Problem solving
And
Programming
With
C
Lab

Assignment by –

Roll NO: 20VV1A1263,

1-1 B. TECH – IT.

EXPERIMENT NO: 1

INTRODUCTION

TO

ALGORITHMS

AND

FLOWCHARTS.

1.1)

AIM: Implement Algorithm Development for Exchange the values of two numbers.

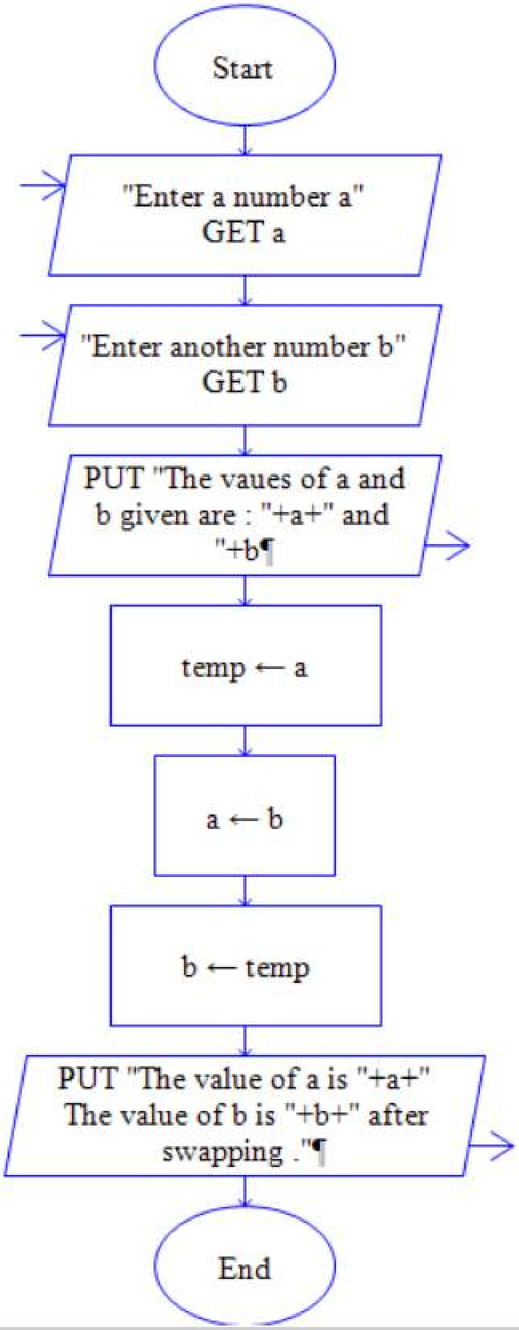
DESCRIPTION:

- The algorithm developed for this problem takes input of two values into two variables declared, namely a and b from the user.
- A temporary variable temp is declared which stores the value of a pre-given by user, later a is updates with value of b and similarly b with temp's value.
- This changes the value of a to b and b to a.
- Temp variable is created to avoid losing of a's value when it is updated.
- The program now prints these two values.

ALGORITHM:

```
Step1:   START.
Step2:   take input of variable a.
Step3:  take input of variable b.
Step4:   print the values before swapping.
Step5:   assign variable temp value of a.
Step6:  similarly assign a value of b.
Step7:  assign b with value of temp.
Step8:   print the values after swapping.
Step9:   END.
```

FLOWCHART:

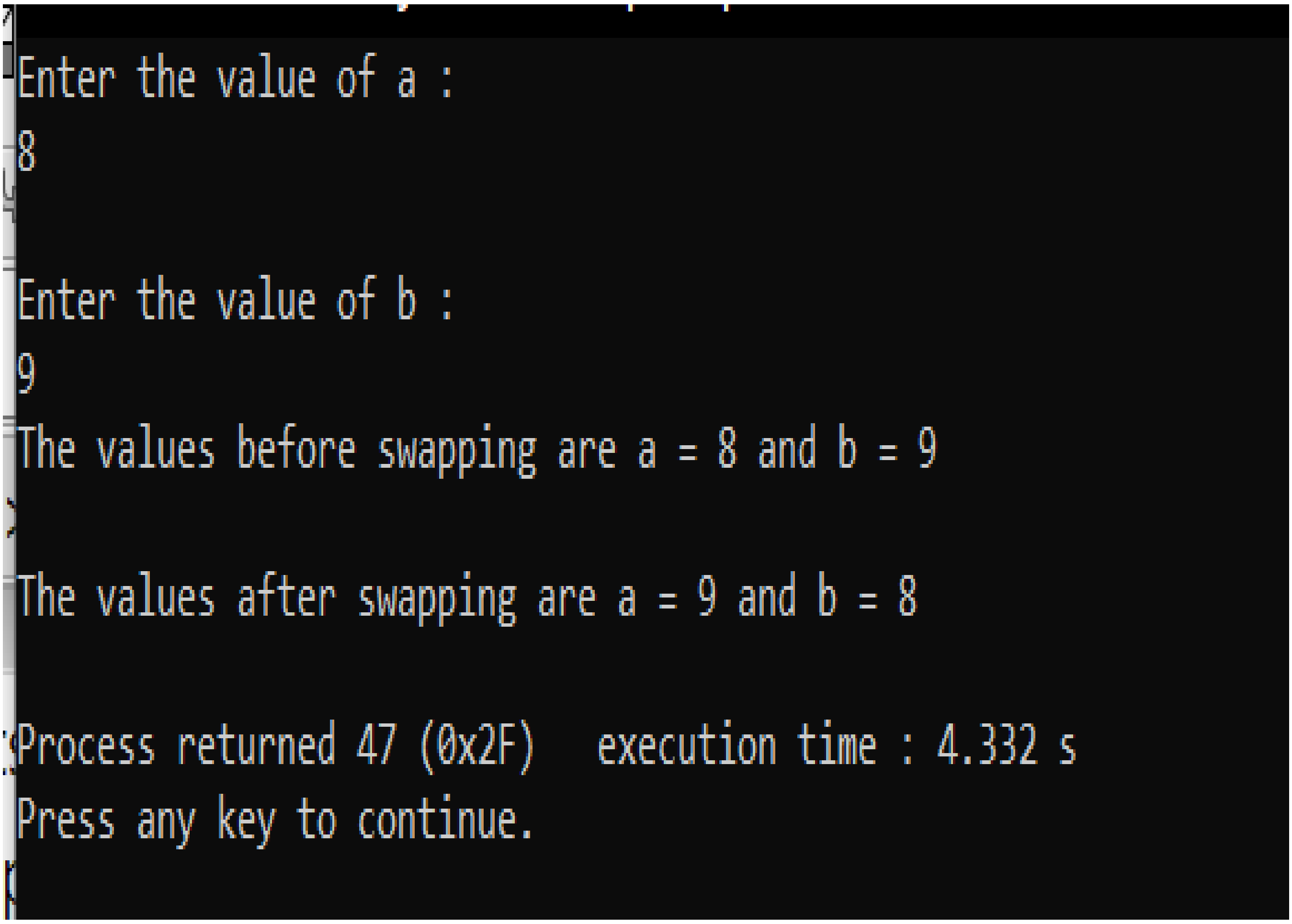

Start
"Enter a number a"
GET a
"Enter another number b"
GET b
PUT "The vaues of a and b given are : "+a+" and "+b¶
temp ← a
a ← b
b ← temp
PUT "The value of a is "+a+" The value of b is "+b+" after swapping ."¶
End

PROGRAM:

```
#include<stdio.h>
void main(){
  int a,b,temp;
```

```
    printf("Enter the value of a : \n");
    scanf("%d",&a);
    printf("\n");
    printf("Enter the value of b : \n");
    scanf("%d",&b);
    printf("The values before swapping are a = %d and b =
%d \n",a,b);
    temp = a;
    a = b;
    b = temp;
    printf("\n");
    printf("The values after swapping are a = %d and b = %d
\n",a,b);
}
```

OUTPUT:



```
Enter the value of a :
8

Enter the value of b :
9
The values before swapping are a = 8 and b = 9

The values after swapping are a = 9 and b = 8


Process returned 47 (0x2F)   execution time : 4.332 s
Press any key to continue.
```

1.2)

AIM:

Given a set of n student’s examination marks (in the range 0-100) make a count of the number of students that passed the examination. A Pass is awarded for all of 50 and above.

DESCRIPTION:

The program developed here takes input for the total number of students in the class.

Takes input for marks of each student one of another.

If the marks are out of the specified range, it again asks to enter a valid marks.

When the whole inputs are given.

A variable count keeps a count of the students who passed the criteria of pass mark.

Prints the number of students passed.

ALGORITHM:

Step1: Enter the number of students in class into a variable n.

Step2: Initiate count to 0.

Step3: Initiate i=1.

Step4: if i>n

Print the number of students passed.

Step5: Else

Enter the marks of student no i.

Step6: if marks>= 0 and marks <=100

If yes,

Compare if marks>=50

Step7: if yes,

Count++,

i++. Go to step 4.

Step8: if no ,

i++, Go to step 4.

Step9: from Step6 if no,

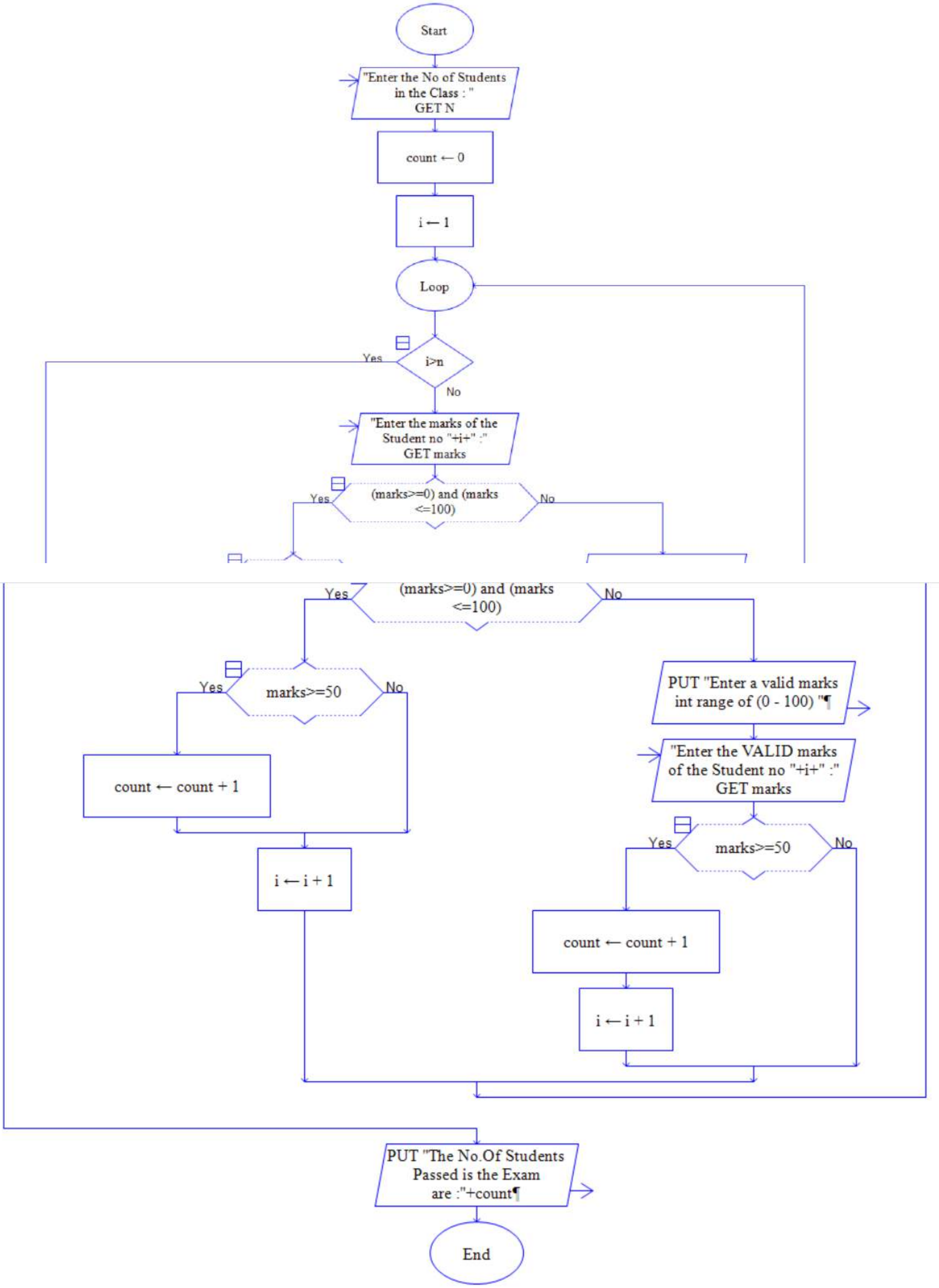
Step 10: Enter a VALID marks of the student int range of (0-100).

Step 11: input marks of student i.

Step12: Repeat step no 7.

Step13: End.

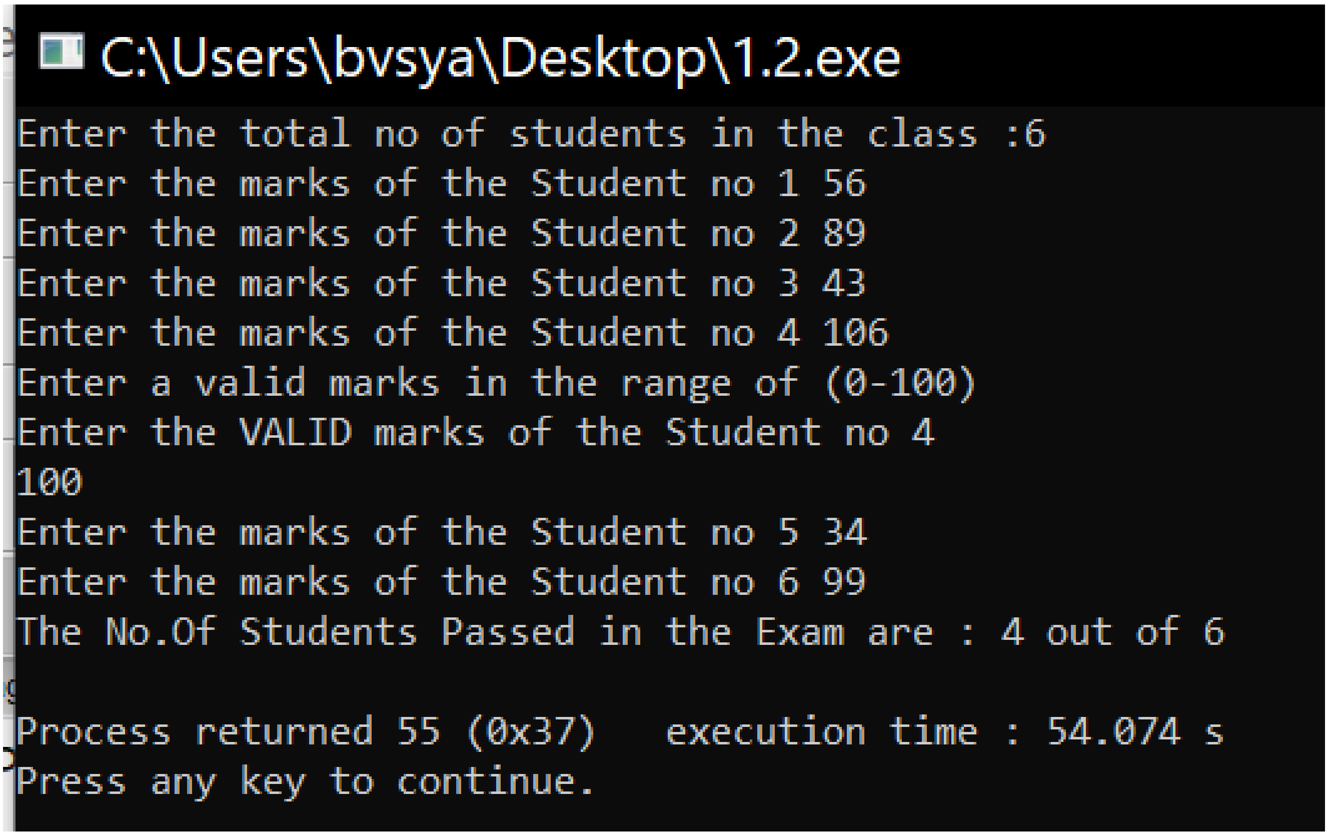
FLOWCHART:


Start
"Enter the No of Students in the Class : " GET N
count ← 0
i ← 1
Loop
i>n
Yes
No
"Enter the marks of the Student no "+i+" :" GET marks
(marks>=0) and (marks <=100)
Yes
No
marks>=50
Yes
No
count ← count + 1
i ← i + 1
PUT "Enter a valid marks int range of (0 - 100) "¶
"Enter the VALID marks of the Student no "+i+" :" GET marks
marks>=50
Yes
No
count ← count + 1
i ← i + 1
PUT "The No.Of Students Passed is the Exam are :"+count¶
End

PROGRAM:

```
#include<stdio.h>
void main(){
int n,i,marks,count;
printf("Enter the total no of students in the class :");
scanf("%d",&n);
count=0;
i=1;
while(i<=n){
        printf("Enter the marks of the Student no %d ",i);
        scanf("%d",&marks);
        if((marks>=0)&&(marks<=100)){
            if(marks>=50){
                count++;
                i++;
            }
            else{
                i++;
            }
        }
        else{
            printf("Enter a valid marks in the range of (0-100)\n");
        printf("Enter the VALID marks of the Student no %d\n",i);
        scanf("%d",&marks);
        if(marks>=50){
          count++;
          i++;
        }
      }
}
printf("The No.Of Students Passed in the Exam are : %d out of
%d\n",count,n);
}
```

OUTPUT:



```
Enter the total no of students in the class :6
Enter the marks of the Student no 1 56
Enter the marks of the Student no 2 89
Enter the marks of the Student no 3 43
Enter the marks of the Student no 4 106
Enter a valid marks in the range of (0-100)
Enter the VALID marks of the Student no 4
100
Enter the marks of the Student no 5 34
Enter the marks of the Student no 6 99
The No.Of Students Passed in the Exam are : 4 out of 6

Process returned 55 (0x37)   execution time : 54.074 s
Press any key to continue.
```

1.3)

AIM:

Given a set of n numbers design an algorithm that adds these numbers and returns the resultant sum. Assume N is greater than or equal to zero.

