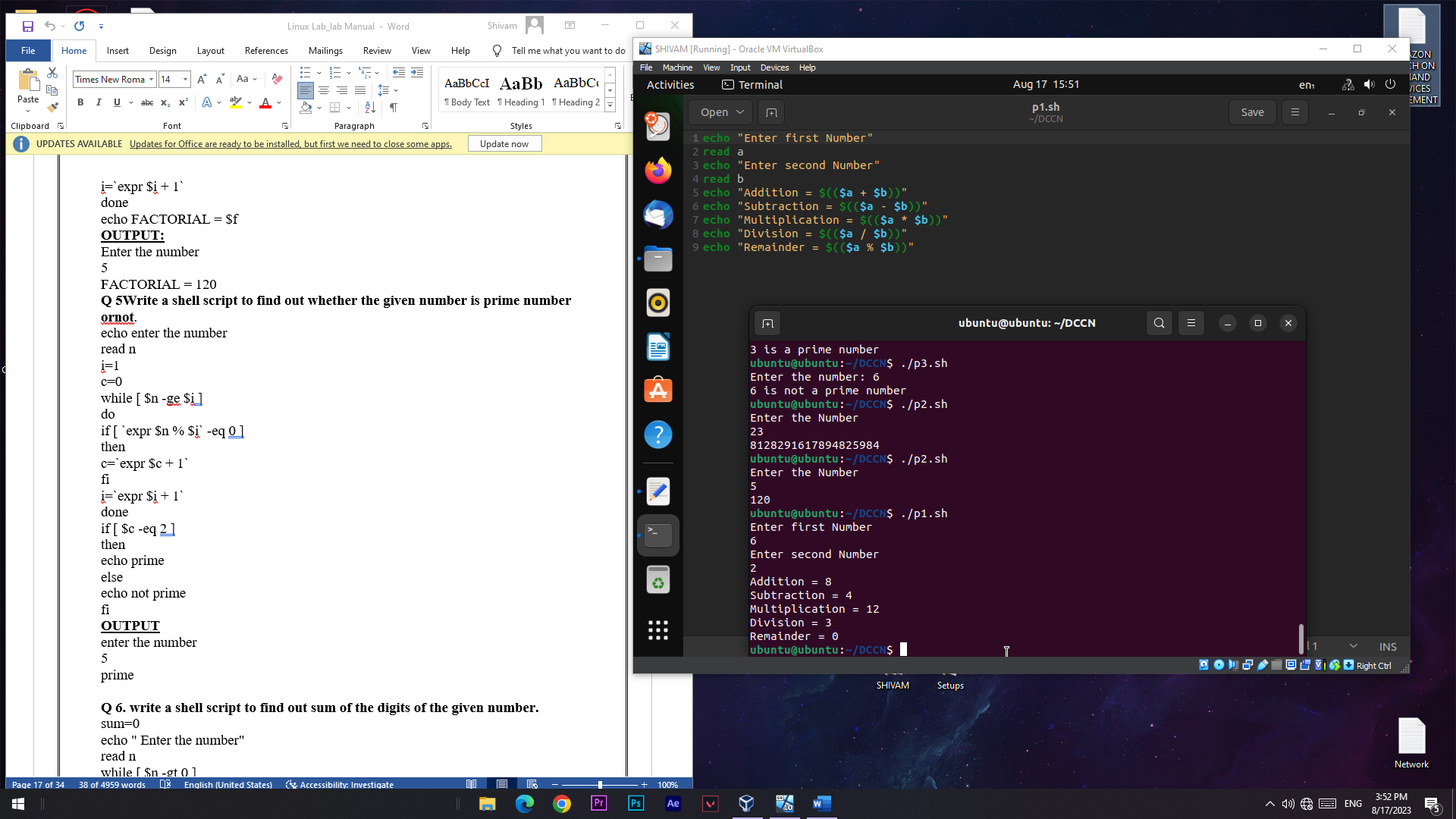
**Syntax:**

**cut OPTION... [FILE]...**The cut command in UNIX is a command for cutting out the sections from each line of files and writing the result to standard output. It can be used to cut parts of a line by **byte position, character and field**. Basically the cut command slices a line and extracts the text. It is necessary to specify option with command otherwise it gives error. If more than one file name is provided then data from each file is **not precedes** by its file name.