DESCRIPTION:

- ❖ The algorithm developed here takes input n
- ❖ Adds the input to sum if it is a positive number or 0.
- ❖ Returns the sum of the inputs given

ALGORITHM:

Step 1: START.

Step 2: Enter the number of numbers to be added .Take the value into variable n.

Step 3: Initiate i=1,sum=0.

Step 4: Check if i>n

If yes,

End the programme.

Step 5: If no,

Take input for N.

Step 6: check if N>=0.

Step 7: if yes,

Sum= sum +N. (SUM IS GETTING UPDATED)

i++. (INCREMENTING i by 1)

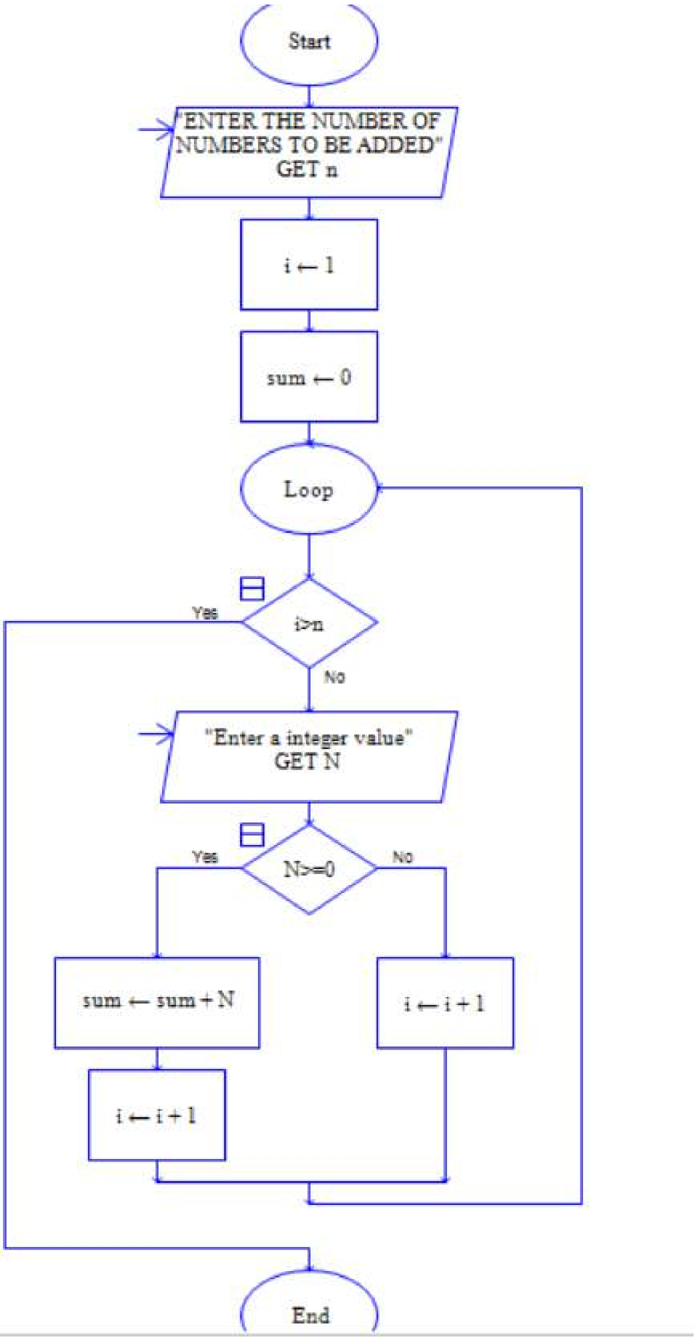
Go to step no 4.

Step 8: if no,

i++.

Go to step no 4.

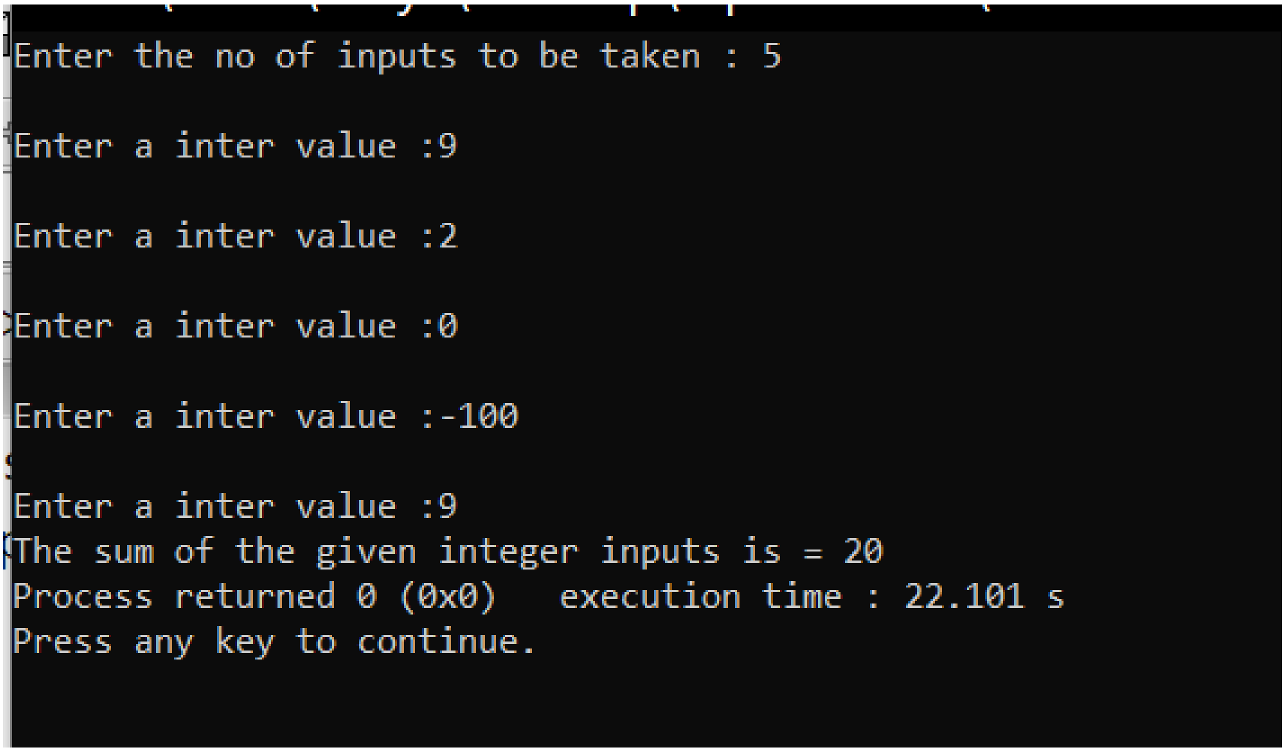
FLOWCHART:



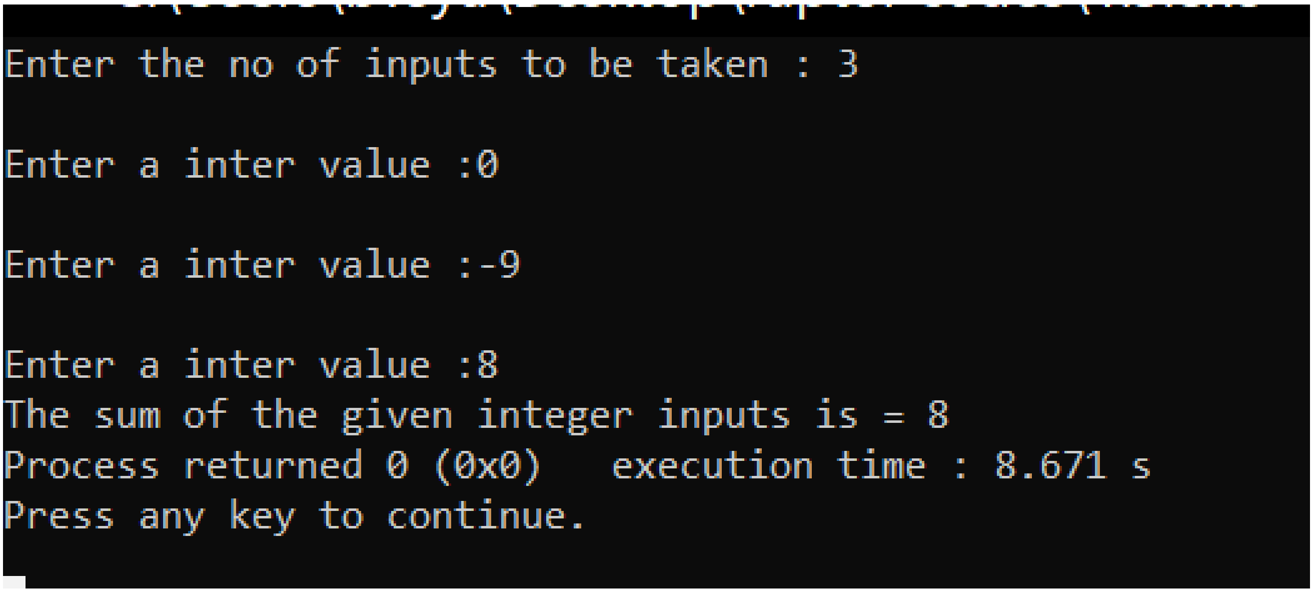
PROGRAM:

```
#include<stdio.h>

int main(){

int i,n,N,sum;

printf("Enter the no of inputs to be taken : ");

scanf("%d",&n);

i=1;

sum=0;

while(i<=n){

        printf("\nEnter a inter value :");

        scanf("%d",&N);

        if(N>=0){

           sum = sum + N;

           i++;

        }

        else{

           i++;

        }

}

printf("The sum of the given integer inputs is = %d",sum);

return 0;

}
```

Output:



```
Enter the no of inputs to be taken : 5

Enter a inter value :9

Enter a inter value :2

Enter a inter value :0

Enter a inter value :-100

Enter a inter value :9
The sum of the given integer inputs is = 20
Process returned 0 (0x0)   execution time : 22.101 s
Press any key to continue.
```



```
Enter the no of inputs to be taken : 3

Enter a inter value :0

Enter a inter value :-9

Enter a inter value :8
The sum of the given integer inputs is = 8
Process returned 0 (0x0)   execution time : 8.671 s
Press any key to continue.
```

EXPERIMENT NO 2.

INTRODUCTION TO C PROGRAMMING

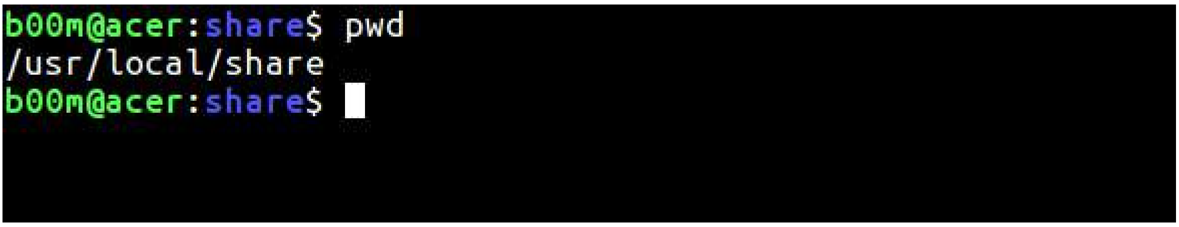
2.1) Basic Linux Commands.

Basic linux commands list

Now we'll discus about some basic **linux commands with examples**, you're almost always going to need those commands, so better to remember them. However from my experience, it's much easier to remember if you write them with pen on paper, rather than just typing on terminal.

1. pwd command

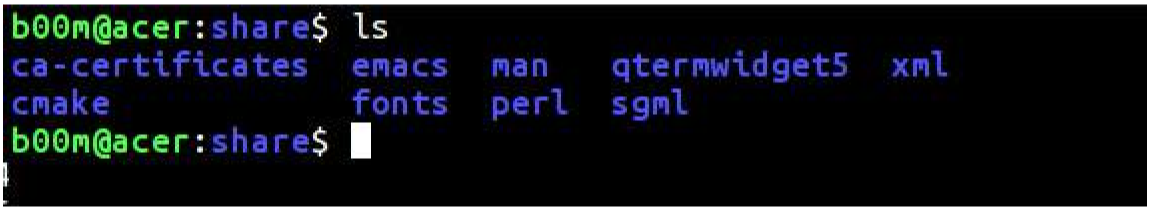
This command prints the location of your current working directory. It's important to know actually where you're before going to a parent or sub directories.



```
b00m@acer:share$ pwd
/usr/local/share
b00m@acer:share$ 
```

2. ls command

ls is one of the most used basic linux commands, used to **print** contents of a directory, by default it lists contents of current working directory(**pwd**).

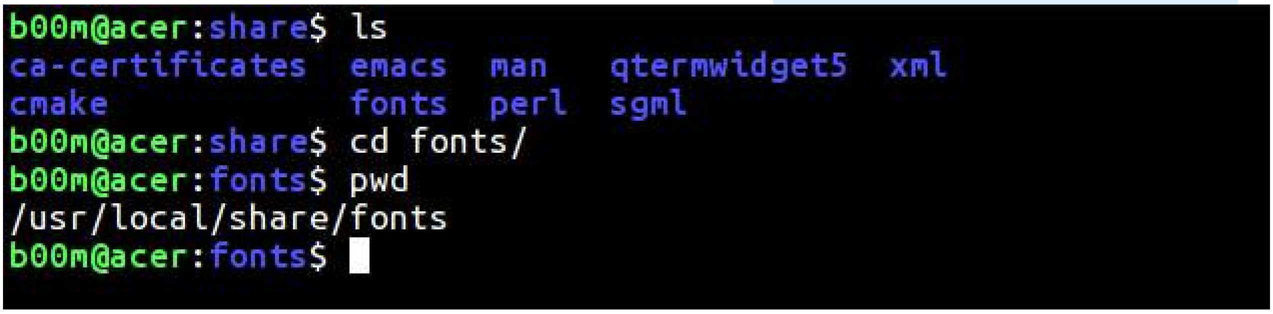


```
b00m@acer:share$ ls
ca-certificates  emacs  man   qtermwidget5  xml
cmake            fonts  perl  sgml
b00m@acer:share$ 
```

Example, use ls /usr/bin to list contents of the **/usr/bin** folder.

3. cd command

After knowing your **pwd** and getting an overview with the **ls**, it's time to move around with **cd** command. Clarification, assume you're on your **Home** directory, you need to go to the **/usr/local/share/fonts** directory, use cd /usr/local/share/fonts.



```
b00m@acer:share$ ls
ca-certificates  emacs  man   qtermwidget5  xml
cmake            fonts  perl  sgml
b00m@acer:share$ cd fonts/
b00m@acer:fonts$ pwd
/usr/local/share/fonts
b00m@acer:fonts$ 
```

There's three shortcut, if you need to move one directory up, use cd .. and go straight to your Home folder with cd, and use cd - to go back to your last working directory.

4. cat command

It's used to print the contents of a file to the screen(**stdout** more precisely), really useful when you want to have a quick look on contents of a file. As example, use cat a_text_file to get the inside contents of that file in your screen.

5. cp command

cp , You can copy files and directories with this command. Typical usage is like cp file_a file_1_copy or cp directory_a dir_a_copy Also don't forget to use proper path when you're coping something to different location.

6. mv command

The mv command is used to **move** or **rename** directories and files. To rename a file use mv old_name new_name.