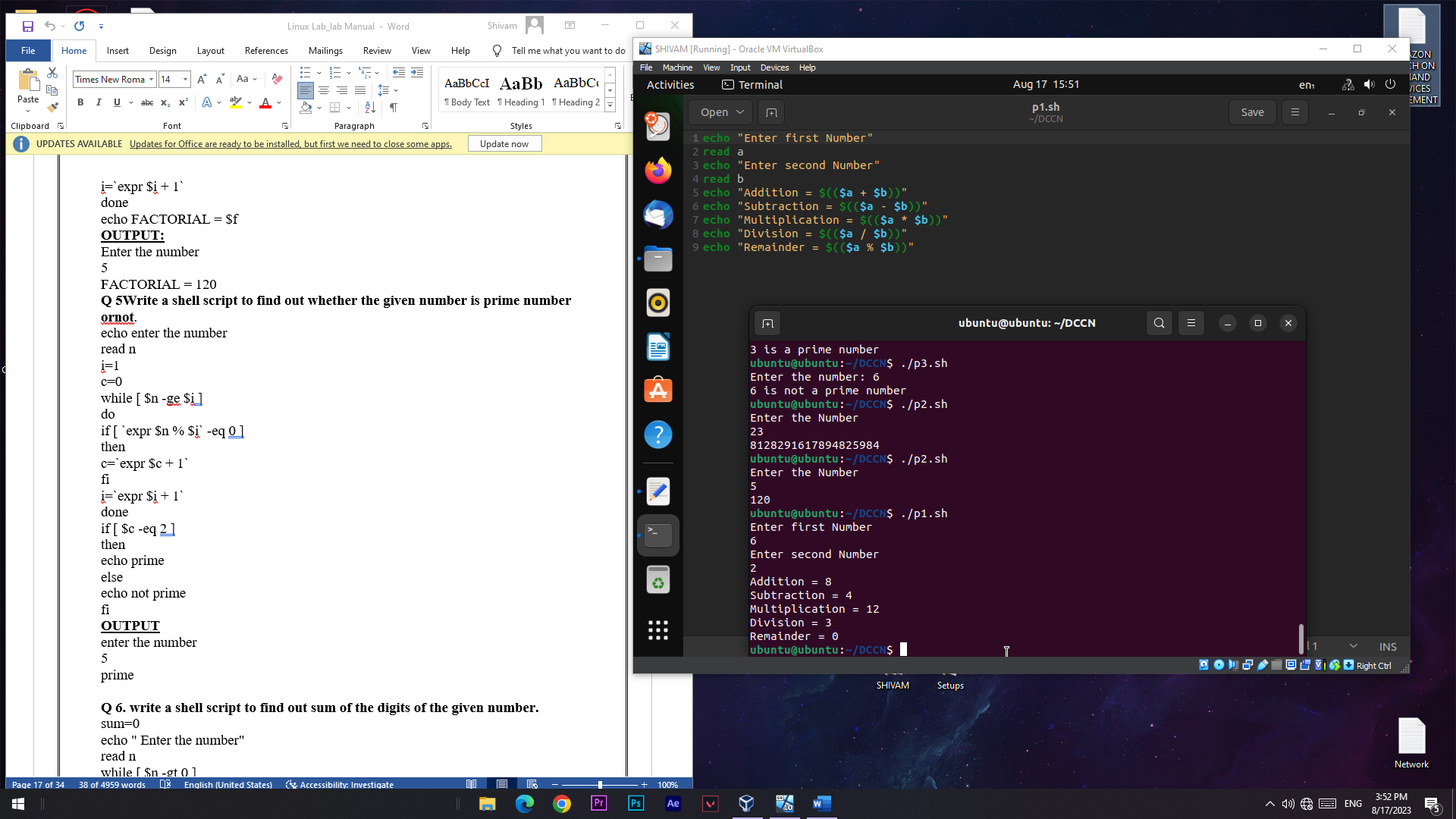
**Experiment No:-2**

**Aim:** Script to perform integer arithmetic operations

**Script:**



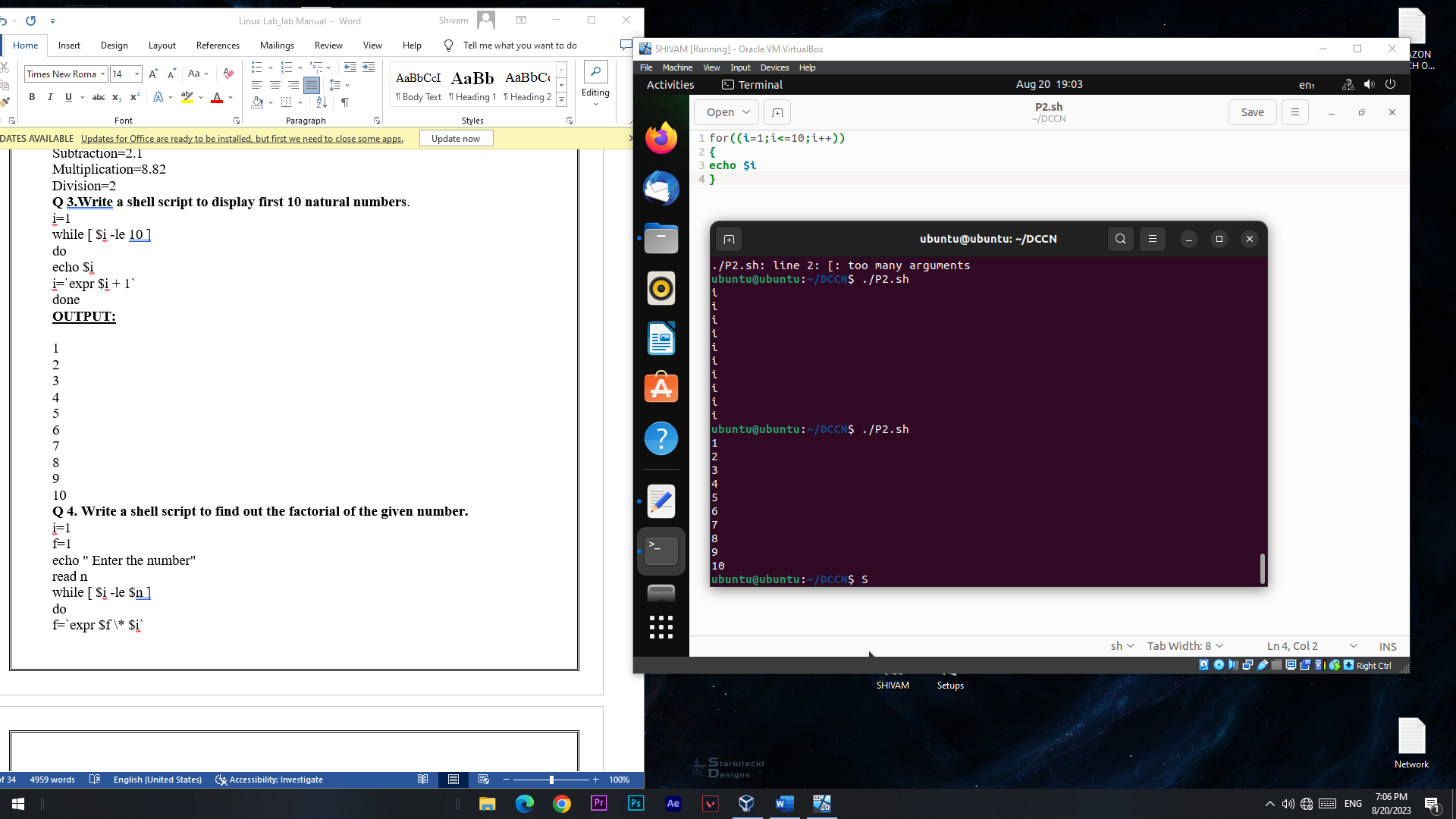
**Output:**

****

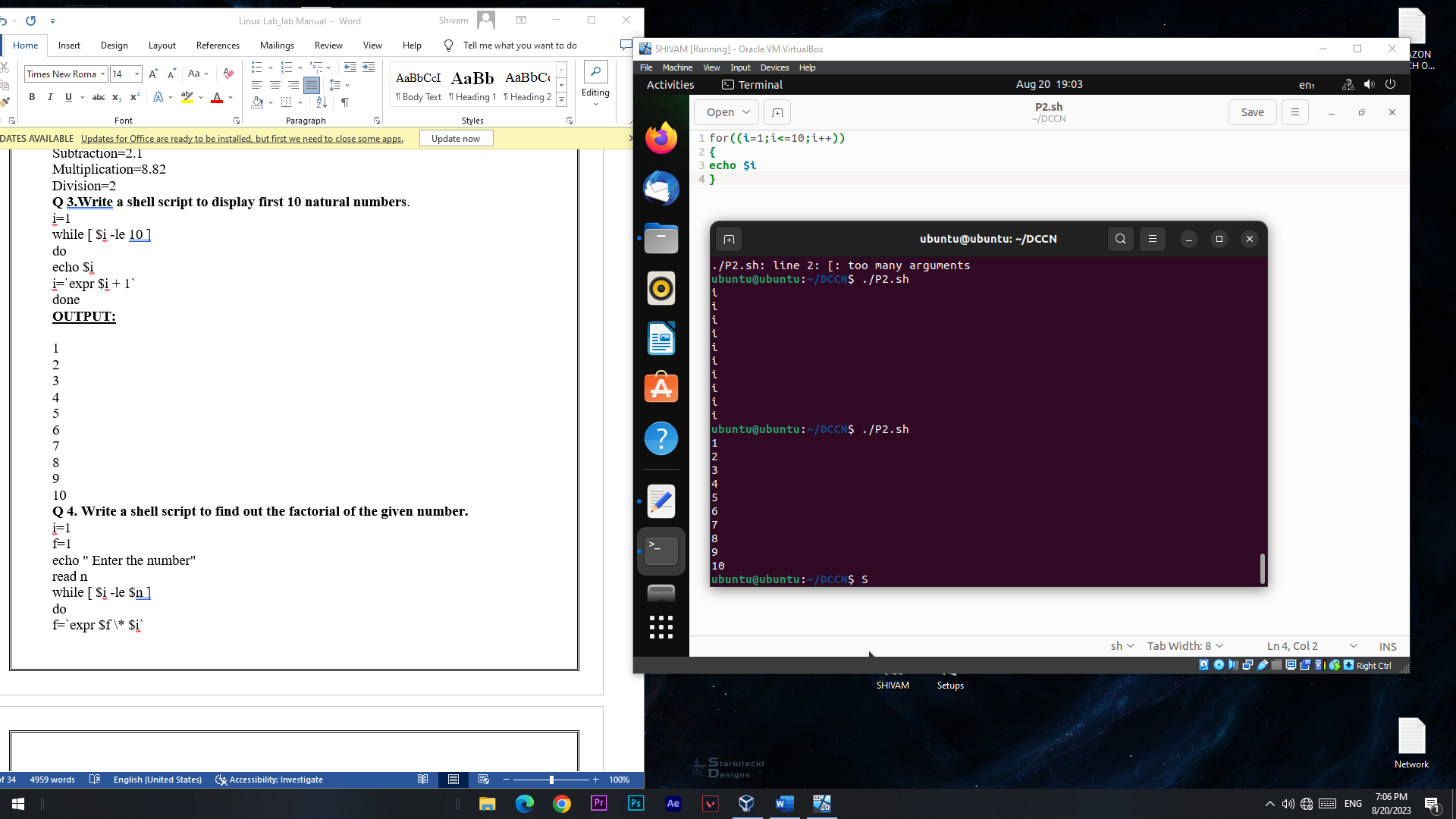
**Experiment No:-3**

**Aim:** Script to display first ten natural numbers

**Script:**

****

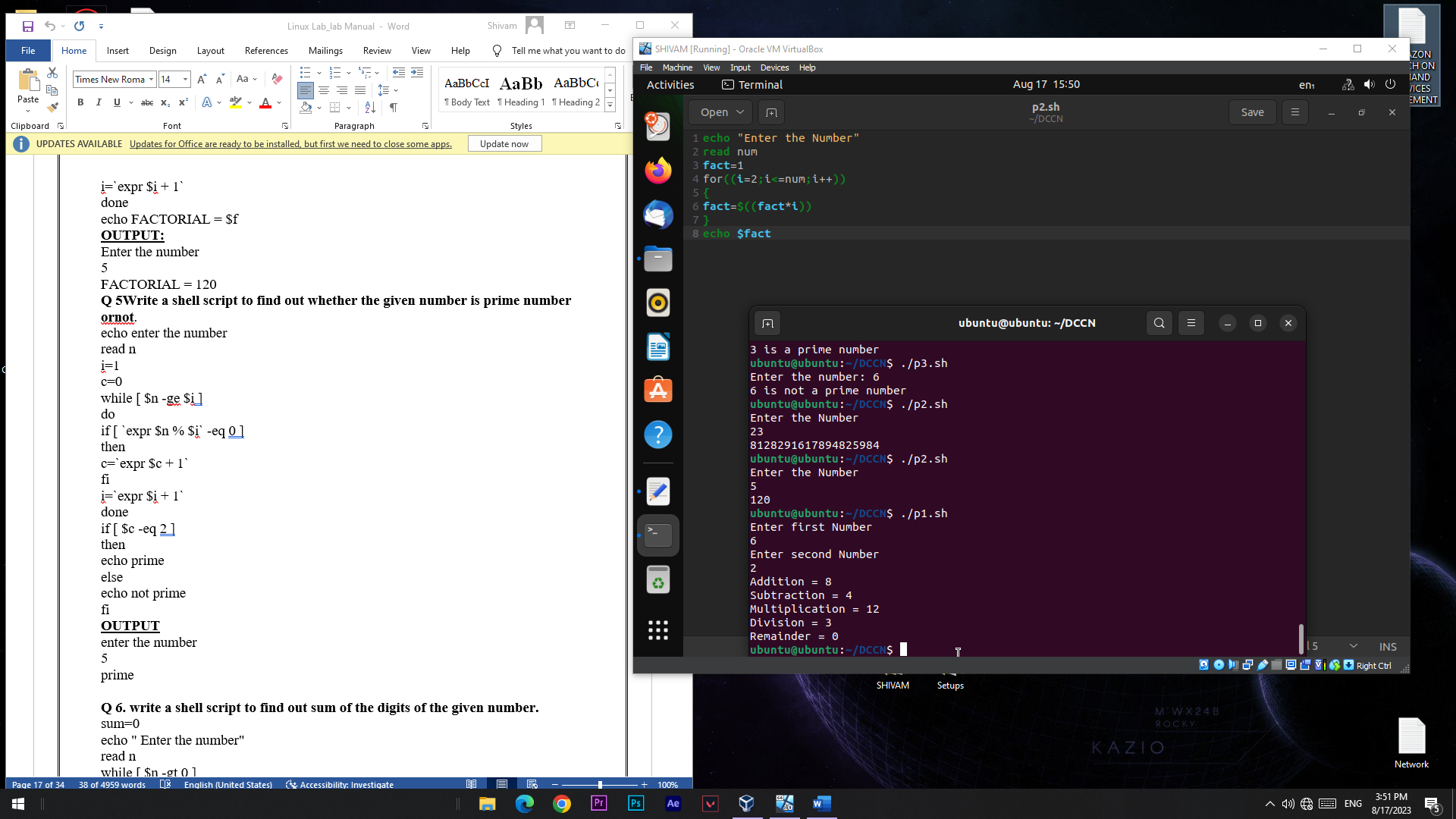
**Output:**

****

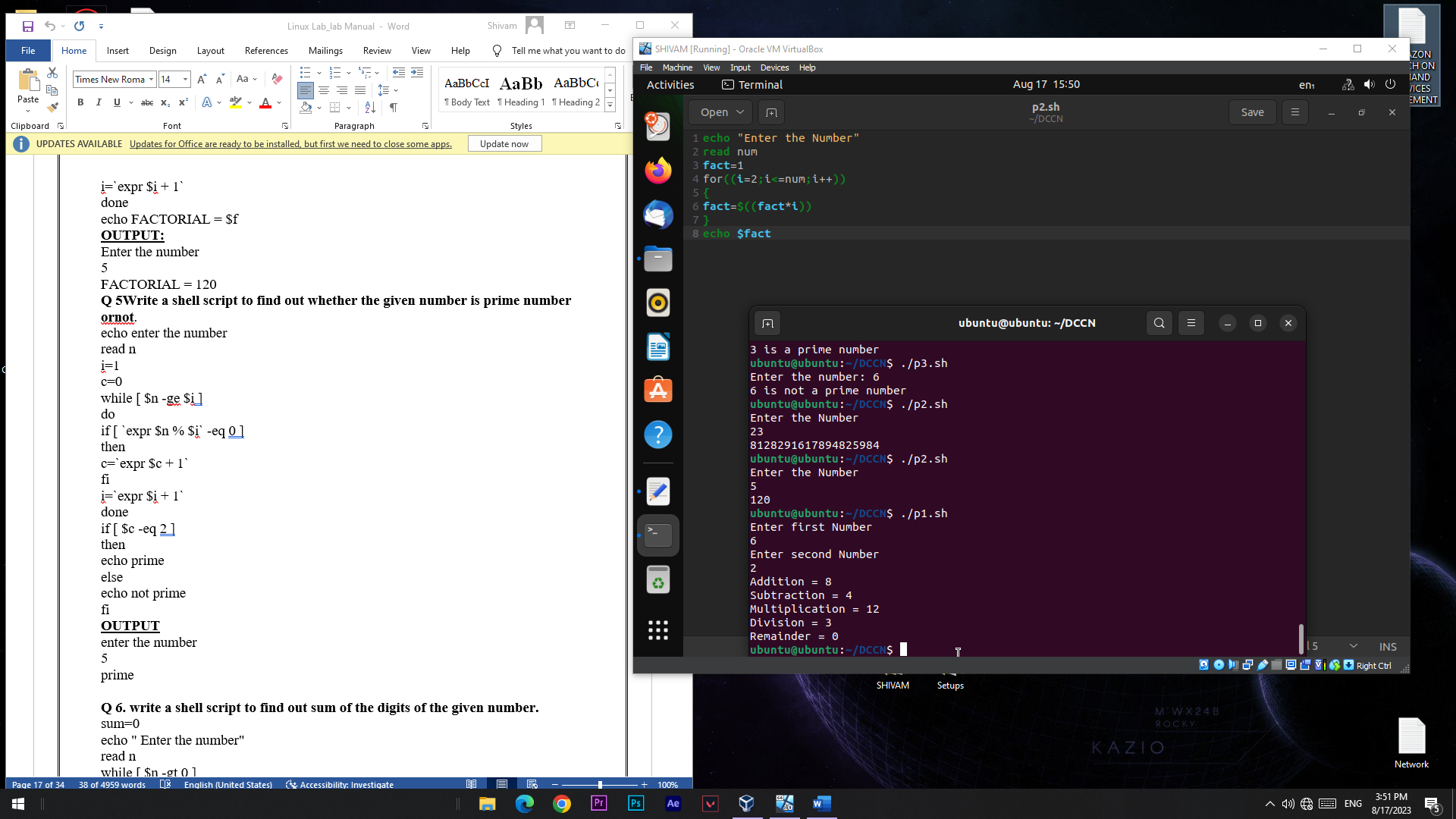
**Experiment No:-4**

**Aim:** Script to find factorial

**Script:**

****

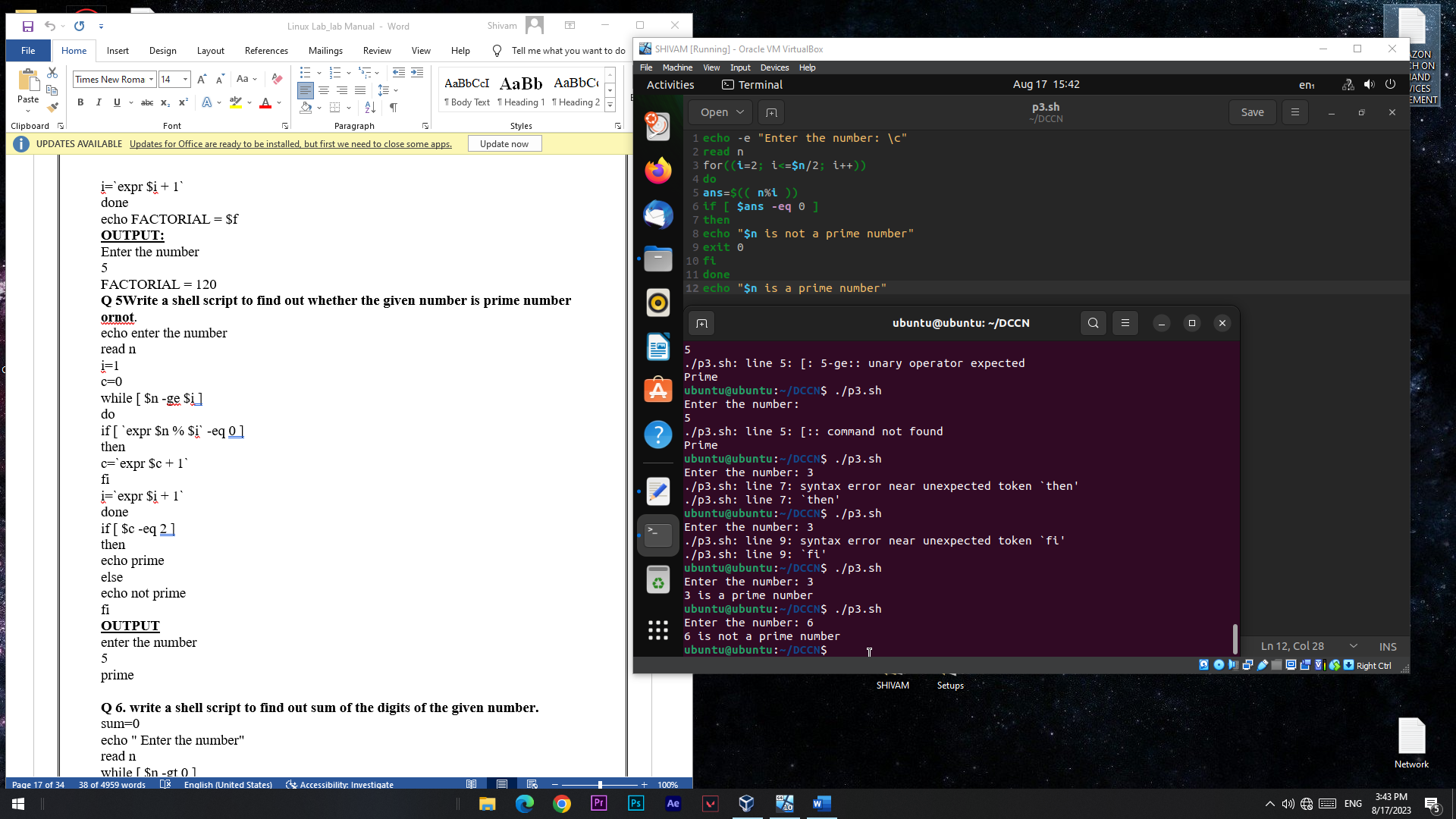
**Output:**

****

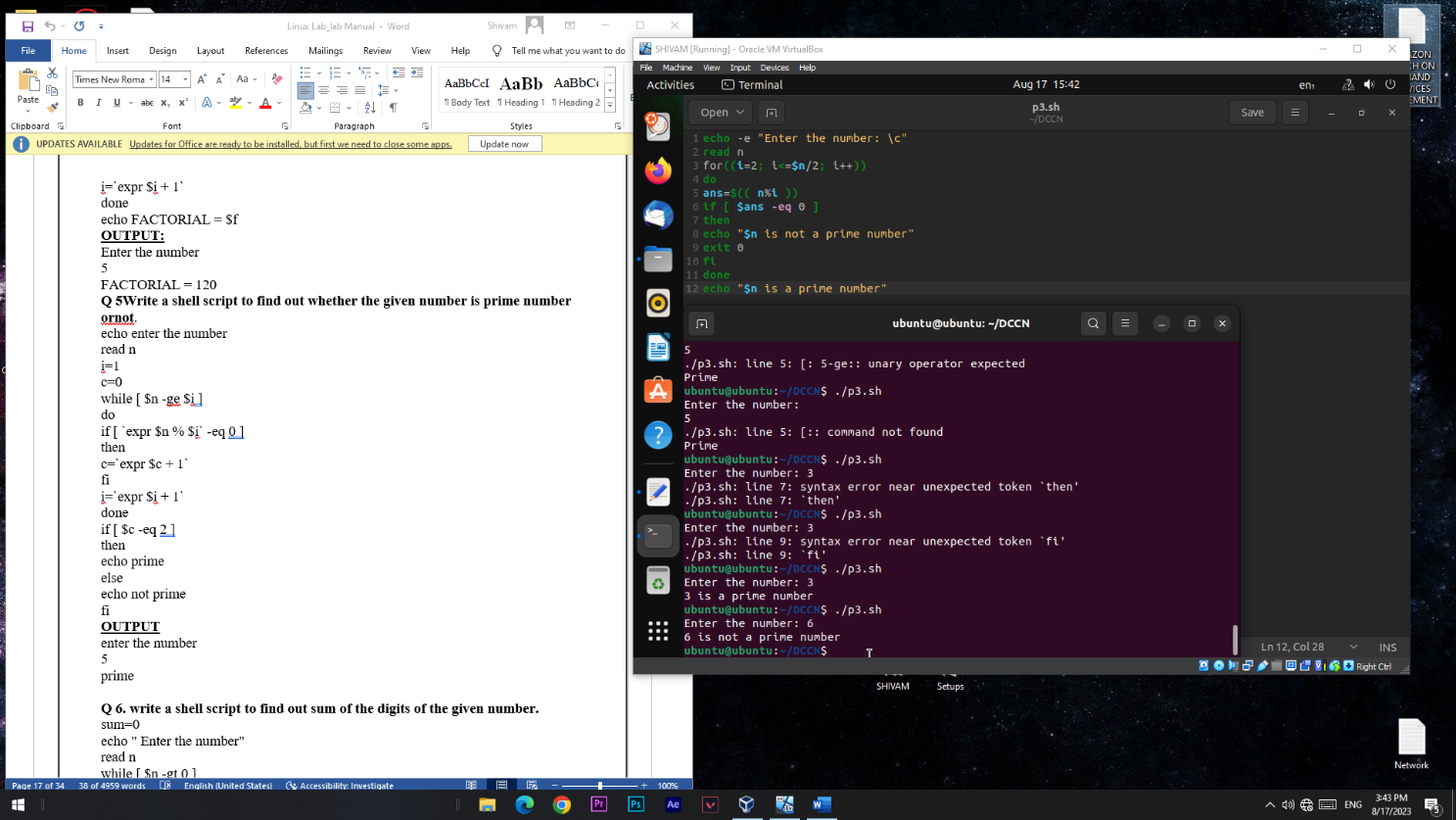
**Experiment No:-5**

**Aim:** Script to check whether number is prime or not

**Script:**

****

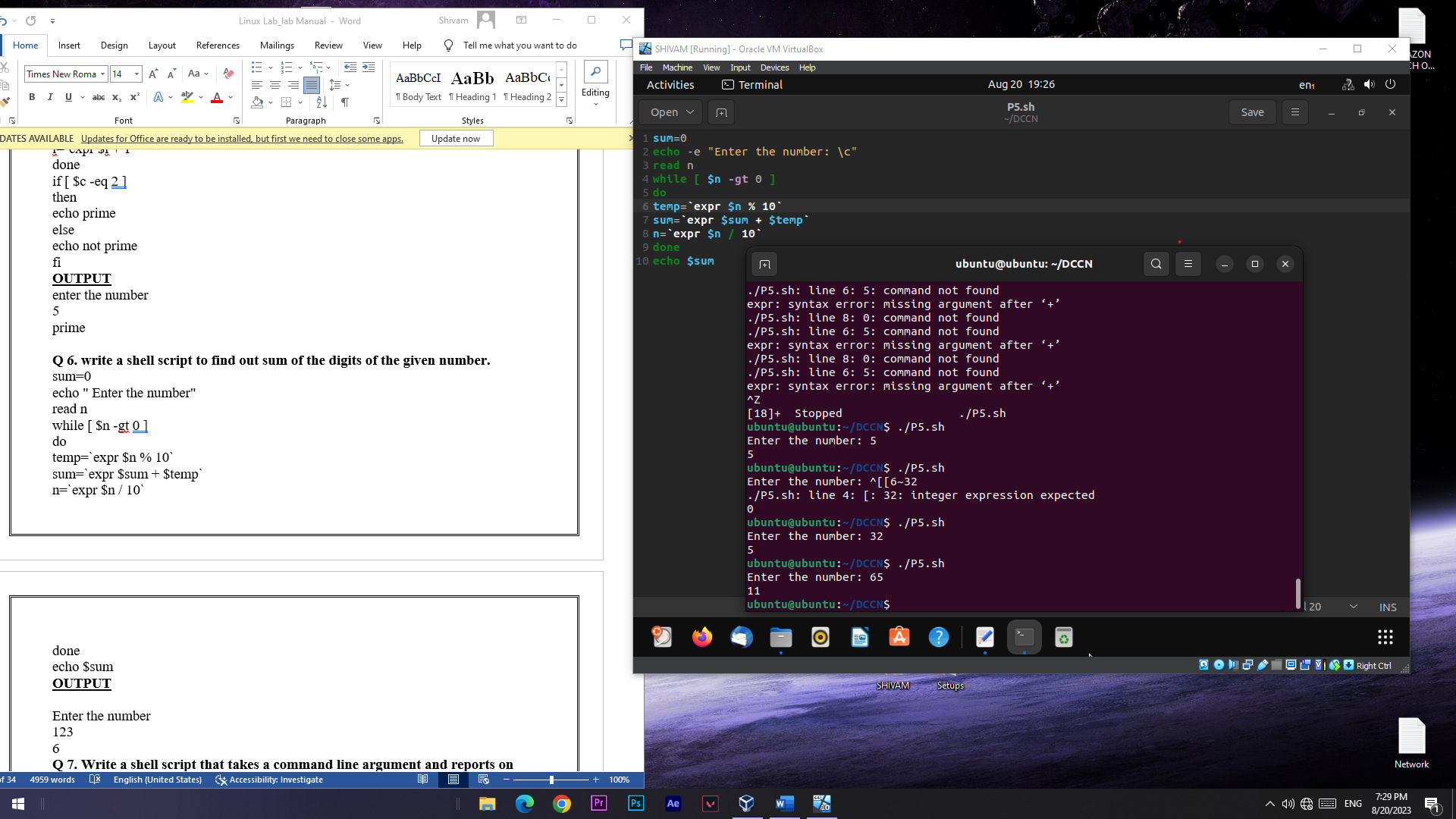