7. rm command

The `rm` command is used to remove directory or files. Like use `rm -r /tmp/backup` to remove everything that folder. Of course you've to be careful before removing anything.

8. mkdir command

mkdir, it's used to make a new directory in linux. Example, use `mkdir my_new_dir` to make a new directory named my_new_directory. The `-p` argument is useful, when you don't want to make parent directories manually.

9. rmdir command

rmdir, if you need to remove a directory, use this command. As example, use `rmdir my_dir` to remove that specific directory. More details about the rmdir command.

10. touch command

touch, It's the equivalent command of mkdir for files. You can create a blank file with touch command. As example, use `touch ~/Public/index.html` to create a blank index.html file under the Public directory.

11. In command

This command is used to make link between files and directories. As example, you need to make a symbolic link of the `/var/www` directory to the `/tmp` directory.

```
ln -s /var/www/ /tmp/
```

To un-link that symlink, use

```
unlink /tmp/www
```

You've to be extra careful with complete path and trailing slashes while linking and un-linking.

12. sudo command

sudo , that's an essential yet potentially dangerous command. Whenever you're getting a Permission denied, Authorization failed or something like that use sudo.

As example, the /var/www directory is not writable by the normal user. So to create a blank **index.html** file under the **/var/www** directory use `sudo touch /var/www/index.html`

13. head command

If you need to print first few lines of a file(any type) then you can use head command. A nice practical example w'd be

```
head -20 /var/log/syslog
```

This will print the first 20 lines of the **rsyslogd** log to the stdout. By default head command prints first 10 lines.

14. tail command

It's similar to the head command, but the function is opposite, prints last 10 lines of any file by default. Here's an example, how to print last 30 lines of the kernel log.

```
tail -30 /var/log/kern.log

15. chmod command
```

It's also a very important command, used to change file and directory permission. As the chmod command is a very long topic, so here I'll explain it in brief.

Basically there's three type of permission, read, write and execute. Each of them denoted by a number.

- 4 for **read** permission
- 2 for **write** permission
- 1 for **execute** permission

So if you need to set universal read/write permission to a file, you can use

```
chmod 666 my_file_name.
```

Assume you need to make a script executable, you can use.

```
chmod +x my_script_name
```

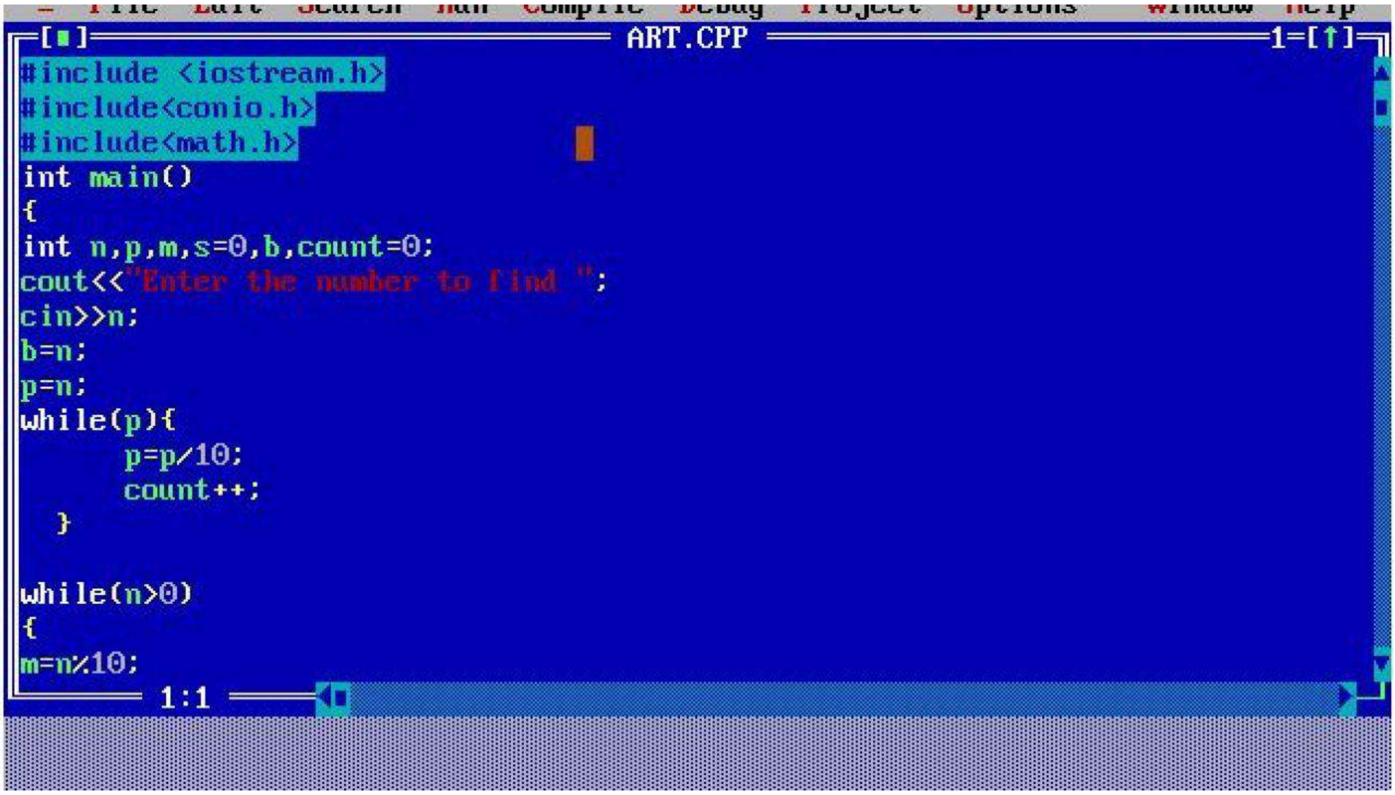
There'll be a full chmod tutorial very soon, to explain you in detail.

2.2) EXPOSURE TO Turbo C, Vi, Emacs, Code Blocks IDE, Dev C++.

Turbo C:

Well Turbo C is an IDE which is used for C language programming. It has source code editor, compiler, linker, inbuilt debugger and an offline help for reference.

It used to be a popular compiler a long time ago when you start learning C language in your school days(at-least my school made me use this IDE). But it’s use had been discontinued after arrival of much better IDEs like Code::Blocks, Visual Studio, Emacs, NetBeans, etc. having support for C/C++ compilers and debuggers like GCC, clang, MSVC, gdb, etc.


File Edit Search Run Compile Debug Project Options Window Help
ART.CPP
1:1

```
#include <iostream.h>
#include<conio.h>
#include<math.h>
int main()
{
int n,p,m,s=0,b,count=0;
cout<<"Enter the number to find ";
cin>>n;
b=n;
p=n;
while(p){
      p=p/10;
      count++;
  }

while(n>0)
{
m=n%10;
```

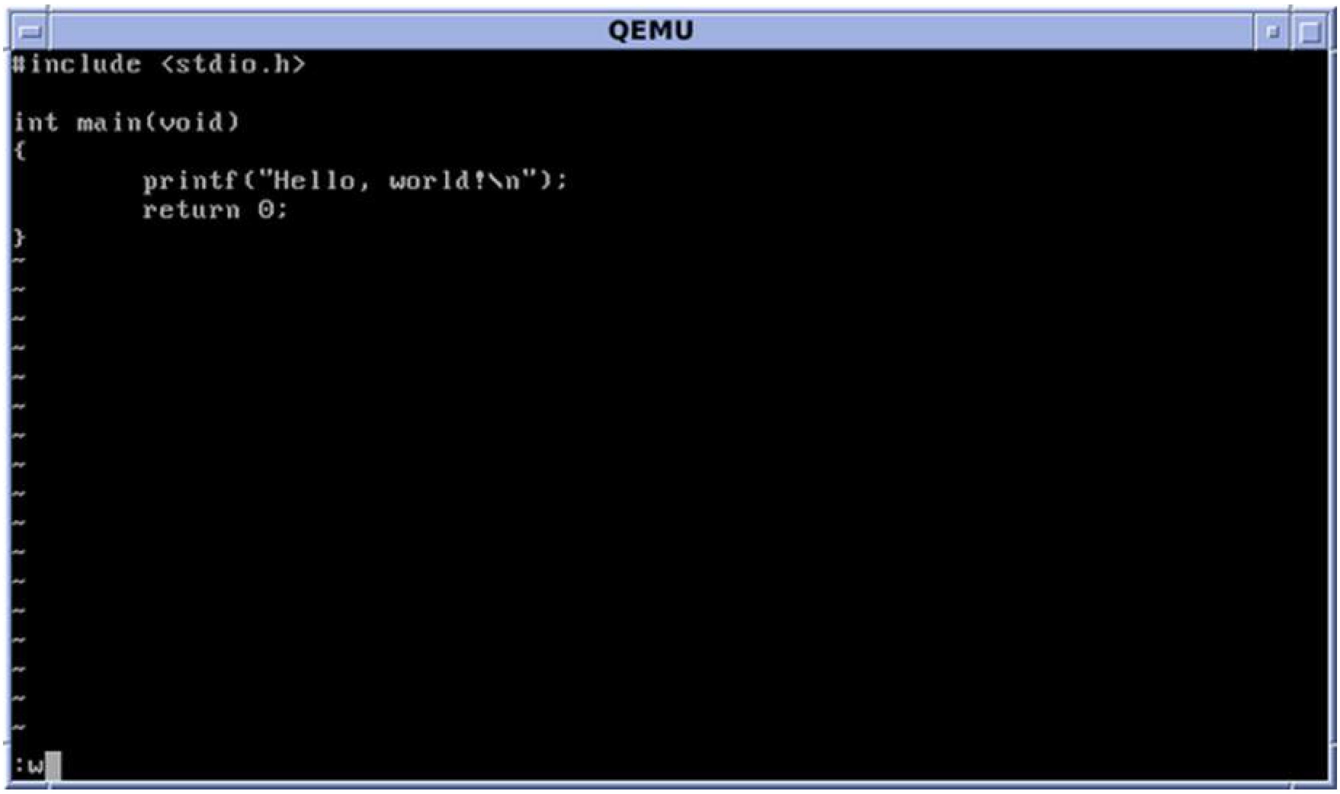
2.Vi TEXT EDITOR:

vi (pronounced as distinct letters., is a screen-oriented text editor originally created for the Unix operating system. The portable subset of the behavior of vi and programs based on it, and the ex editor language supported within these programs, is described by (and thus standardized by) the Single Unix Specification and POSIX.

The original code for vi was written by Bill Joy in 1976, as the visual mode for a line editor called ex that Joy had written with Chuck Haley. Bill Joy's ex 1.1 was released as part of the first Berkeley Software Distribution (BSD) Unix release in March 1978. It was not until version 2.0 of ex, released as part of Second BSD in May 1979 that the editor was installed under the name "vi" (which took users straight into ex's visual mode), and the name by which it is known today. Some current implementations of vi can trace their source code ancestry to Bill Joy; others are completely new, largely compatible reimplementation.

The name "vi" is derived from the shortest unambiguous abbreviation for the ex command visual, which switches the ex line editor to visual mode. The name is pronounced /ˈviːˈaɪ/ (the English letters *v* and *i*).

In addition to various non–free software variants of vi distributed with proprietary implementations of Unix, vi was opensourced with OpenSolaris, and several free and open source software vi clones exist. A 2009 survey of *Linux Journal* readers found that vi was the most widely used text editor among respondents, beating gedit, the second most widely used editor, by nearly a factor of two (36% to 19%)



```
#include <stdio.h>

int main(void)
{
        printf("Hello, world!\n");
        return 0;
}
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:w
```

EMACS:

Emacs is a text editing tool that comes out-of-the-box with Linux and macOS. As a (less popular) cousin of Vim, Emacs also offers powerful capabilities with easy-to-install language support, and can even help you navigate faster in macOS with the same keybindings.

Emacs vs. IDEs

There’s the ancient question of whether text editors or IDEs (e.g., the JetBrains suite or Eclipse) are better for everyday coding activities. Everyone has different preferences, and the best way for you to find yours is to try both. You may find out that a hybrid approach works best for you.

CODE BLOCKS IDE:

Code::Blocks is an open-source IDE that uses **C, C++, and Fortran coding languages**. The main functionality of the program is the focus around a plugin-based extension platform enabling each coder to develop the software in the way that they want to. The software was made in C++ and operates as a GUI tool.

Powerful programming tool

Having the proper tools to code is highly important. Blocks was built as a **platform** to develop out of combining all of the essentials into one program. The unique thing about Block is that you can do what you want with it.

As the software is **open-source**, those with programming knowledge have the ability to make modifications and improve the software. Even if you are using it without the confidence to change the code, you can add plugins to extend functionality or change features.

Compiling and **debugging** already come in the initial download however. Also, the feel of the platform remains consistent even when importing new plugins into the system. It uses **xxWidgets** to keep the software smooth across multiple platforms.

DEV C++:

Dev-C++ is a free integrated development environment (IDE) for programming in C/C++. Dev-C++ is developed by Bloodshed software. It is shipped with the open source MinGW compiler. MinGW uses GCC,the GNU g++ compiler collection. With Dev-C++ you can write Windows or console-based C/C++ programs easily, you can even create installer for your application. Dev-C++ is hosted on Sourceforge. Current available version is 4.9.9.2(i.e Version 5 Beta). There is no news of recent updates for this IDE. Also Dev-C++ runs solely on windows, linux port no longer exists.

DevPaks

DevPaks is the most famous extention of Dev-C++. Devpaks are usually libraries that contains GUI utilities,Toolkits, Compression libraries ,Graphic libraries etc. Devpaks for famous toolkits like Wxwidgets,   GTK ,python, OpenGL are also available. There are many devpaks available for more advanced function use. These packs contain precompiled version of the library ,so that any new user can download & develop without having to worry about library. Devpaks’s website has a list of paks in various categories.

2.3)

AIM: Writing simple programs using printf(), scanf() .

DESCRIPTION:

- This section illustrates the usage of printf and scanf statements in C, using two simple programs described along with their flowcharts.

Printf():

Aim: A basic printf program printing safety guidelines for Covid 19 protection.

DESCRIPTION:

- The program is designed to print the safety measues to be followed during covid 19 with a printf statement.

- When the user starts the program it automatically prints the instructions on the screen of the device using.

ALGORITHM:

STEP 1: START.

Step 2: print MEASURES FOR PREVENTION AGAINST COVID 19

Step 3: print WEAR MASK

Step 4: print MAINTAIN SOCIAL DISTANCING OF ABOUT 3 METERS

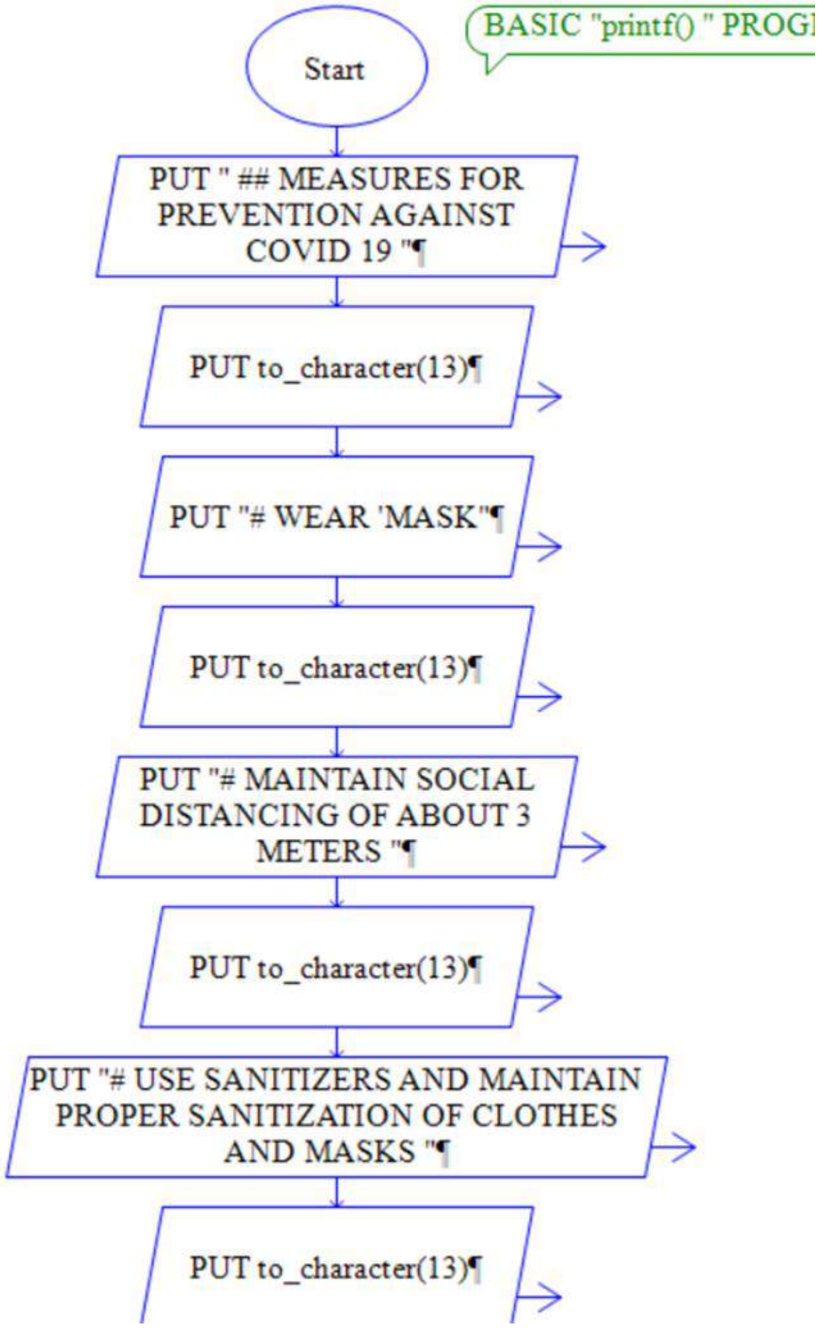
Step 5: print USE SANITIZERS AND MAINTAIN PROPER SANITIZATION OF CLOTHES AND MASKS

Step 6: print WASH HANDS WITH SOAP FOR ATLEAST 40 SEC

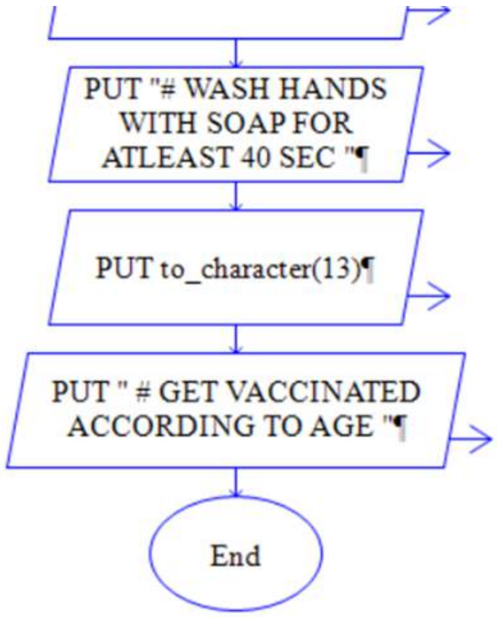
step 7: print GET VACCINATED ACCORDING TO AGE

STEP 8: END.