**Output:**

****

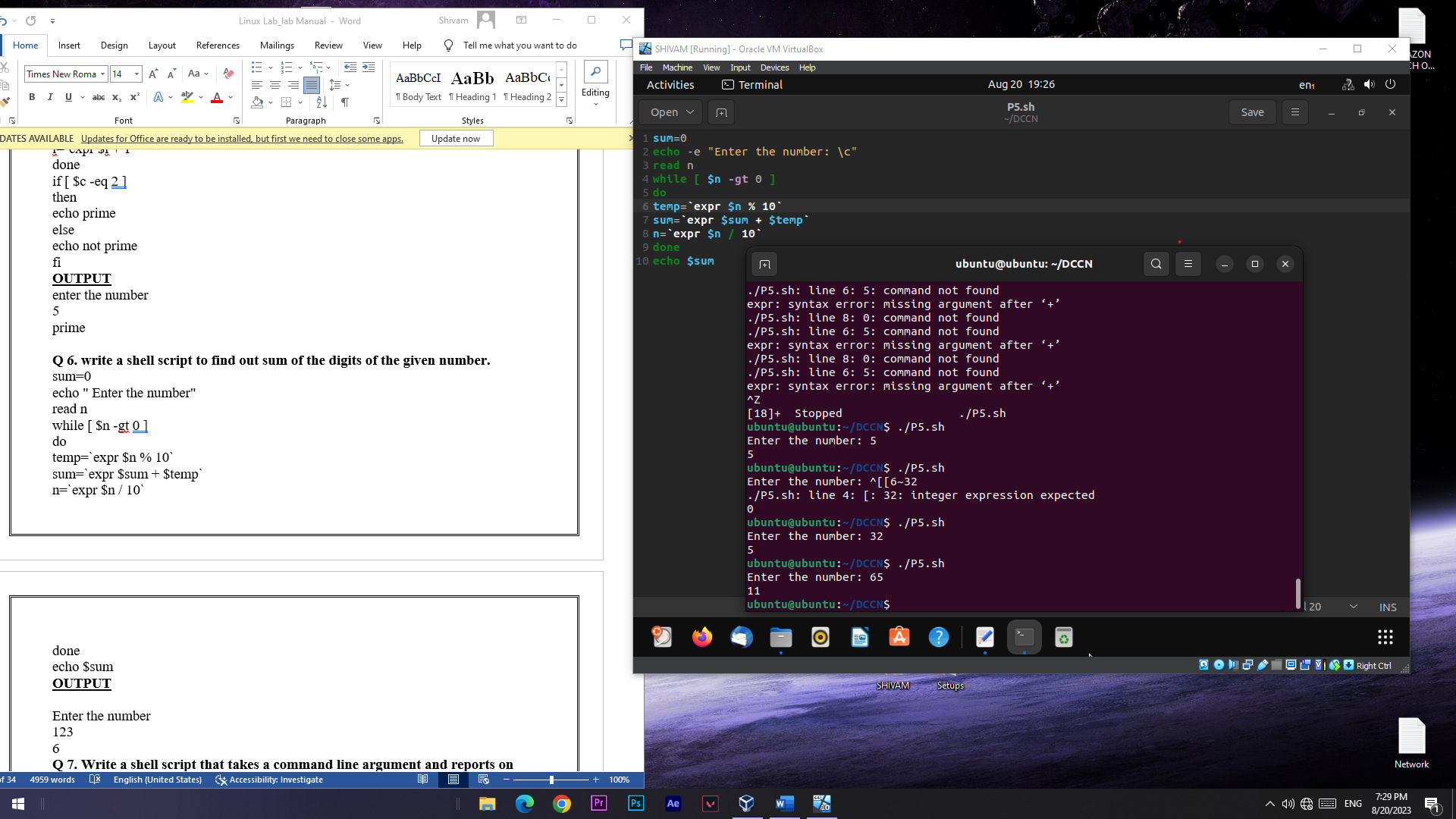
**Experiment No:-6**

**Aim:** Script to find sum of digits of given number

**Script:**

****

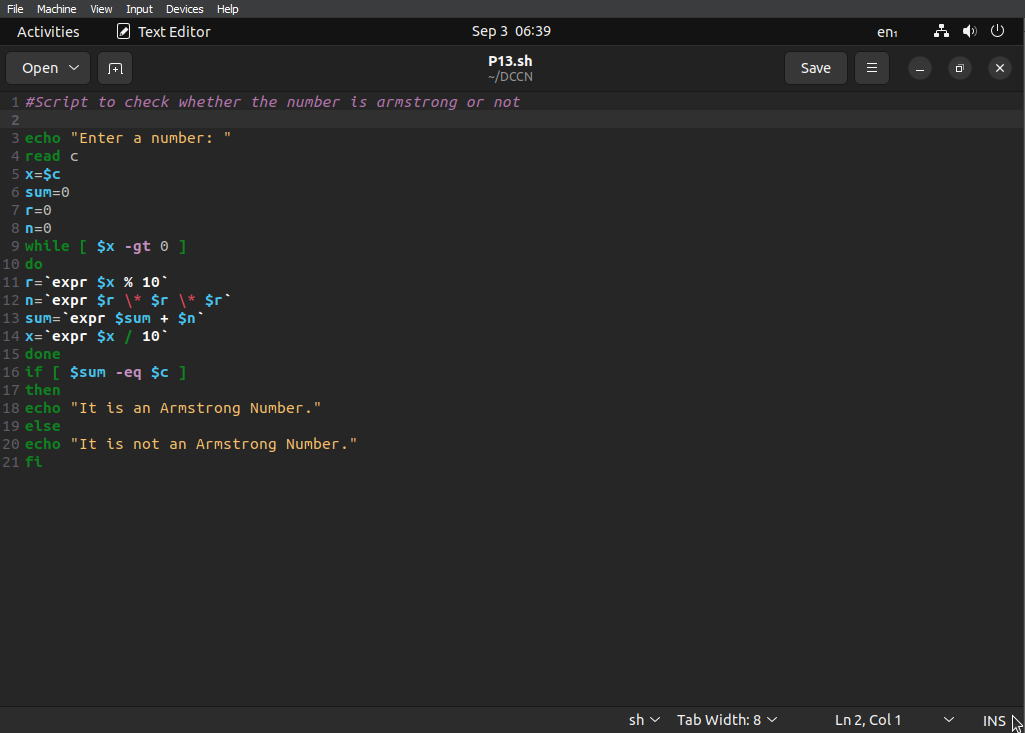
**Output:**

****

**Experiment No:-7**

**Aim:** Script to check Armstrong number

**Script:**

****

**Output:**

Enter a number:

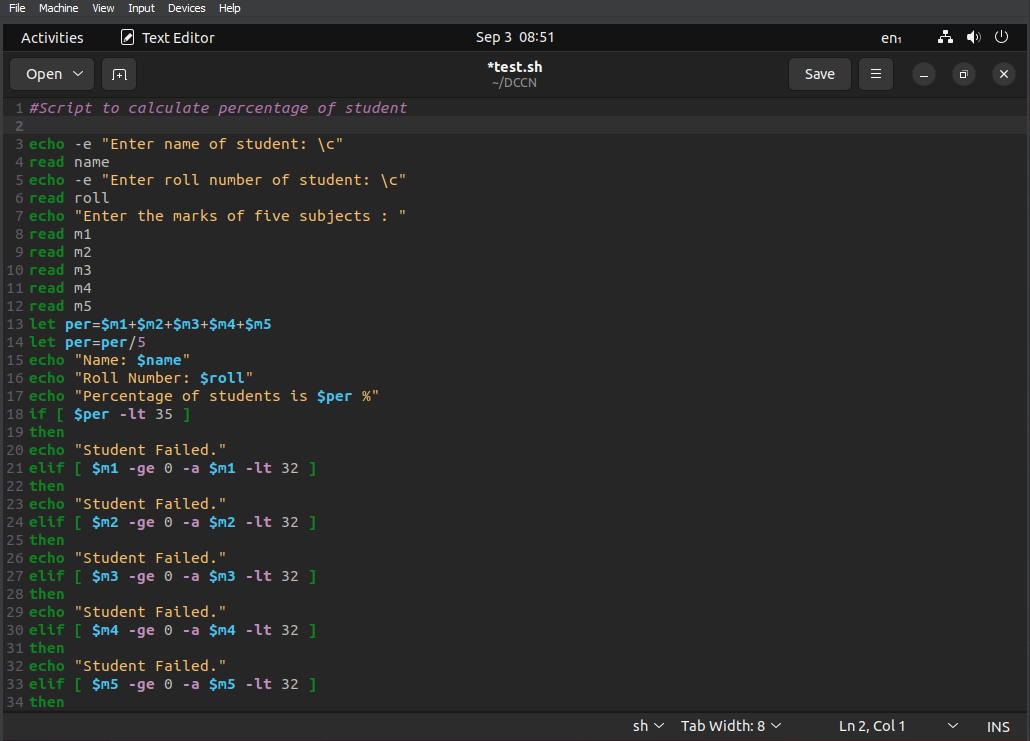
56

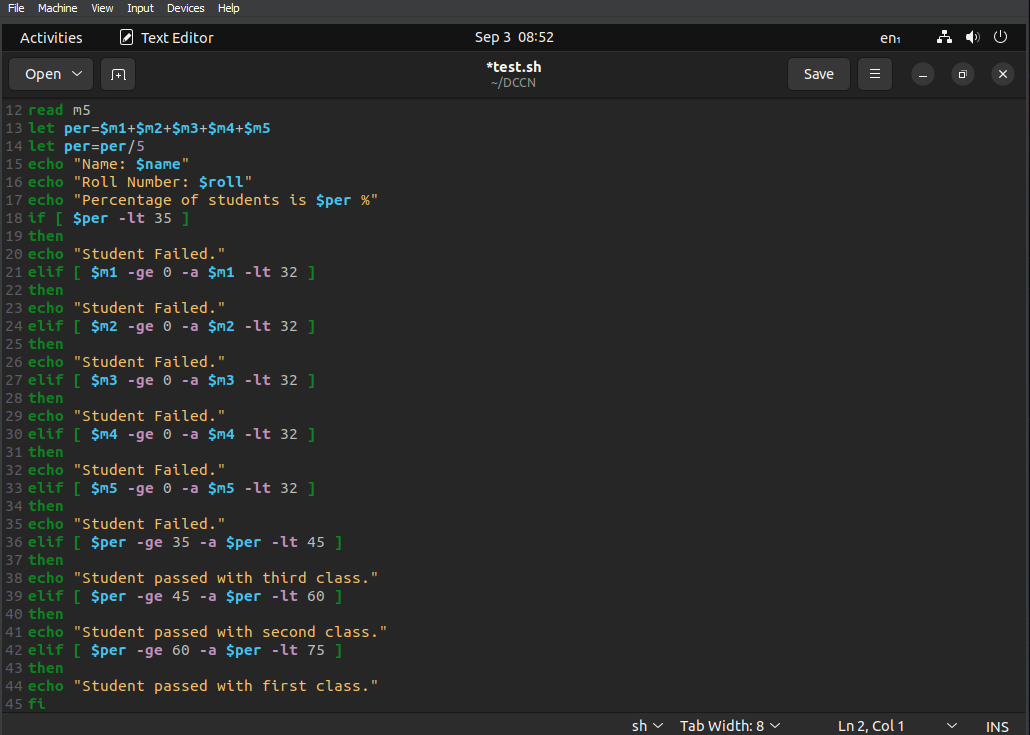
It is not an Armstrong Number.