FLOWCHART:



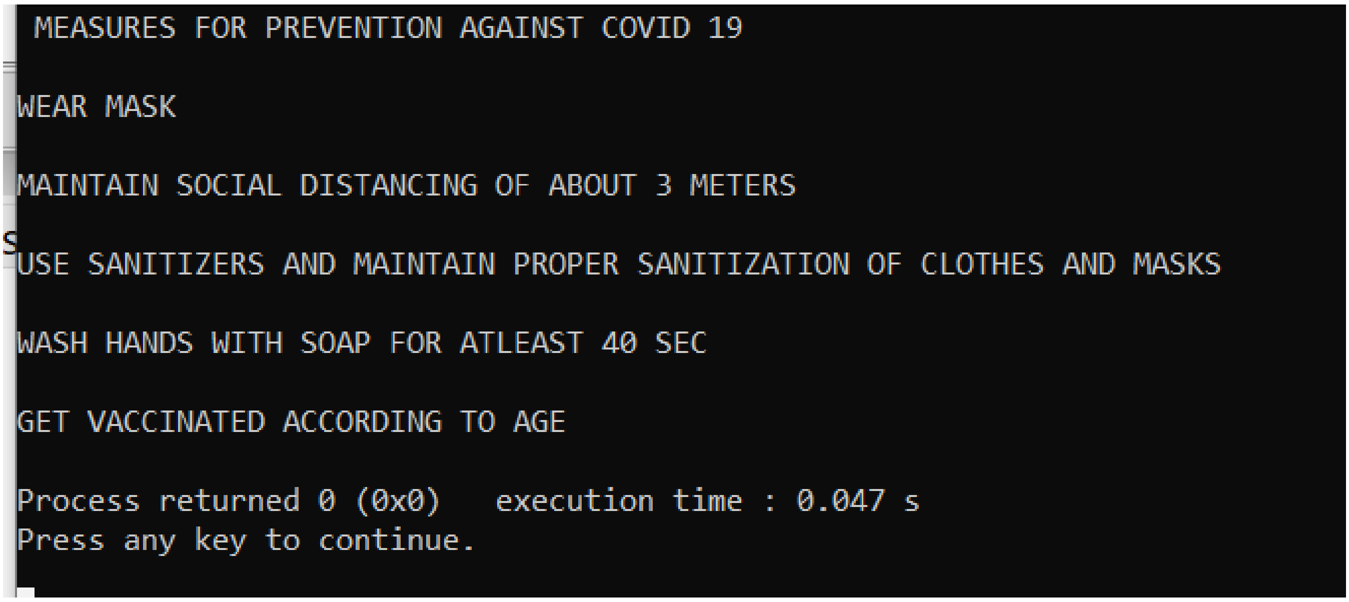
```
PUT " ## MEASURES FOR PREVENTION AGAINST COVID 19 "¶
PUT to_character(13)¶
PUT "# WEAR 'MASK"¶
PUT to_character(13)¶
PUT "# MAINTAIN SOCIAL DISTANCING OF ABOUT 3 METERS "¶
PUT to_character(13)¶
PUT "# USE SANITIZERS AND MAINTAIN PROPER SANITIZATION OF CLOTHES AND MASKS "¶
PUT to_character(13)¶
```


PUT "# WASH HANDS WITH SOAP FOR ATLEAST 40 SEC "¶
PUT to_character(13)¶
PUT " # GET VACCINATED ACCORDING TO AGE "¶
End

PROGRAM:

```
#include<stdio.h>

int main(){

  printf(" MEASURES FOR PREVENTION AGAINST COVID 19 \n");

  printf(" \nWEAR MASK\n");

  printf(" \nMAINTAIN SOCIAL DISTANCING OF ABOUT 3 METERS \n");

  printf(" \nUSE SANITIZERS AND MAINTAIN PROPER SANITIZATION OF CLOTHES AND MASKS \n");

  printf(" \nWASH HANDS WITH SOAP FOR ATLEAST 40 SEC \n ");

  printf(" \nGET VACCINATED ACCORDING TO AGE \n");

return 0;

}
```

OUTPUT:



```
 MEASURES FOR PREVENTION AGAINST COVID 19

WEAR MASK

MAINTAIN SOCIAL DISTANCING OF ABOUT 3 METERS

USE SANITIZERS AND MAINTAIN PROPER SANITIZATION OF CLOTHES AND MASKS

WASH HANDS WITH SOAP FOR ATLEAST 40 SEC

GET VACCINATED ACCORDING TO AGE

Process returned 0 (0x0)   execution time : 0.047 s
Press any key to continue.
```

Scanf():

Aim: A basic SCANF program.

DESCRIPTION:

- **The program developed prints some text as ouput by taking some text as input with the use of scanf.**

ALGORITHM:

Step 1: START.

Step 2: declare a character NAME.

Step3: print **ENTER THE NAME OF THE PERSON.**

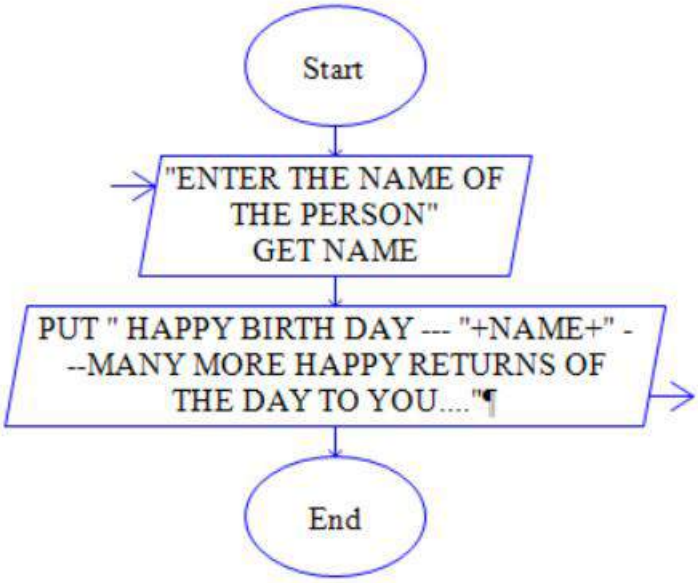
Step 4: take input for NAME.

Step 5: print as ouput the following

" HAPPY BIRTH DAY --- %s (NAME)---MANY MORE HAPPY RETURNS OF THE DAY TO YOU....".

Step 6: END.

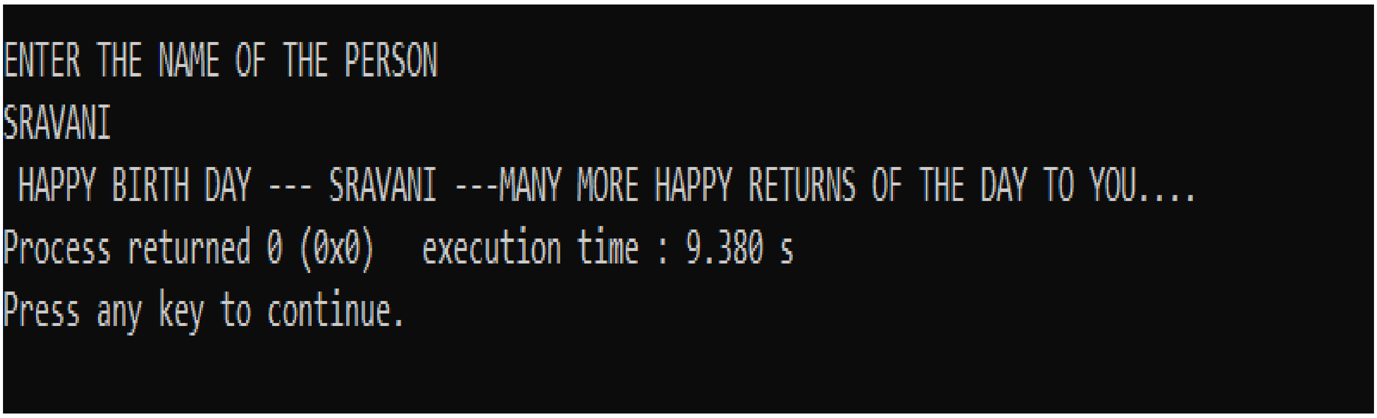
FLOWCHART:


Start
"ENTER THE NAME OF THE PERSON"
GET NAME
PUT " HAPPY BIRTH DAY --- "+NAME+" ---MANY MORE HAPPY RETURNS OF THE DAY TO YOU...."¶
End

PROGRAM:

```
#include<stdio.h>

int main(){

    char NAME;

   printf("\nENTER THE NAME OF THE PERSON\n");



   scanf("%s",&NAME);

   printf(" HAPPY BIRTH DAY --- %s ---MANY MORE HAPPY RETURNS OF
THE DAY TO YOU....");

return 0;
}
```

OUTPUT:



```
ENTER THE NAME OF THE PERSON
SRAVANI
 HAPPY BIRTH DAY --- SRAVANI ---MANY MORE HAPPY RETURNS OF THE DAY TO YOU....
Process returned 0 (0x0)   execution time : 9.380 s
Press any key to continue.
```

EXPERIMENT NO 3
RAPTOR

3.1) AIM: Installation and Introduction to Raptor.

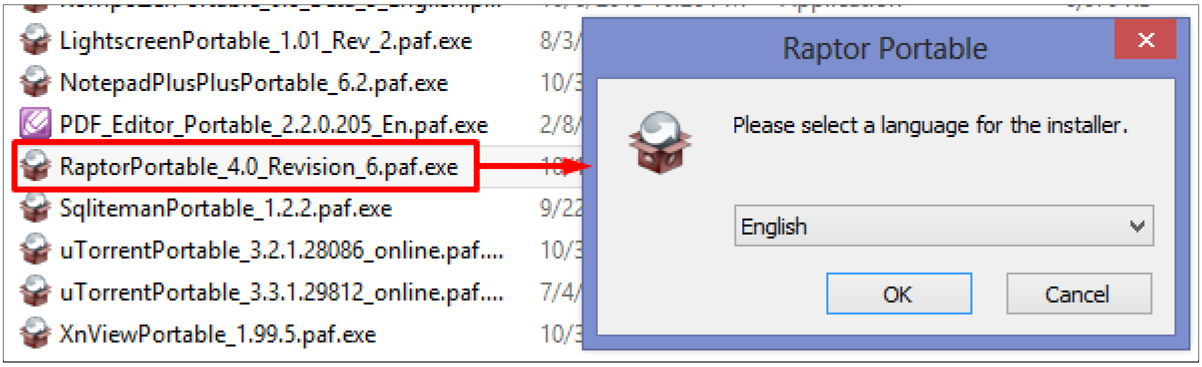
1) SETTING UP.

1.1) This tutorial is based on Raptor Portable. Download from here, http://raptor.martincarlisle.com/RaptorPortable_4.0_Revision_6.paf.exe

or from here,
https://docs.google.com/file/d/0B86b-ALn-1MGbnU0eG9pZFlJOVk/edit?usp=sharing

1.2) Run the RaptorPortable paf file.

Choose English Language.



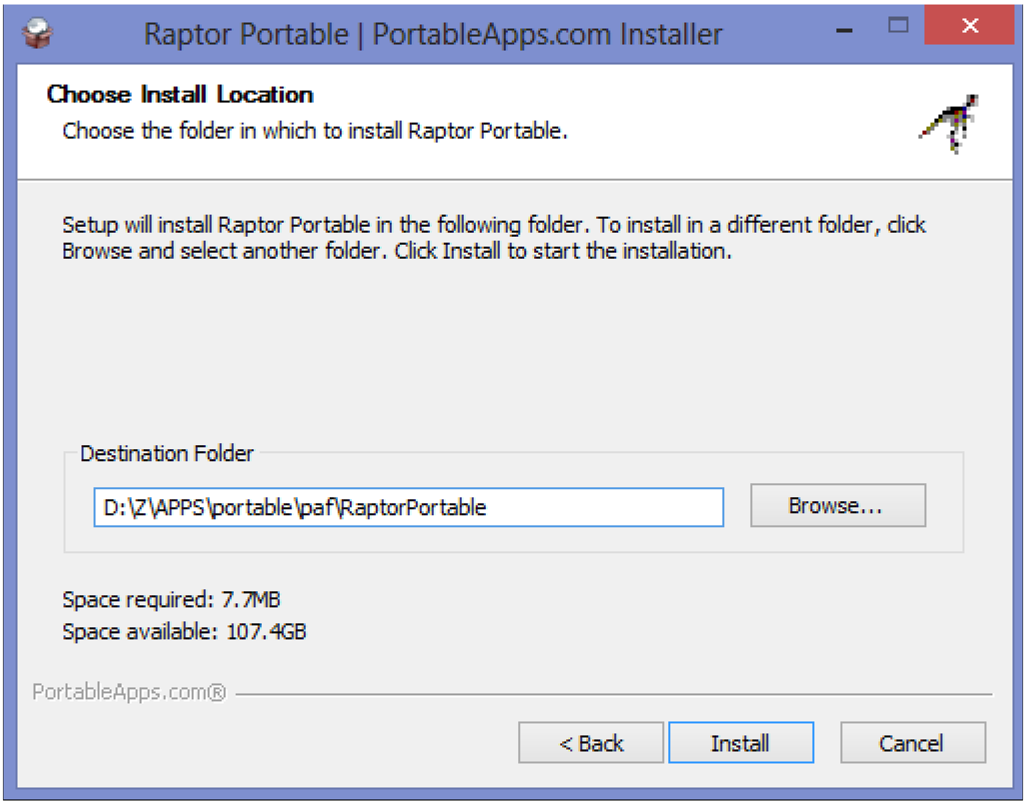
LightscreenPortable_1.01_Rev_2.paf.exe 8/3/
NotepadPlusPlusPortable_6.2.paf.exe 10/3
PDF_Editor_Portable_2.2.0.205_En.paf.exe 2/8/
RaptorPortable_4.0_Revision_6.paf.exe 10
SqlitemanPortable_1.2.2.paf.exe 9/22
uTorrentPortable_3.2.1.28086_online.paf.... 10/3
uTorrentPortable_3.3.1.29812_online.paf.... 7/4/
XnViewPortable_1.99.5.paf.exe 10/3

Raptor Portable

Please select a language for the installer.

English

1.3) Set the install location.



Raptor Portable | PortableApps.com Installer

Choose Install Location

Choose the folder in which to install Raptor Portable.

Setup will install Raptor Portable in the following folder. To install in a different folder, click Browse and select another folder. Click Install to start the installation.