**Experiment No:-8**

**Aim:** Script to Script to calculate percentage.

**Script:**

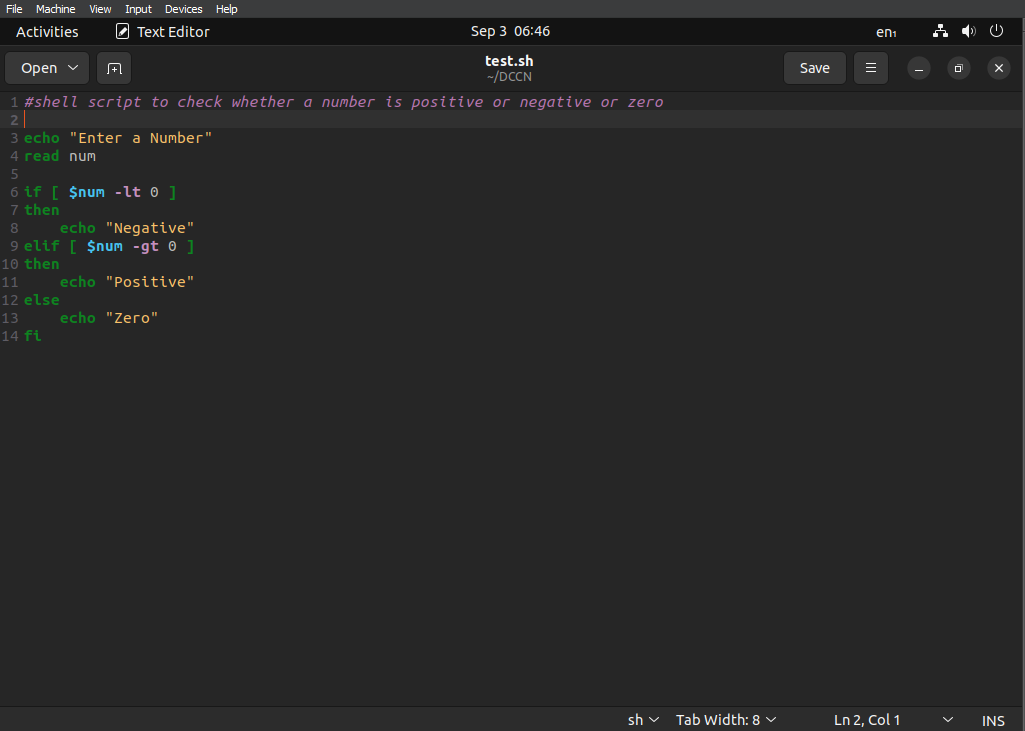
****

****

**Experiment No:-9**

**Aim:** Script to check whether number is positive, negative or zero

**Script:**

****

**Output:**

Enter a Number

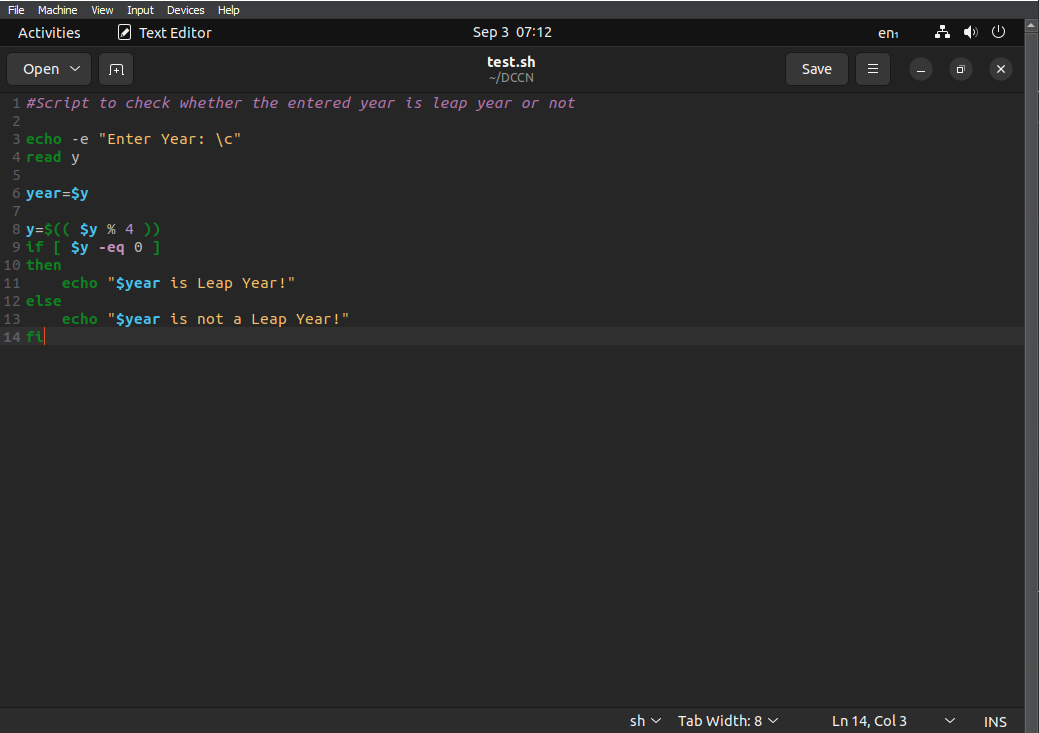
4

Positive

**Experiment No:-10**

**Aim:** Script to Script to check leap year

**Script:**



**Output:**

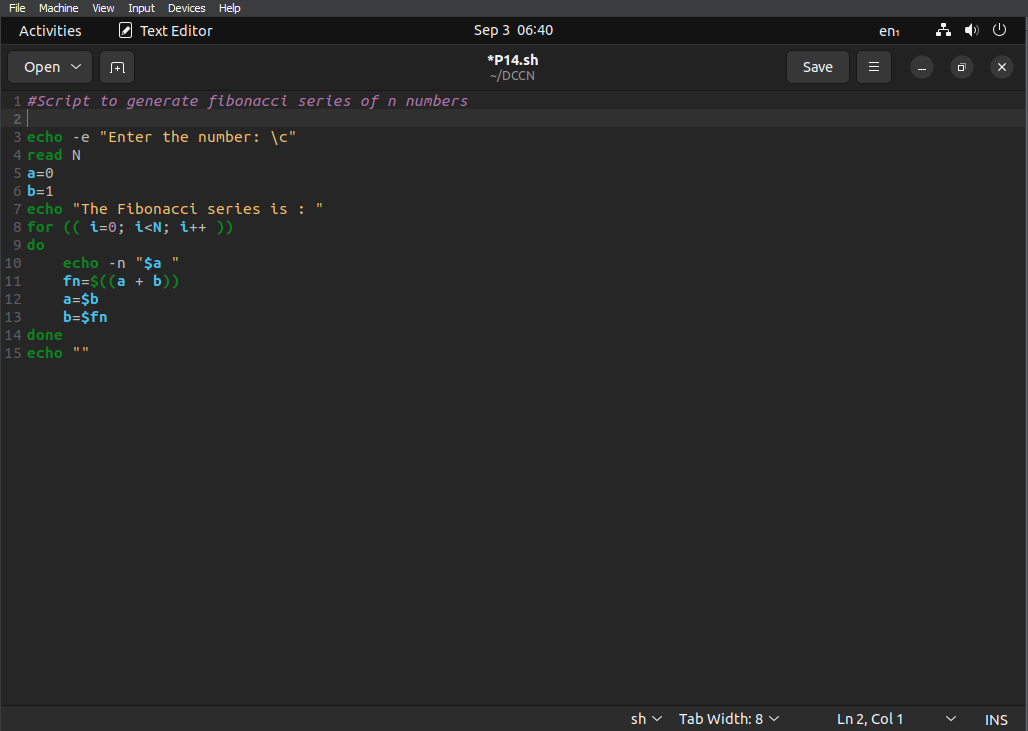
Enter Year: 2003

2003 is not a Leap Year

**Experiment No:-11**

**Aim:** Script to generate Fibonacci series

**Script:**



**Output:**

Enter the Number: 5

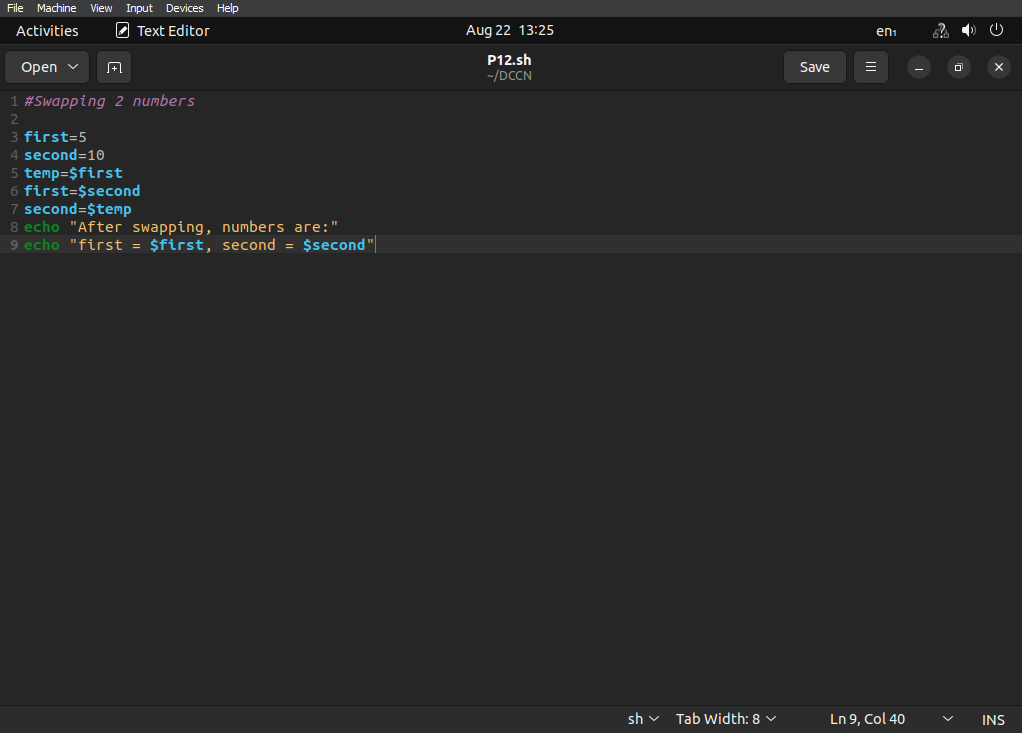
The Fibonacci series is:

01123

**Experiment No:-12**

**Aim:** Script to swap two numbers

**Script:**

****

**Output:**

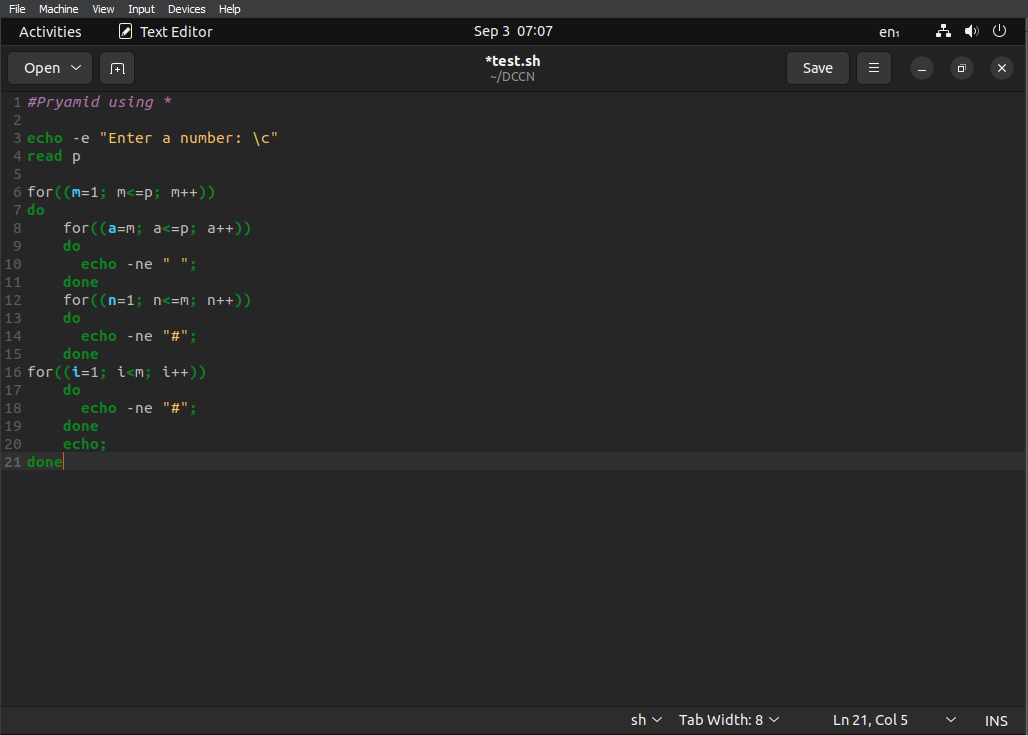
After swapping, numbers are:

first=10, second=5

**Experiment No:-13**

**Aim:** Script to generate pyramid using \*

**Script:**



**Output:**

Enter a number:5

\*

\*\*

\*\*\*

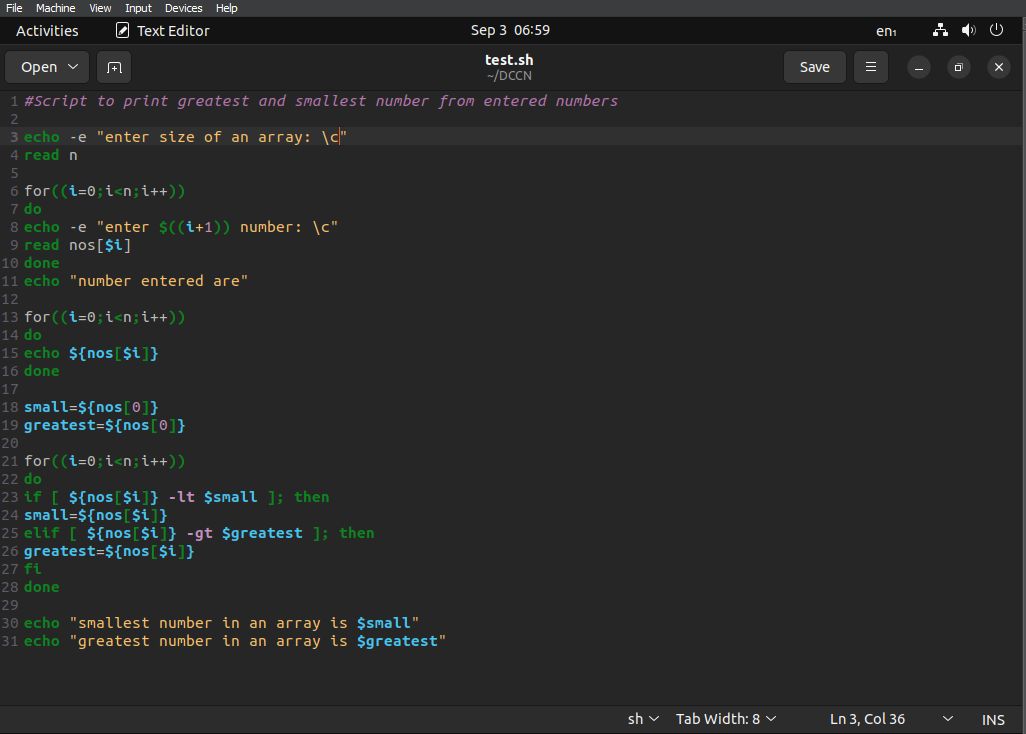
\*\*\*\*

\*\*\*\*\*

**Experiment No:-14**

**Aim:** Script to print greatest and smallest number

**Script:**

****

**Output:**

enter size of an array: 3

enter 1 number: 5

enter 2 number: 4

enter 3 number: 7

number entered are 5 4 7

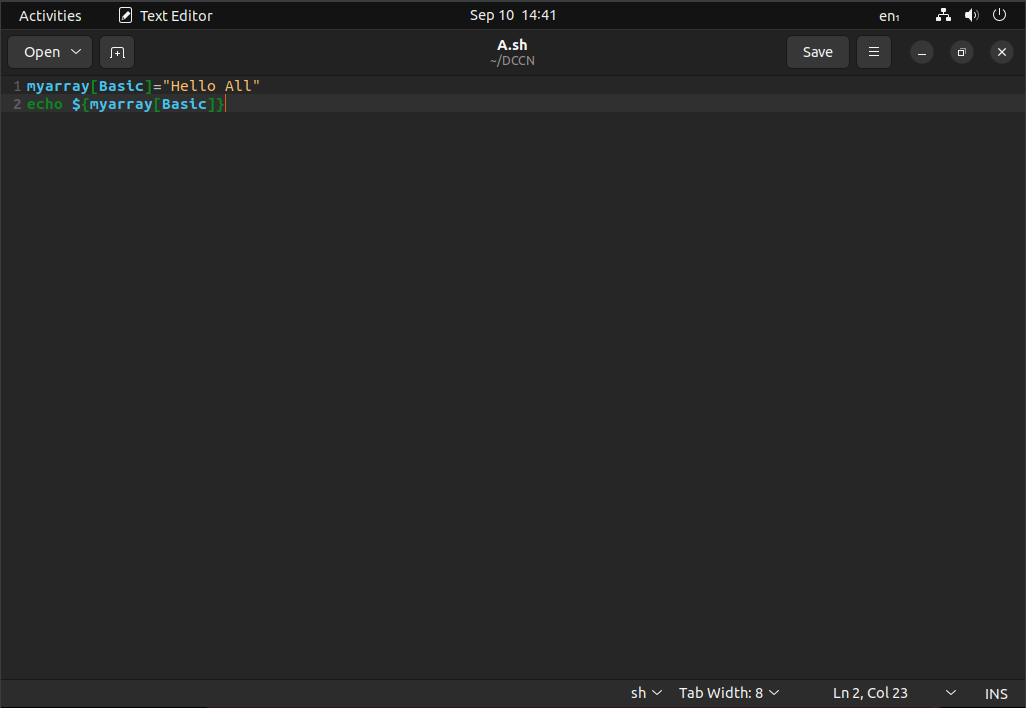
smallest number in an array is 4

greatest number in an array is 7

**Experiment No:-15**

**Aim:** Introduction to arrays

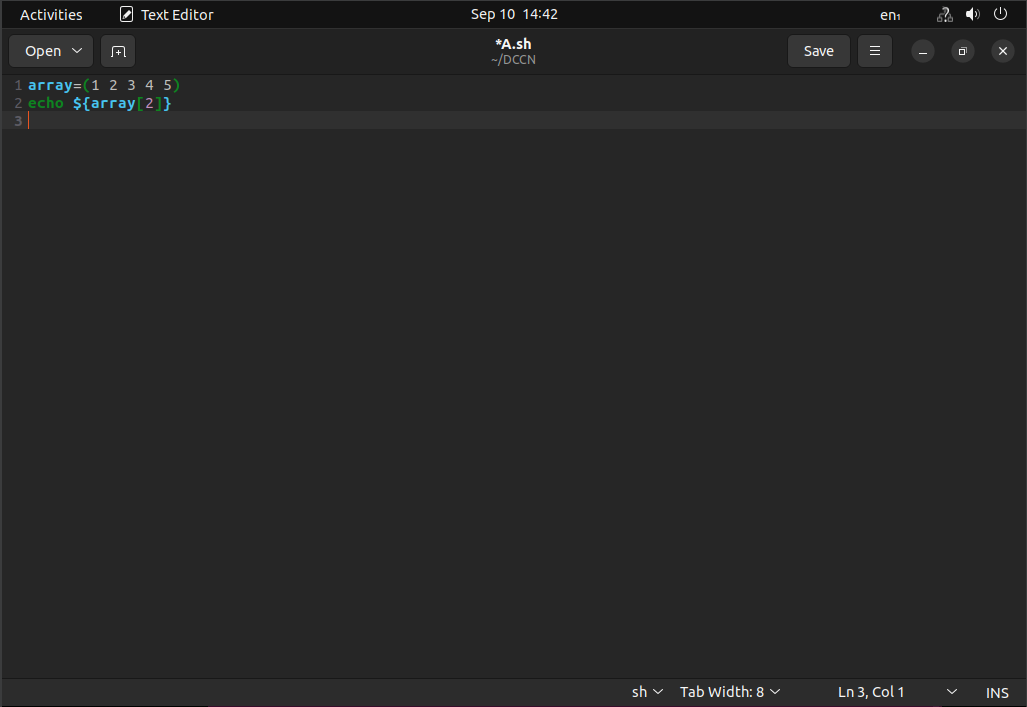
1. **//Accessing array**



**Output:**

Hello All

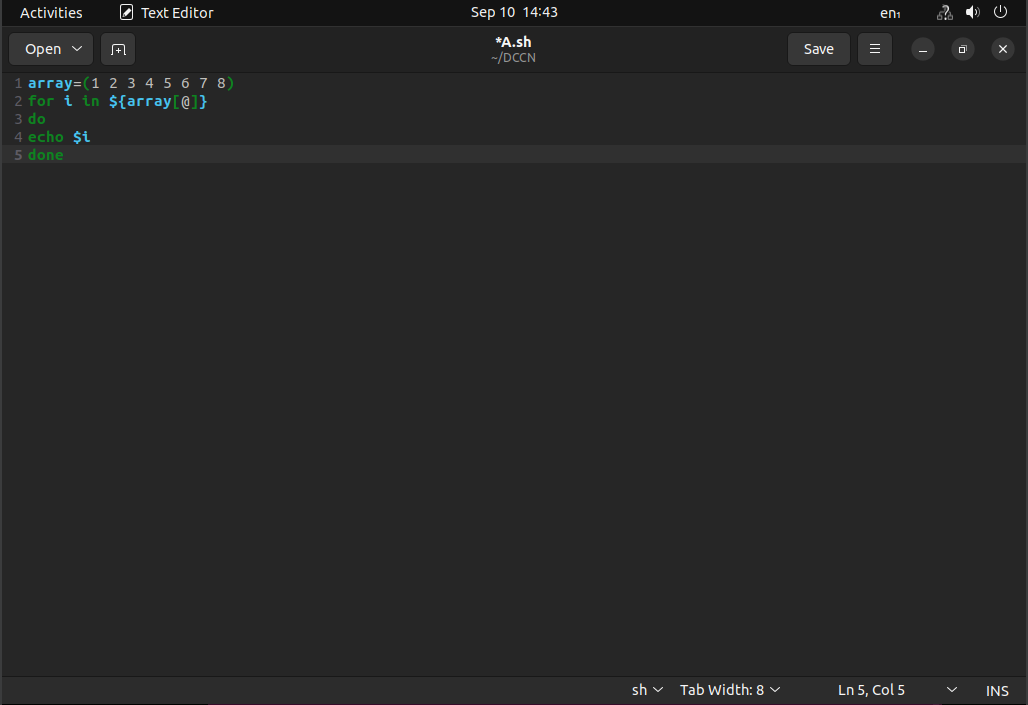
1. **//Indexing array**



**Output :**

3

1. **//Reading array elements**



**Output:**

1

2

3

4

5

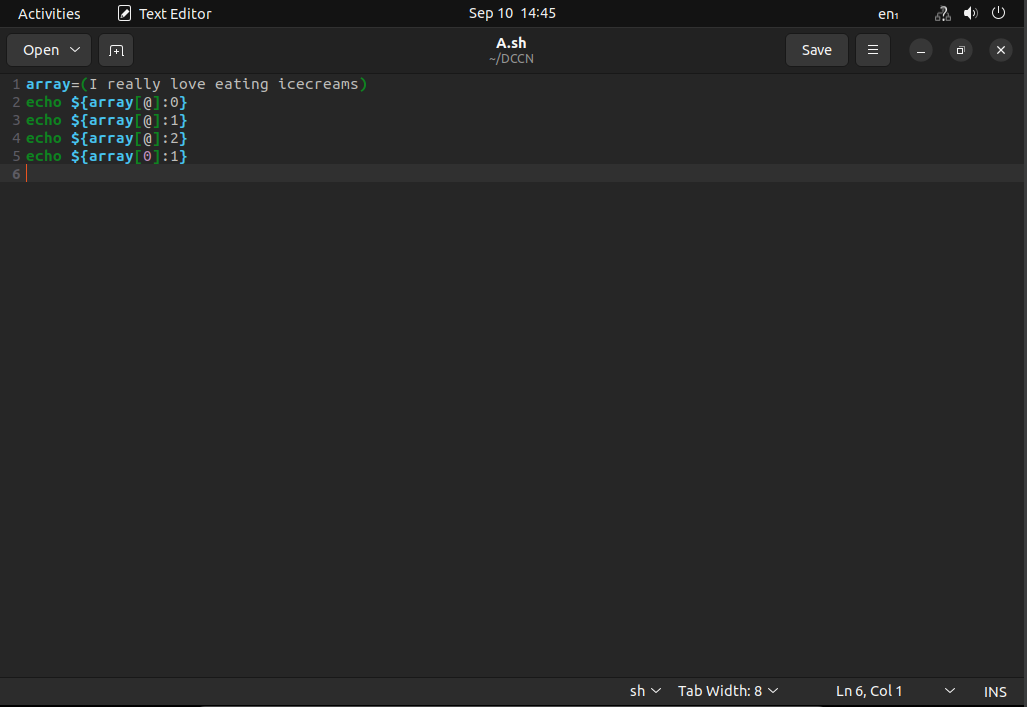
6

7

8

1. **//Use of [@] symbol**

[@] symbol is used to print all the elements at the same time. We can perform the same using [\*] symbol.