Destination Folder

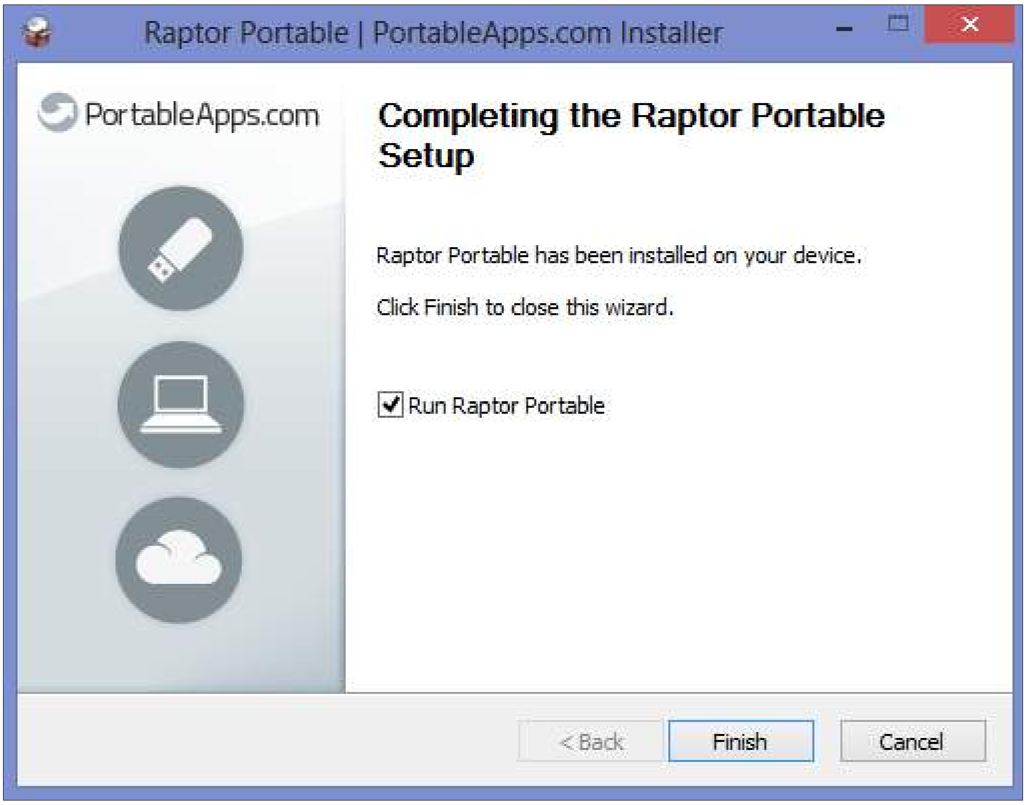
D:\Z\APPS\portable\paf\RaptorPortable

Space required: 7.7MB
Space available: 107.4GB

1.4) Setup process done.

Select Run Raptor Portable.

Click Finish.



Completing the Raptor Portable Setup

Raptor Portable has been installed on your device.

Click Finish to close this wizard.

☑ Run Raptor Portable

2) RUNNING RAPTOR FOR THE FIRST TIME.

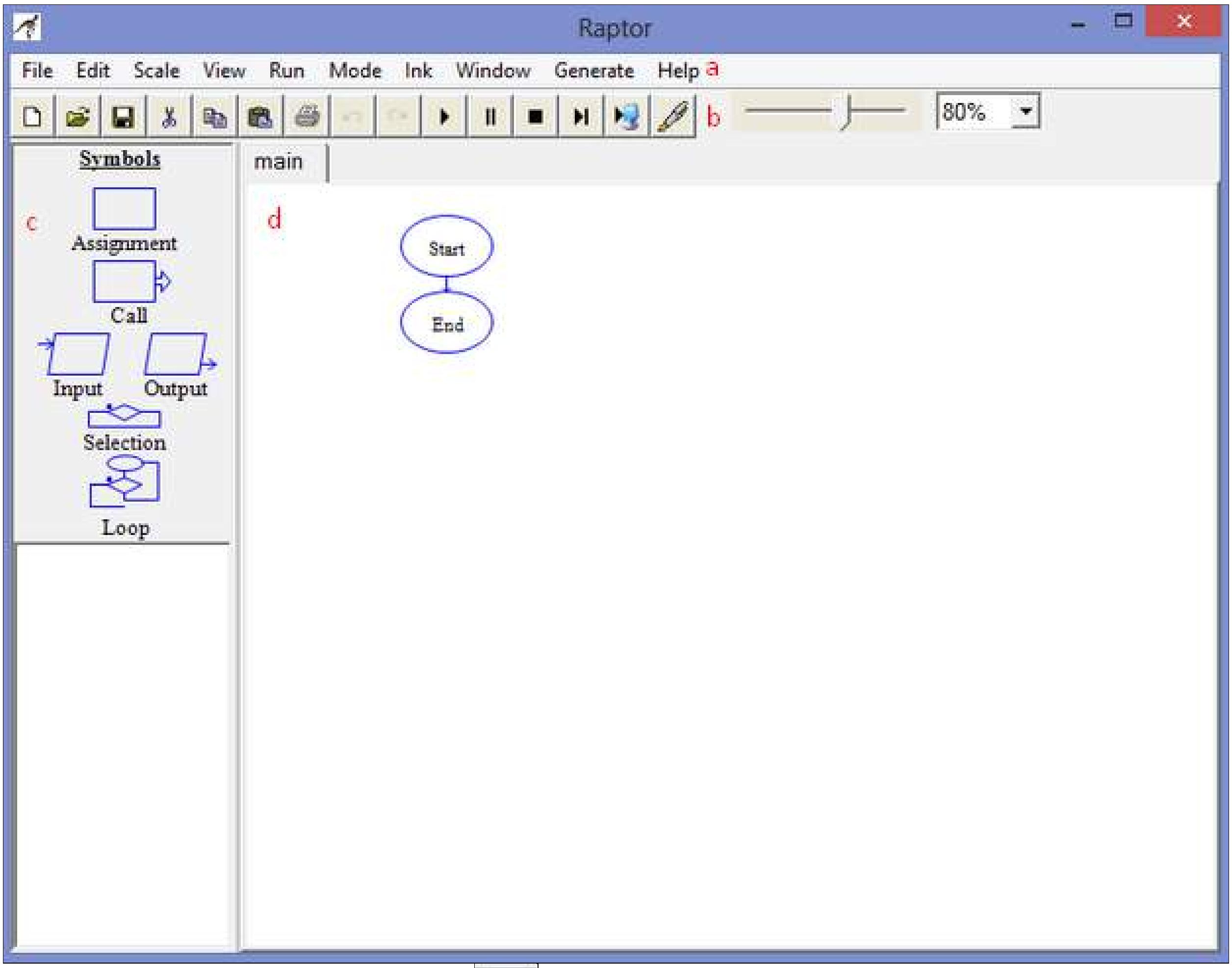
2.1) Raptor Main Application Window.

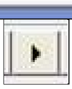
a) Menu bar.

b) Tool bar.

c) Symbols panel.

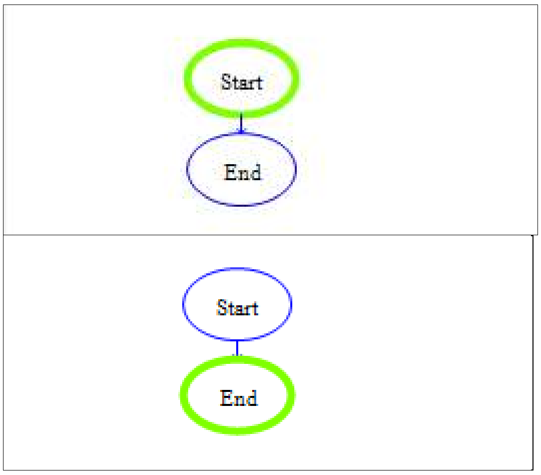
d) Main panel.



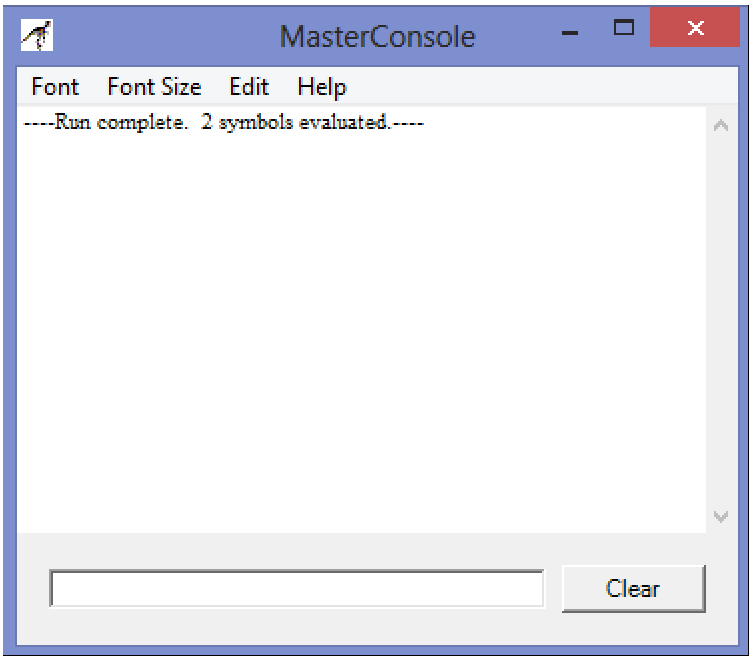
2.2) Click the Run button, 

The main panel shows the following image sequence.

It means that the program is running from Start to End.



2.3) Another window, called MasterConsole, pops up to summarize the events.

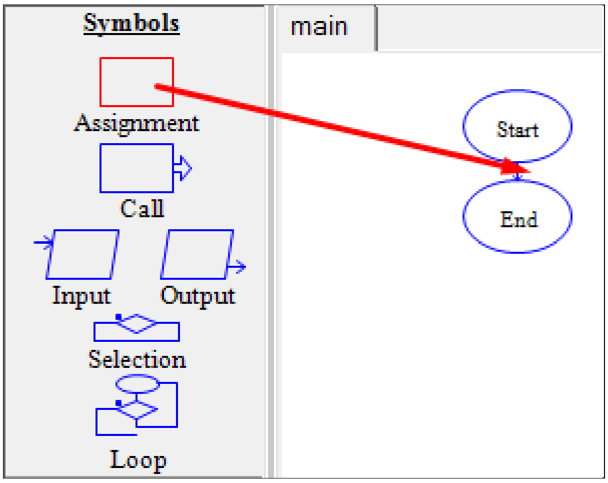


```
----Run complete.  2 symbols evaluated.----
```

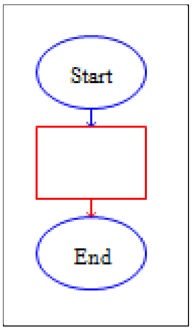
2.4) Save your chart as “testflow”.

3) ADDING SYMBOLS TO FLOWCHART

3.1) Click the symbol Assignment and drag it to a point in between Start and Stop.

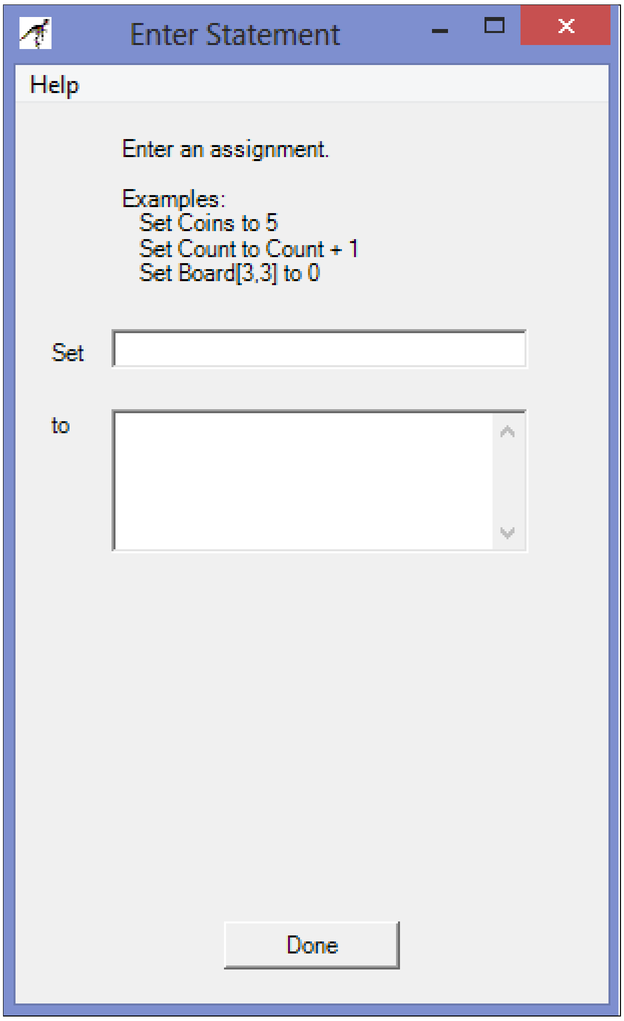


3.2) Outcome:



3.3) Double-click the red rectangle.

Enter Statement window pops up.



Enter an assignment.

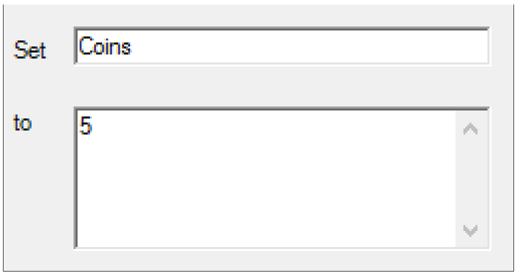
Examples:
  Set Coins to 5
  Set Count to Count + 1
  Set Board[3,3] to 0

Set

to

3.4) Type as follows:

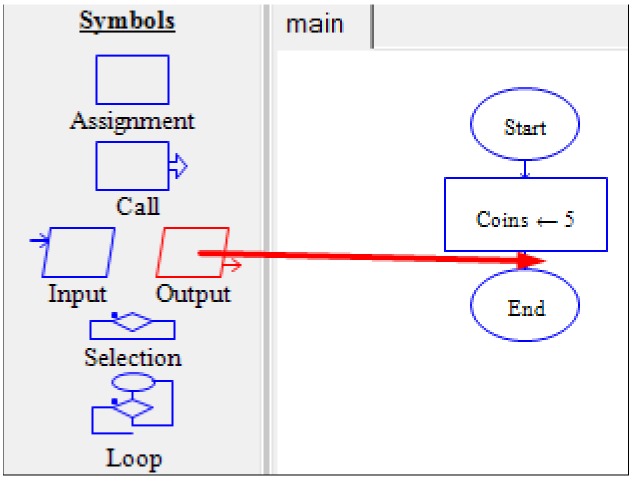
Click Done.



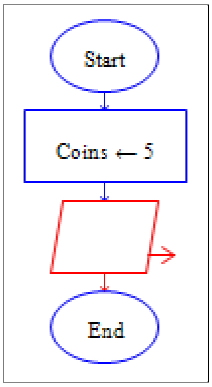
Set Coins

to 5

3.5) Click and drag the Output symbol to a point in between Assignment and End.

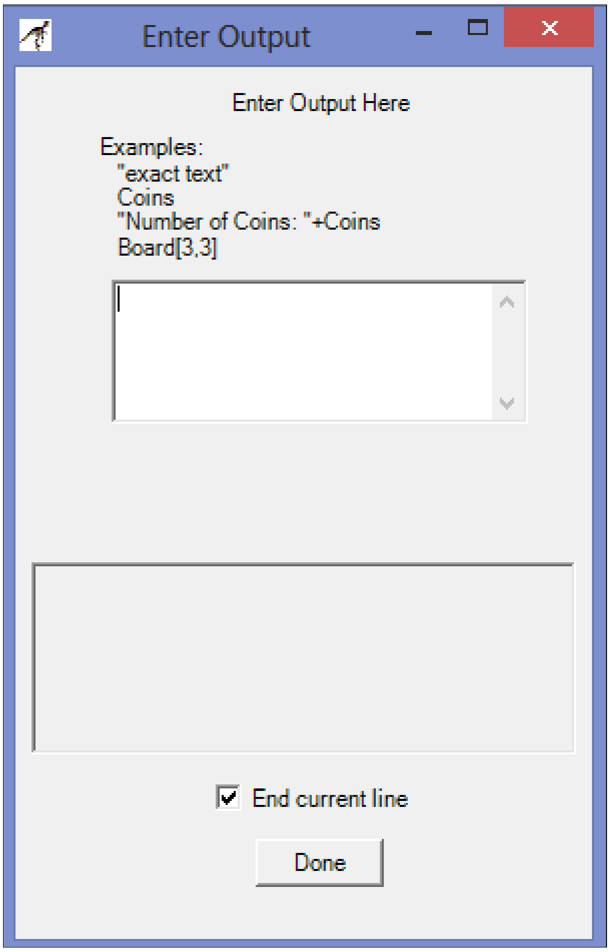


3.6) Outcome:



3.7) Double-click the Output symbol.

Enter Output window pops up.

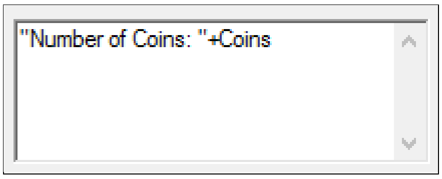


Enter Output Here

Examples:
"exact text"
Coins
"Number of Coins: "+Coins
Board[3,3]

☑ End current line

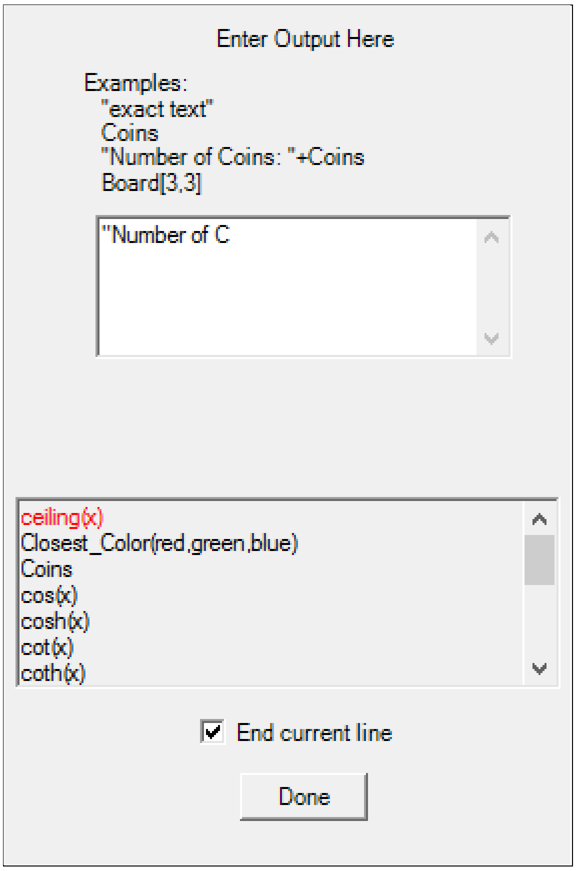
3.8) Type as follows:



"Number of Coins: "+Coins

As you type, you may get suggestions as follows:

a)



Enter Output Here

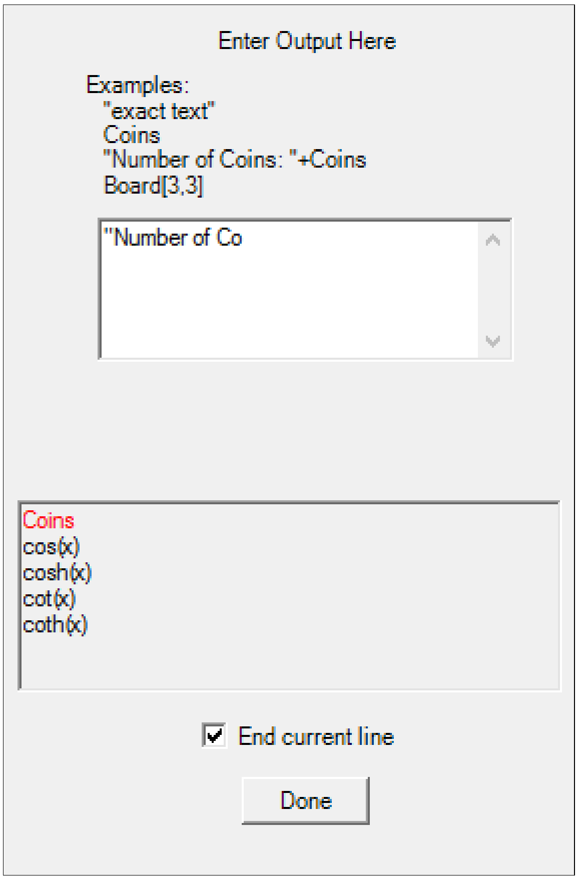
Examples:
"exact text"
Coins
"Number of Coins: "+Coins
Board[3,3]

```
"Number of C
```

ceiling(x)
Closest_Color(red,green,blue)
Coins
cos(x)
cosh(x)
cot(x)
coth(x)

☑ End current line

b)



Enter Output Here

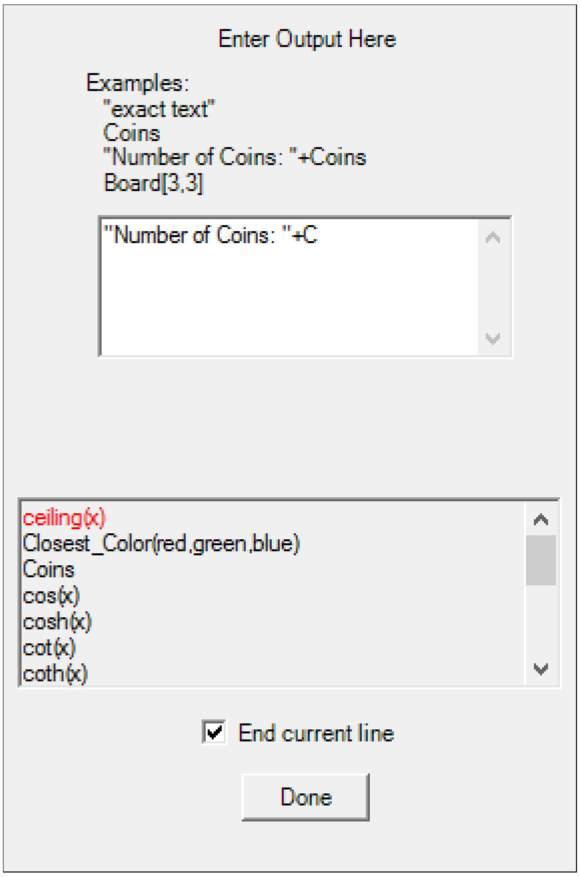
Examples:
"exact text"
Coins
"Number of Coins: "+Coins
Board[3,3]

```
"Number of Co
```

Coins
cos(x)
cosh(x)
cot(x)
coth(x)

☑ End current line

c)

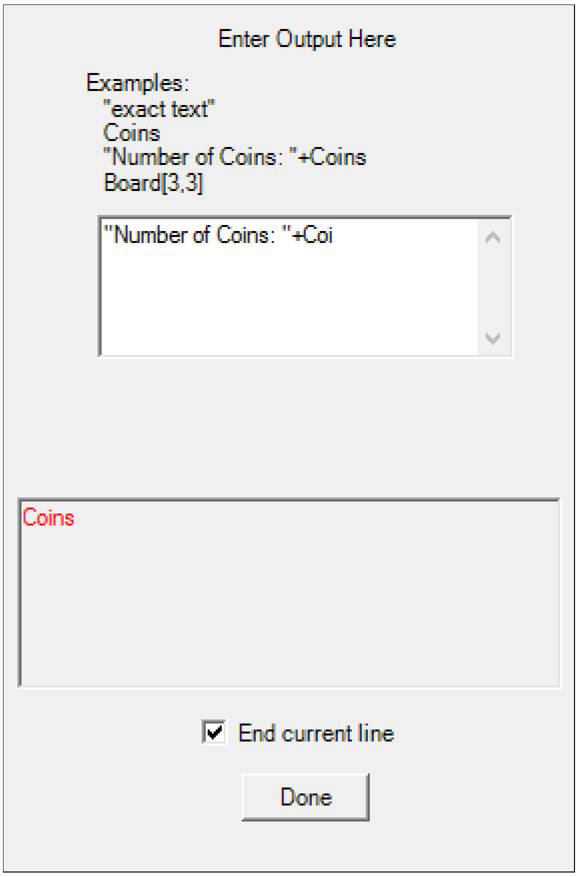


Enter Output Here

Examples:
"exact text"
Coins
"Number of Coins: "+Coins
Board[3,3]

```
"Number of Coins: "+C
```

d)

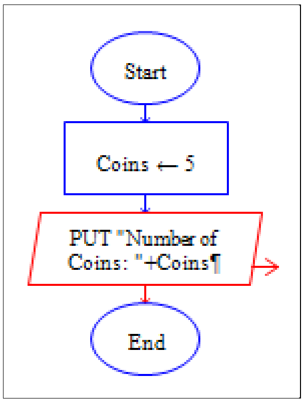


Enter Output Here

Examples:
"exact text"
Coins
"Number of Coins: "+Coins
Board[3,3]

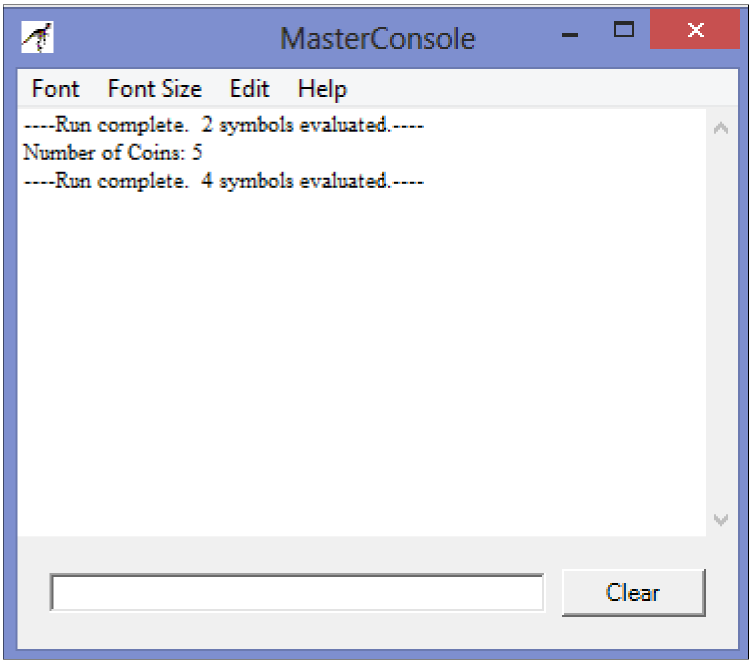
```
"Number of Coins: "+Coi
```

3.9) Outcome:



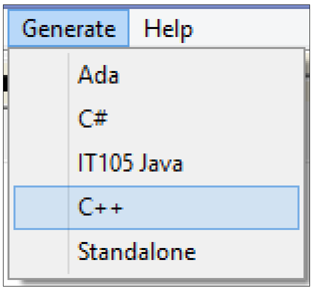
4) GENERATING CODES.

4.1) Click Run.



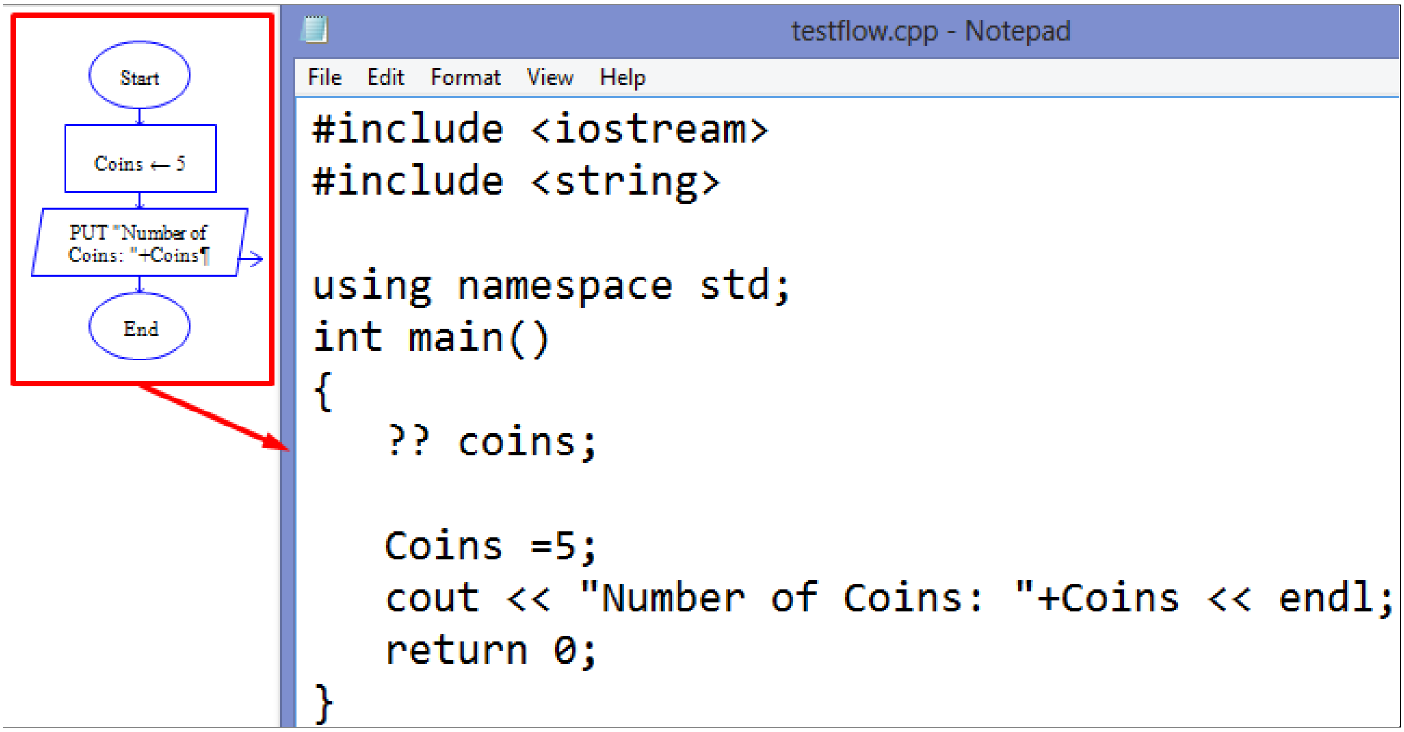
```
----Run complete.  2 symbols evaluated.----
Number of Coins: 5
----Run complete.  4 symbols evaluated.----
```

4.2) Go to menu Generate/C++



4.3) Outcome:

(A text file is generated and automatically opened by Notepad)



```
#include <iostream>
#include <string>

using namespace std;
int main()
{
   ?? coins;

   Coins =5;
   cout << "Number of Coins: "+Coins << endl;
   return 0;
}
```

4.4) Notice that the declaration statement ?? coins; requires you to set the data type for the variable coins, e.g int.

5) DOWNLOAD

https://docs.google.com/file/d/0B86b-ALn-1MGUVFhcmVud04ybHM/edit?usp=sharing

3.2)

AIM: Draw a flow chart to find the Sum of 2 numbers.

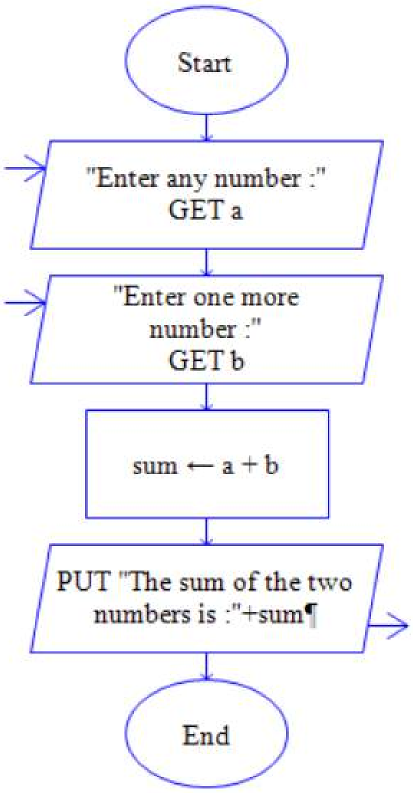
DESCRIPTION:

- ❖ The below flow chat developed using raptor tool shows how addition of two numbers is performed.

ALGORITHM:

To add two numbers.

1. Read the Value of A and B.
2. SUM = A+B.
3. Display SUM.
4. Stop.



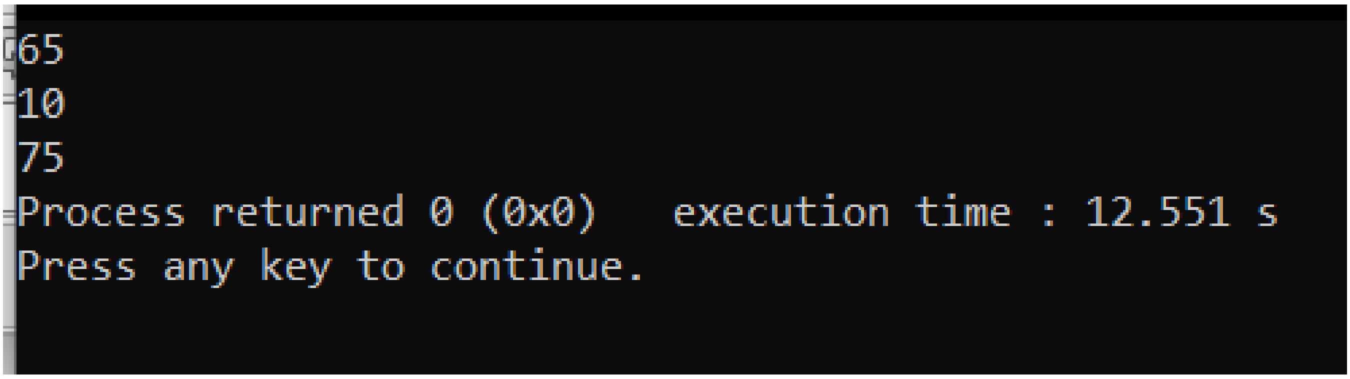
FLOWCHART:

PROGRAM:

#include<stdio.h>

```
int main()

   int a,b,sum;

   scanf("%d",&a);

   scanf("%d",&b);

   sum = a + b;

   printf("%d",sum);

return 0;

}
```

OUTPUT:



```
65
10
75
Process returned 0 (0x0)   execution time : 12.551 s
Press any key to continue.
```

3.3)

AIM: DRAW A FLOW CHART TO FIND

SIMPLE INTEREST.