**Output:**

I really love eating icecream

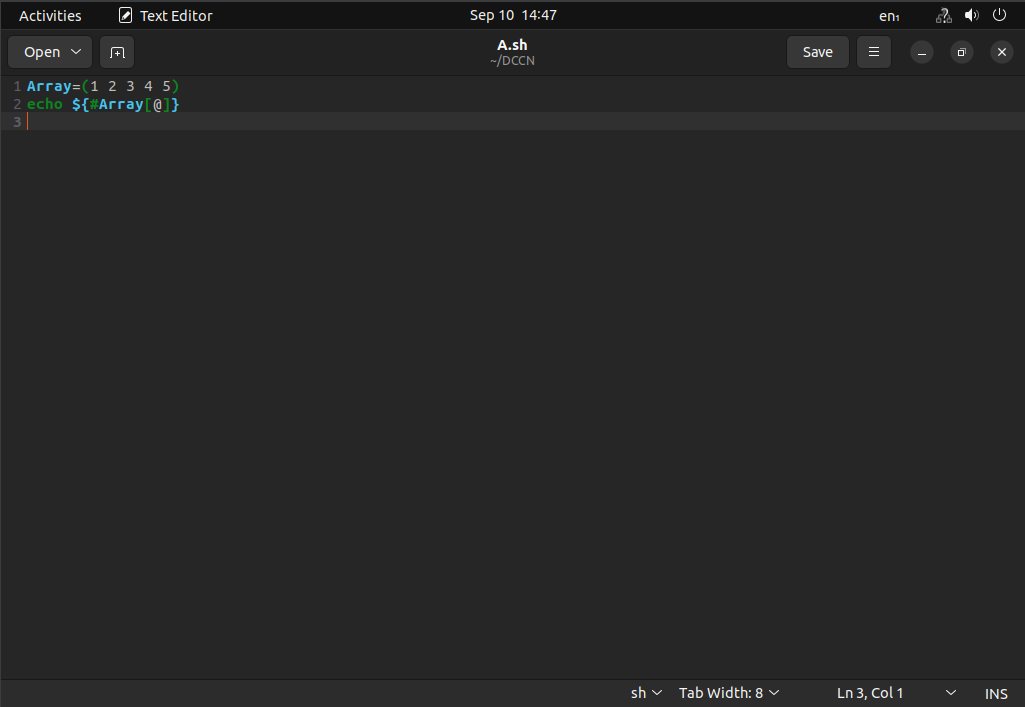
really love eating icecream

love eating icecream

I really love eating icecream

1. **//Use of [#] symbol**

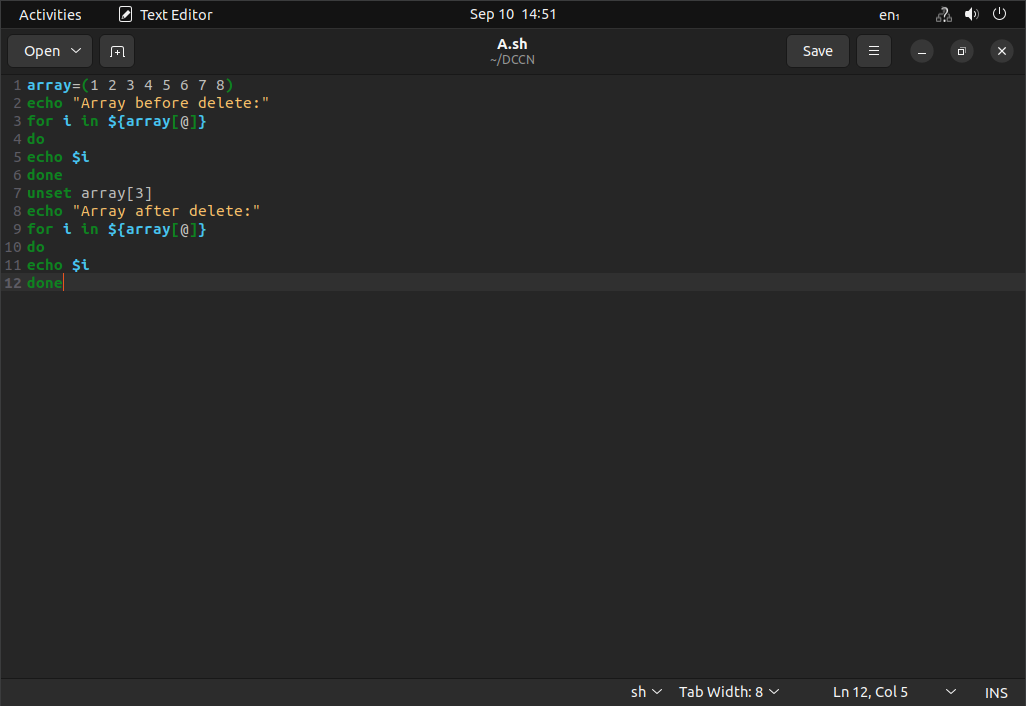
[#] symbol is used to get the count of the elements in the given array



**Output:**

5

1. **//Use of unset**



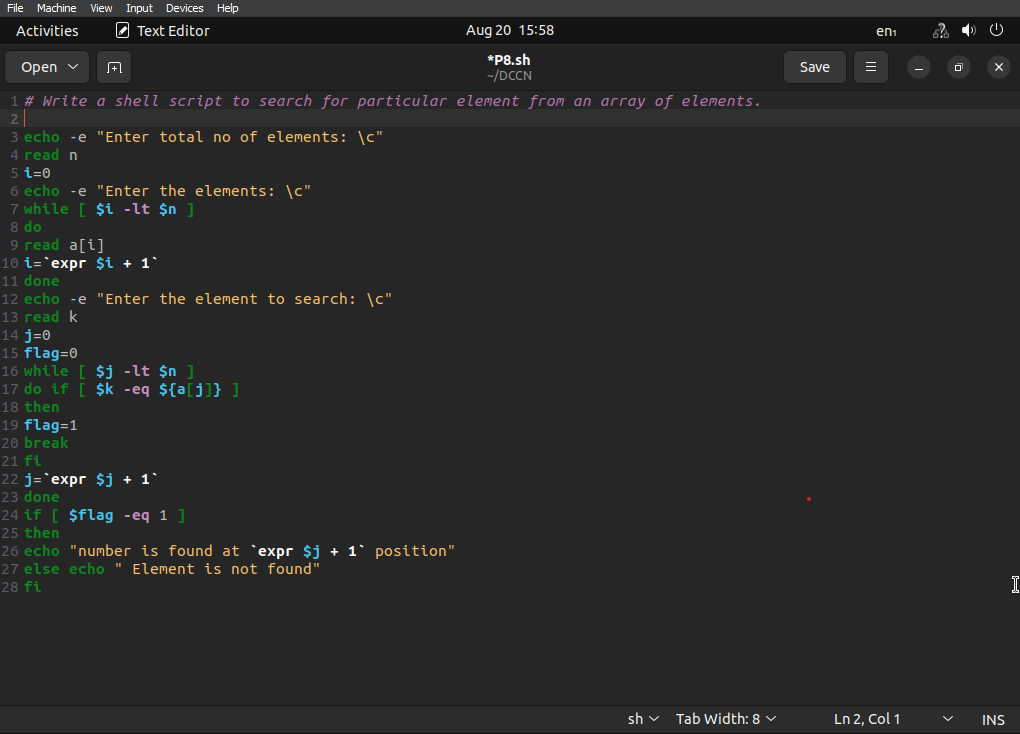
**Output:**

Array before delete: 1 2 3 4 5 6 7 8

Array after delete: 1 2 3 5 6 7 8

**Experiment No:-16**

**Aim:** Script to search a particular element in array

**Script:**

**Output:**

Enter total no of elements: 3

Enter the elements: 1 2 3

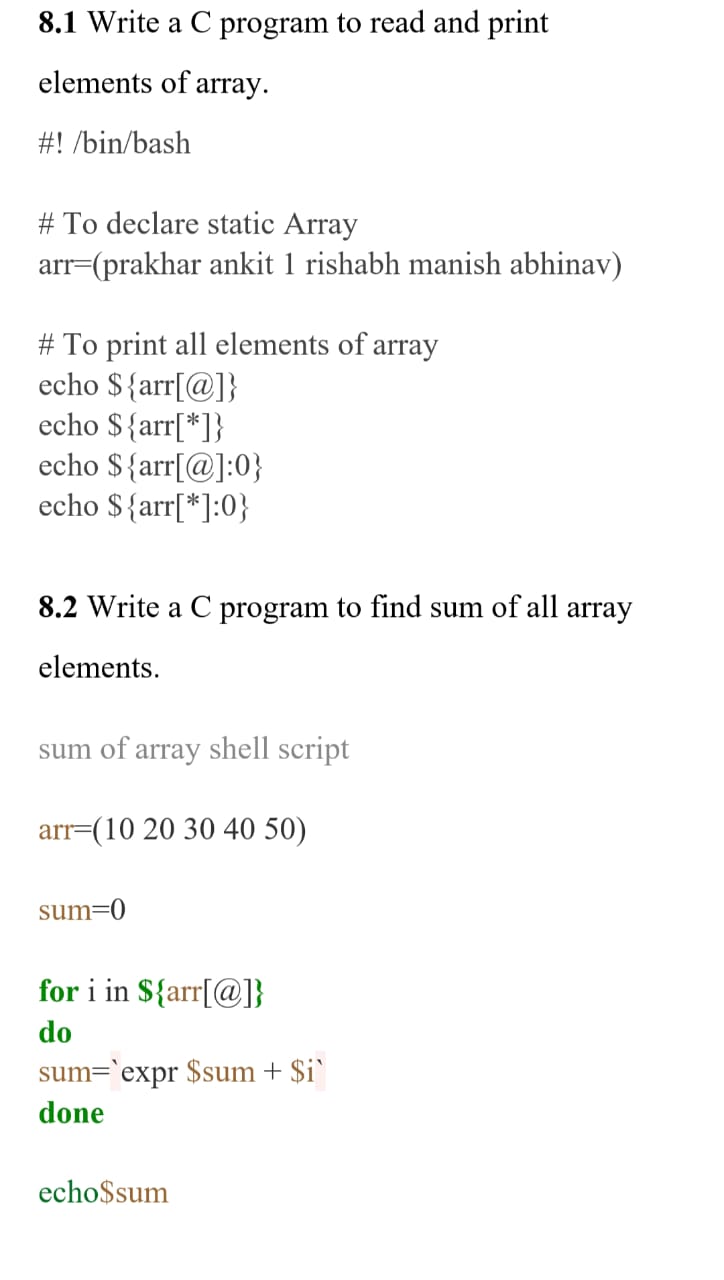
Enter elements to search: 2

Number is found at 2 position

**Experiment No:-17**

**Aim:** Script to read & print elements of array

**Script:**

****

**Output:**

Prakhar ankit 1 rishabh manish abhinav

Prakhar ankit 1 rishabh manish abhinav

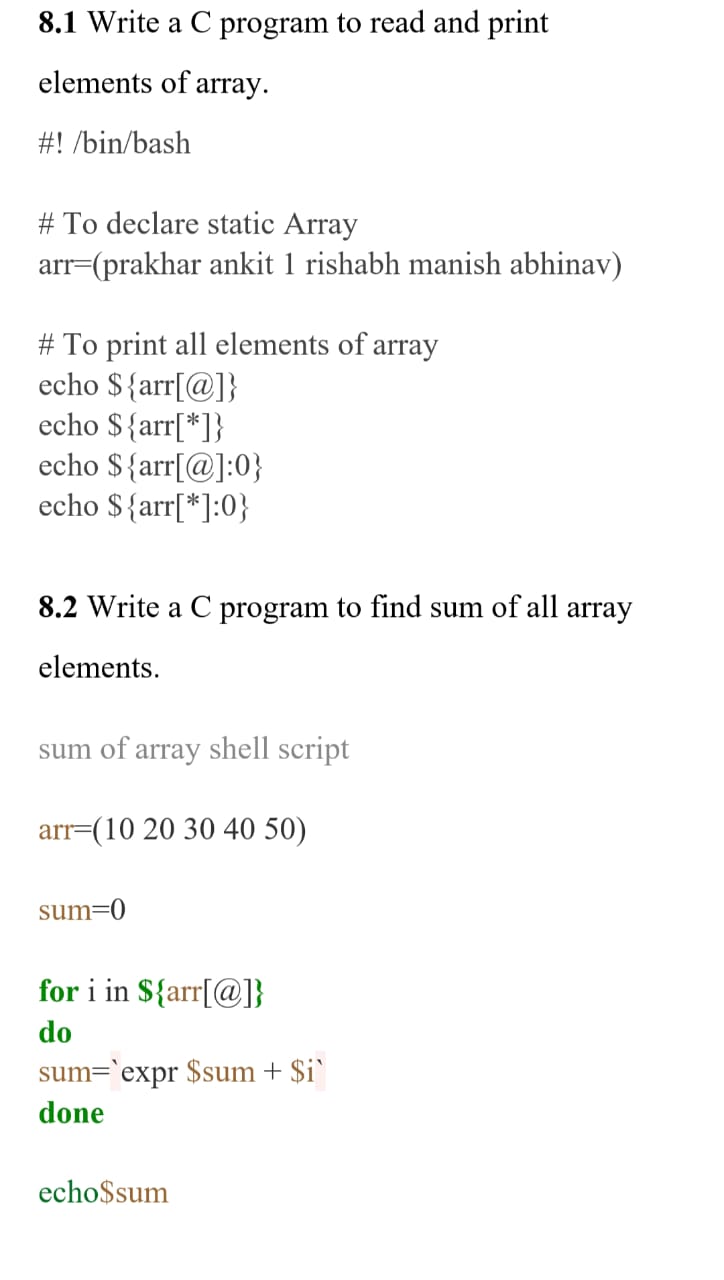
Prakhar ankit 1 rishabh manish abhinav

Prakhar ankit 1 rishabh manish abhinav

**Experiment No:-18**

**Aim:** Script to find sum of all array elements

**Script:**

****

**Output:**

150