DESCRIPTION:

- The algorithm developed in the following program helps in calculating simple interest of any given principal amount given rate of interest, time period.

ALGORITHM:

STEP 1: START.

STEP 2: GET PRINCIPAL AMOUNT

STEP 3: GET RATE OF INTREST

STEP 4: GET TIME

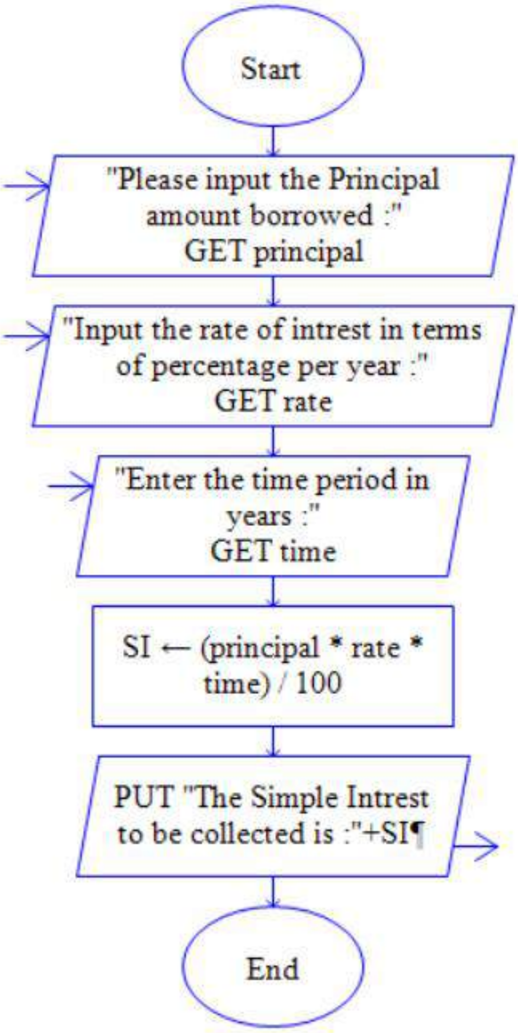
STEP 5: CALCULATE SI USING THE FORMULA

SI = (P*R*T)/100.

STEP 6: DISPLAY SI.

STEP 7: END.

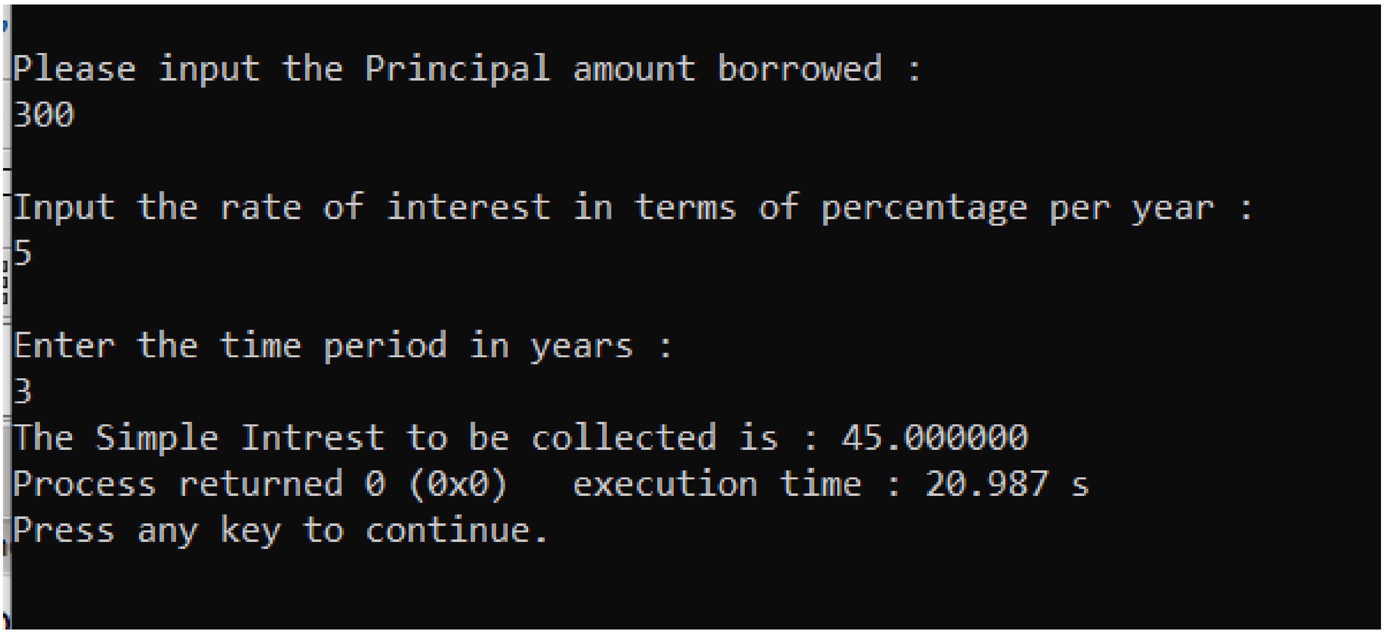
FLOWCHART:


Start
"Please input the Principal amount borrowed :"
GET principal
"Input the rate of intrest in terms of percentage per year :"
GET rate
"Enter the time period in years :"
GET time
SI ← (principal * rate * time) / 100
PUT "The Simple Intrest to be collected is :"+SI¶
End

PROGRAM:

```
 #include<stdio.h>

int main(){

  float si; //simple intrest

  float t;    // time

  float p;   //principal

  float r;   // rate

  printf("\nPlease input the Principal amount borrowed :\n");

  scanf("%f",&p);
  printf("\nInput the rate of interest in terms of percentage per year :");

  scanf("%f",&r);

  printf("\nEnter the time period in years : \n");

  scanf("%f",&t);

  si =((p*r*t)/100);

  printf("The Simple Intrest to be collected is : %f ",si)

  return 0;

}
```

OUTPUT:



```
Please input the Principal amount borrowed :
300

Input the rate of interest in terms of percentage per year :
5

Enter the time period in years :
3
The Simple Intrest to be collected is : 45.000000
Process returned 0 (0x0)   execution time : 20.987 s
Press any key to continue.
```

EXPERIMENT NO 4

BASIC MATH

4.1)

AIM: Write a C Program to convert Celsius to Fahrenheit and vice versa.

DESCRIPTION:

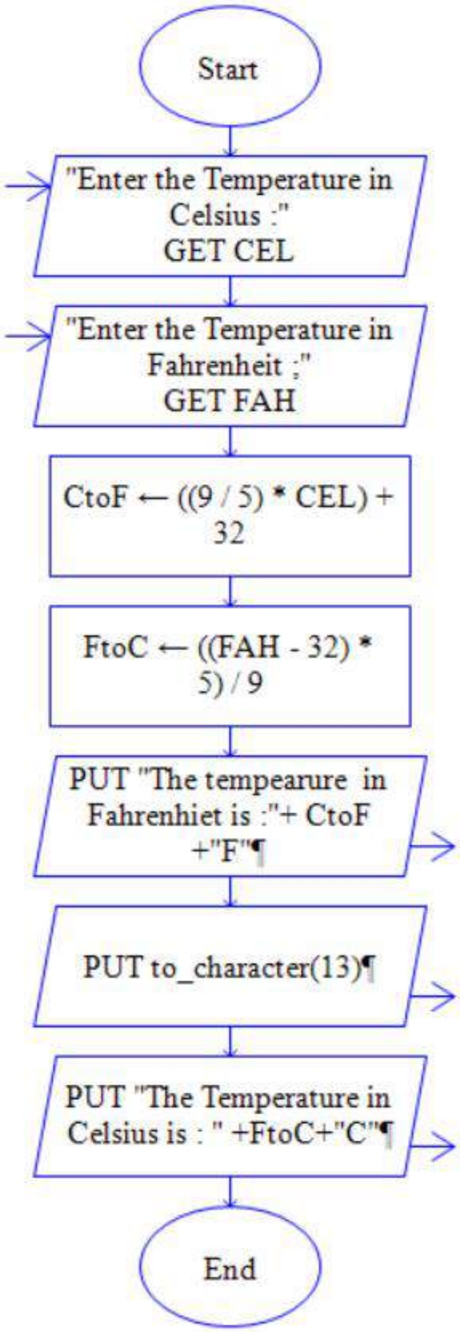
THE algorithm developed here coverts a temperature given in Celsius to Fahreheit and vice vera and both at the same time also.

Inputs are taken from the user and coverted are values get diaplayed.

ALGORITHM:

```
STEP 1: START.

STEP 2: CREATE FOUR VARIABLES CEL, FAH, CtoF, FtoC.

STEP 3: READ CEL.

STEP 4: READ FAH

STEP 5: SET CtoF = ((9/5)CEL)+32.

STEP 7: SET FtoC = ((FAH-32*5)/9.

STEP 8: PRINT VALUES OBTAINED.

STEP 9: END.
```

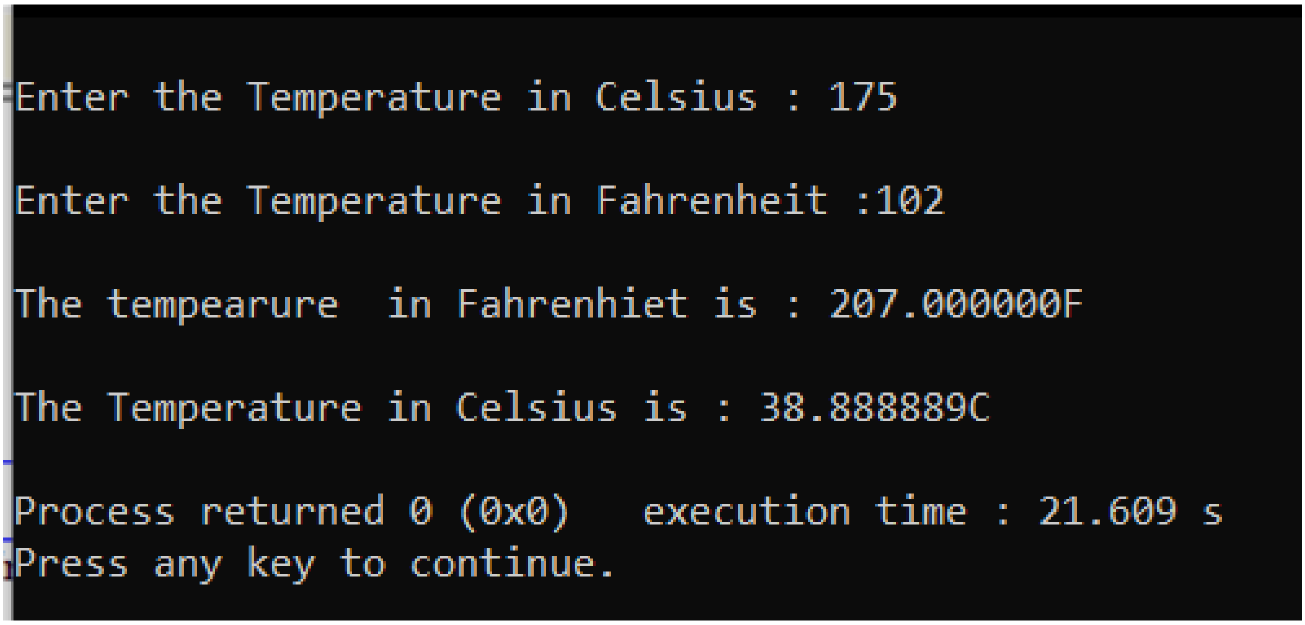
FLOWCHART:


Start
"Enter the Temperature in Celsius :"
GET CEL
"Enter the Temperature in Fahrenheit ;"
GET FAH
CtoF ← ((9 / 5) * CEL) + 32
FtoC ← ((FAH - 32) * 5) / 9
PUT "The tempearure in Fahrenhiet is :"+ CtoF +"F"¶
PUT to_character(13)¶
PUT "The Temperature in Celsius is : " +FtoC+"C"¶
End

PROGRAM:

```
#include<stdio.h>

int main(){

float CEL,FAH,CtoF,FtoC;

  printf("\nEnter the Temperature in Celsius : ");

  scanf("%f",&CEL);

  printf("\nEnter the Temperature in Fahrenheit :");
  scanf("%f",&FAH);

  CtoF =((9/5)*CEL)+32;

  FtoC =((FAH-32)*5)/9;

  printf("\nThe tempearure  in Fahrenhiet is : %fF\n \nThe Temperature in Celsius is : %fC\n",CtoF,FtoC);

return 0;

}
```

OUTPUT:



```
Enter the Temperature in Celsius : 175

Enter the Temperature in Fahrenheit :102

The tempearure  in Fahrenhiet is : 207.000000F

The Temperature in Celsius is : 38.888889C

Process returned 0 (0x0)   execution time : 21.609 s
Press any key to continue.
```

4.2)

AIM: Write a C Program to find largest of three numbers using ternary operator.

DESCRIPTION:

- The algorithm developed here ensures that three numeric values are taken as inputs and compared with each other to finally give an ouput for the maximum among them.

ALGORITHM:

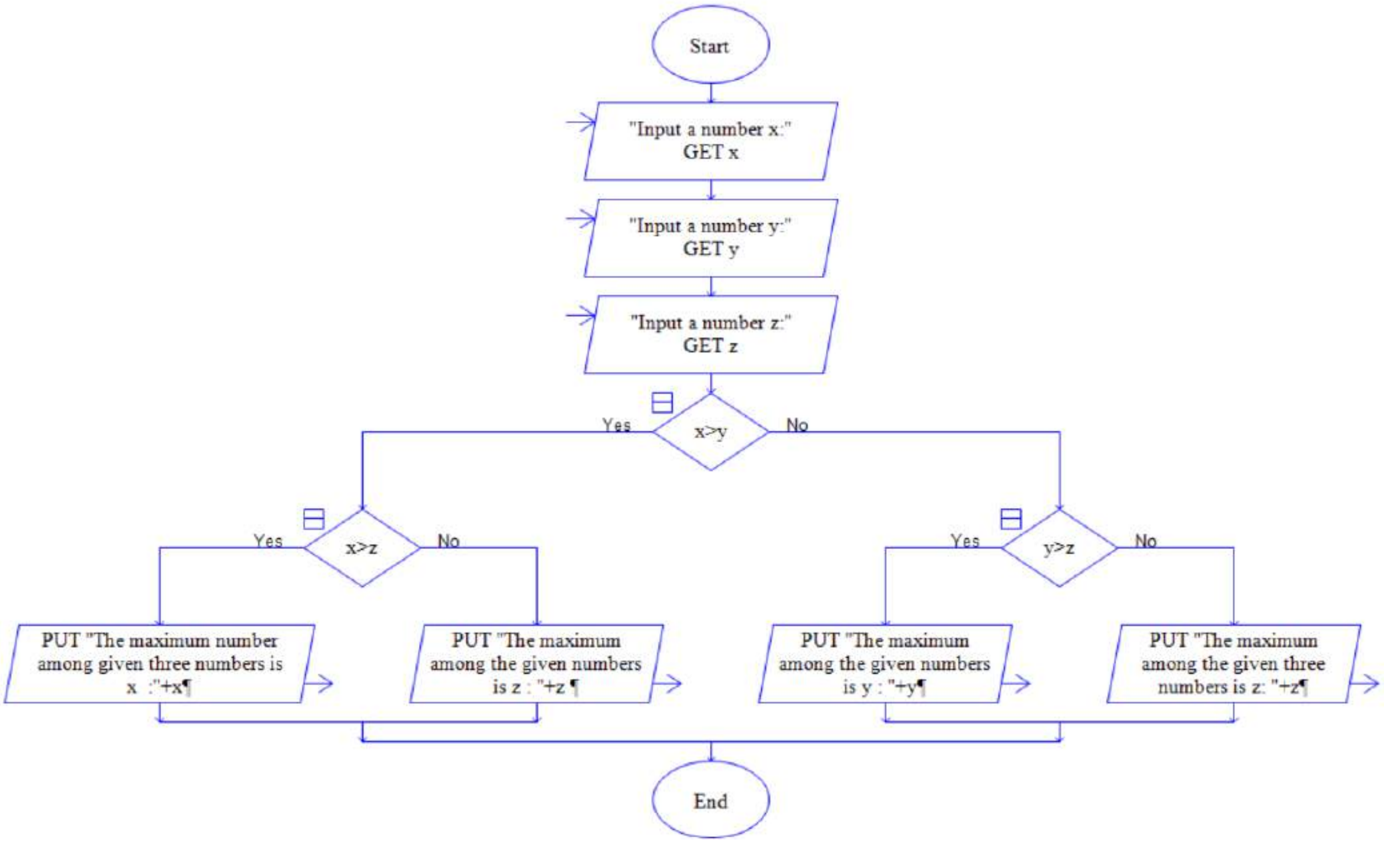
```
STEP 1: START.
STEP 2: CREATE THREE VARIABLES x, y, z.
STEP 3: READ x,y,z.
STEP 4: CHECK IF x>y
IF YES,
CHECK IF x>z
IF YES,
STEP 5: PRINT x AS MAX.
IF NO,
PRINT z AS MAX.
STEP 7: IF NO FROM STEP 4.
CHECK IF y>z.
IF YES,
PRINT y AS MAX
```

STEP 8: IF NO,

PRINT z AS MAX.

STEP 9: END.

FLOWCHART:

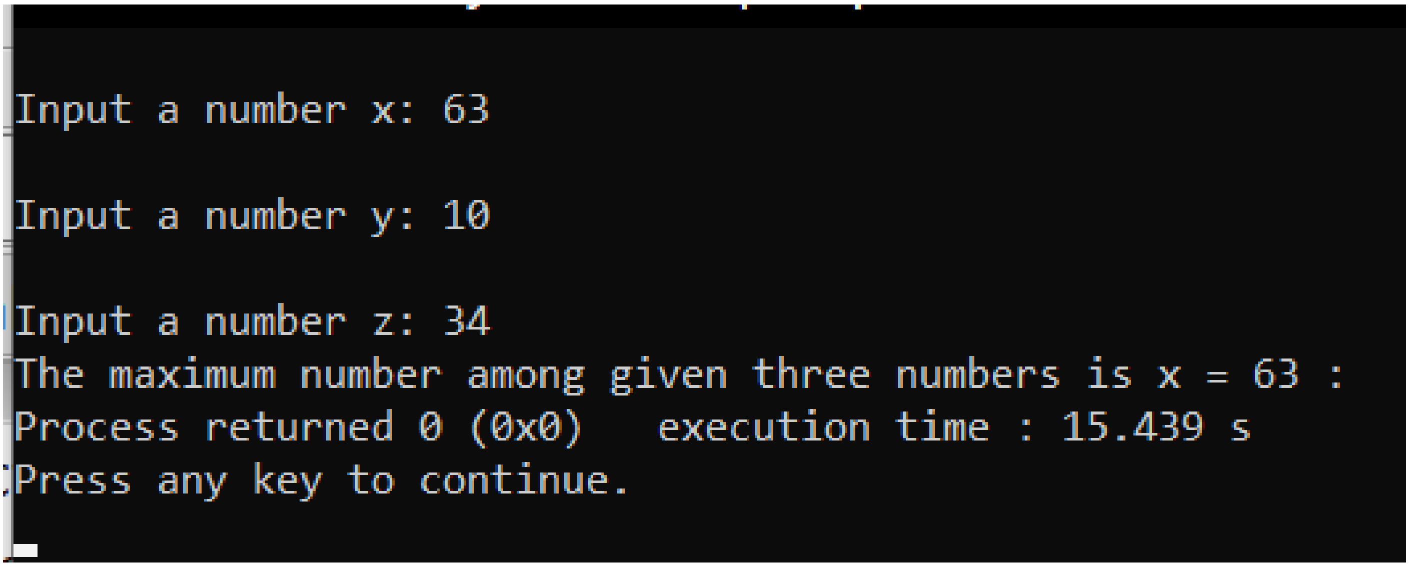

Start
"Input a number x:"
GET x
"Input a number y:"
GET y
"Input a number z:"
GET z
Yes
x>y
No
Yes
x>z
No
Yes
y>z
No
PUT "The maximum number among given three numbers is x :"+x¶
PUT "The maximum among the given numbers is z : "+z ¶
PUT "The maximum among the given numbers is y : "+y¶
PUT "The maximum among the given three numbers is z: "+z¶
End

PROGRAM:

```
#include<stdio.h>

int main(){

int x,y,z;

  printf("\nInput a number x: ");

  scanf("%d",&x);

  printf("\nInput a number y: ");

  scanf("%d",&y);
  printf("\nInput a number z: ");
  scanf("%d",&z);
  if (x>y)
  {
    if (x>z)
    {
      printf("The maximum number among given three numbers is x =
%d :",x );     }
    else
    {
      printf("The maximum among the given numbers is z = %d ",z );     }
  }
  else
  {
    if (y>z)
    {
      printf("The maximum among the given numbers is y = %d ",y);     }
    else
    {
      printf("The maximum among the given three numbers is z = %d
",z);     }
  }
```

```
return 0;

}
```

OUTPUT:



```
Input a number x: 63

Input a number y: 10

Input a number z: 34
The maximum number among given three numbers is x = 63 :
Process returned 0 (0x0)   execution time : 15.439 s
Press any key to continue.
```

4.3)

AIM: Write a C Program to Calculate area of a Triangle using Heron's formula

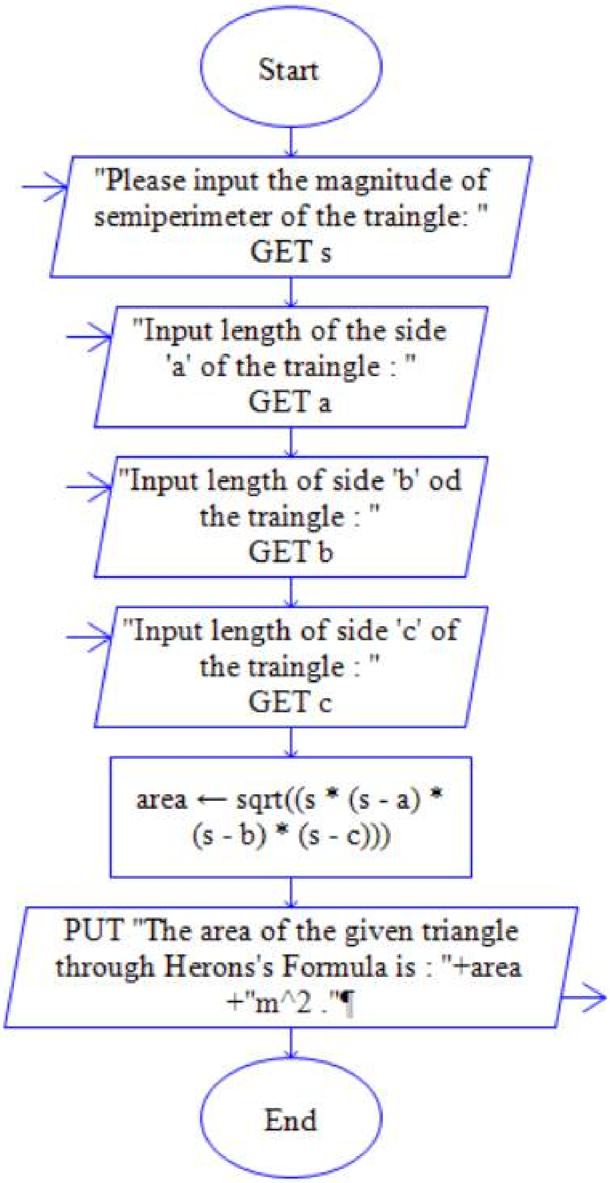
DESCRIPTION:

- The algorithm developed helps in calculating the area of any given triangle with its semi-perimeter, length of all the three sides of it.
- The formula for area = SquareRootOf(s*(s-a)*(s-b)*(s-c))

ALGORITHM:

```
STEP 1: START.
STEP 2: CREATE VARIABLES a, b, c, s.
STEP 3: READ s
STEP 4: READ a
STEP 5: READ b
STEP 6: READ c
STEP 7: area = Sqrt(s*(s-a)*(s-b)*(s-c)).
STEP 8: PRINT area.
STEP 9: END.
```

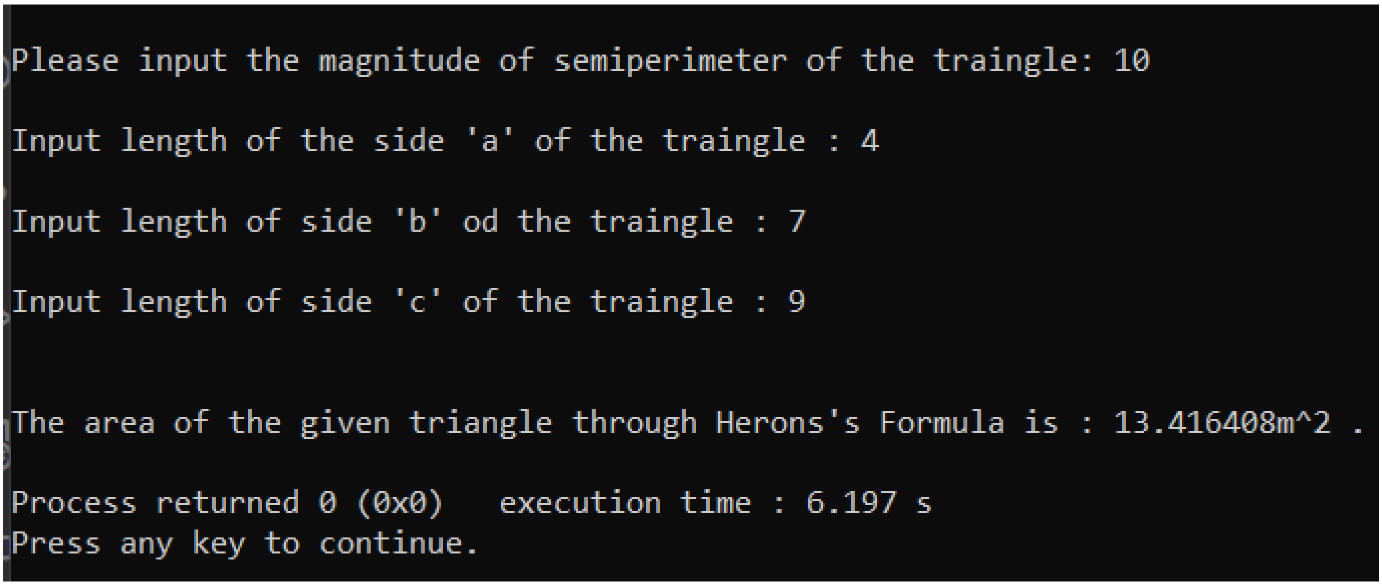
FLOWCHART:


Start
"Please input the magnitude of semiperimeter of the traingle: "
GET s
"Input length of the side 'a' of the traingle : "
GET a
"Input length of side 'b' od the traingle : "
GET b
"Input length of side 'c' of the traingle : "
GET c
area ← sqrt((s * (s - a) * (s - b) * (s - c)))
PUT "The area of the given triangle through Herons's Formula is : "+area +"m^2 ."¶
End

PROGRAM:

```
#include<stdio.h>
int main(){
int s,a,b,c;
float area;
  printf("\nPlease input the magnitude of semiperimeter of the traingle: ");
  scanf("%d",&s);
  printf("\nInput length of the side 'a' of the traingle : ");
  scanf("%d",&a);
  printf("\nInput length of side 'b' od the traingle : ");
   scanf("%d",&b);
  printf("\nInput length of side 'c' of the traingle : ");
  scanf("%d",&c);
  area =sqrt((s*(s-a)*(s-b)*(s-c)));
  printf("\n\nThe area of the given triangle through Herons's Formula is : %fm^2 .\n",area);
return 0;
}
```

OUTPUT:



```
Please input the magnitude of semiperimeter of the traingle: 10

Input length of the side 'a' of the traingle : 4

Input length of side 'b' od the traingle : 7

Input length of side 'c' of the traingle : 9


The area of the given triangle through Herons's Formula is : 13.416408m^2 .

Process returned 0 (0x0)   execution time : 6.197 s
Press any key to continue.
```

EXPERIMENT NO 5

CONTRO FLOW – I.

5.1)

AIM: Write a C Program to Find Whether the Given Year is a Leap Year or not.

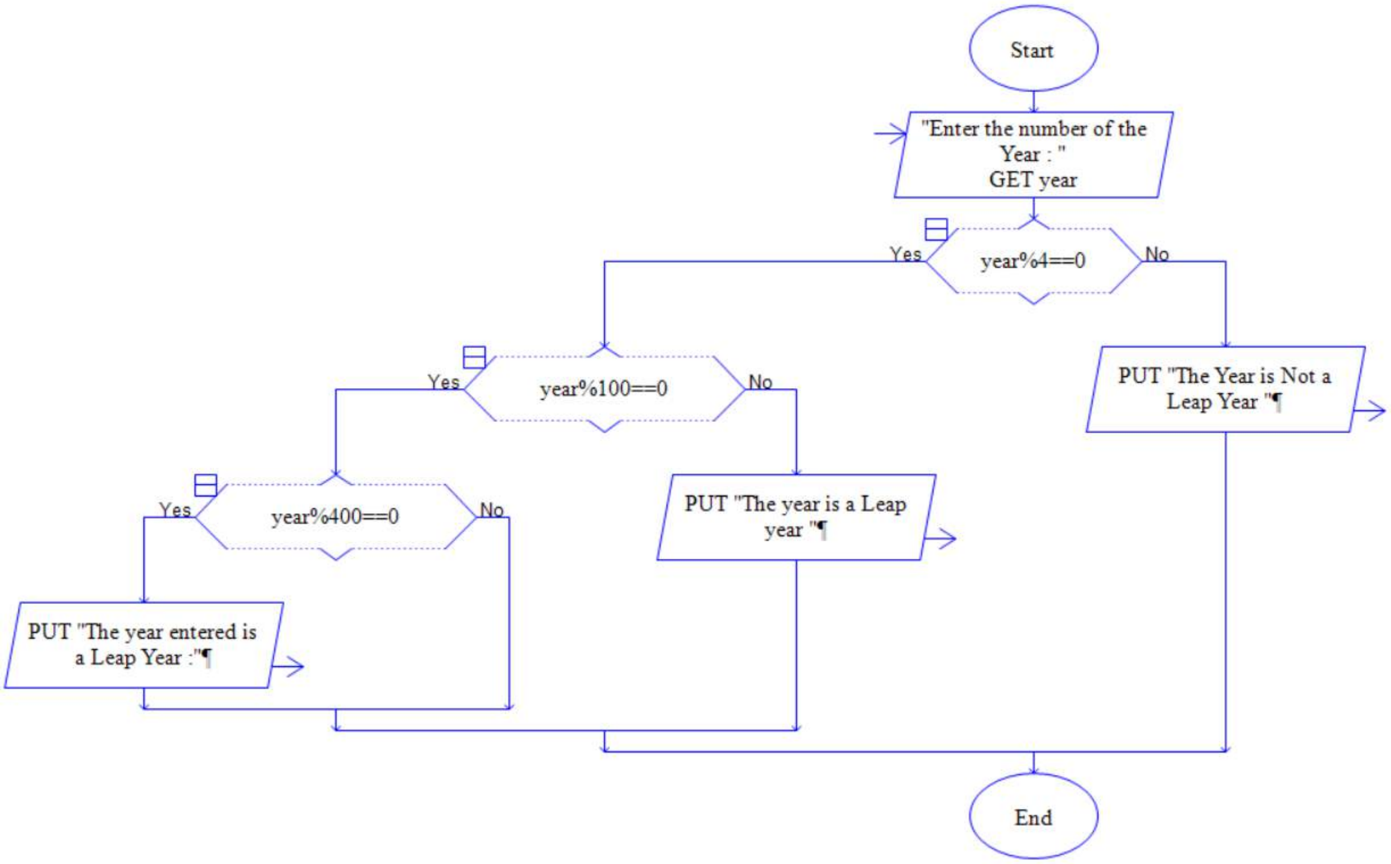
DESCRIPTION:

- The aim of the program helps in development of an algorithm, helping us to find out whether a year is a leap year or not.

ALGORITHM:

```
STEP 1: START.
STEP 2: CREATE A VARIABLE year.
STEP 3: READ year.
STEP 4: If the year is evenly divisible by 4, go to step 8. Otherwise, go to step 8.
STEP 5: If the year is evenly divisible by 100, go to step 3. Otherwise, go to step 7.
STEP 7: If the year is evenly divisible by 400, go to step 4. Otherwise, go to step 8.
STEP 8: The year is a leap year (it has 366 days).
STEP 9: The year is not a leap year (it has 365 days).
STEP 10: END.
```

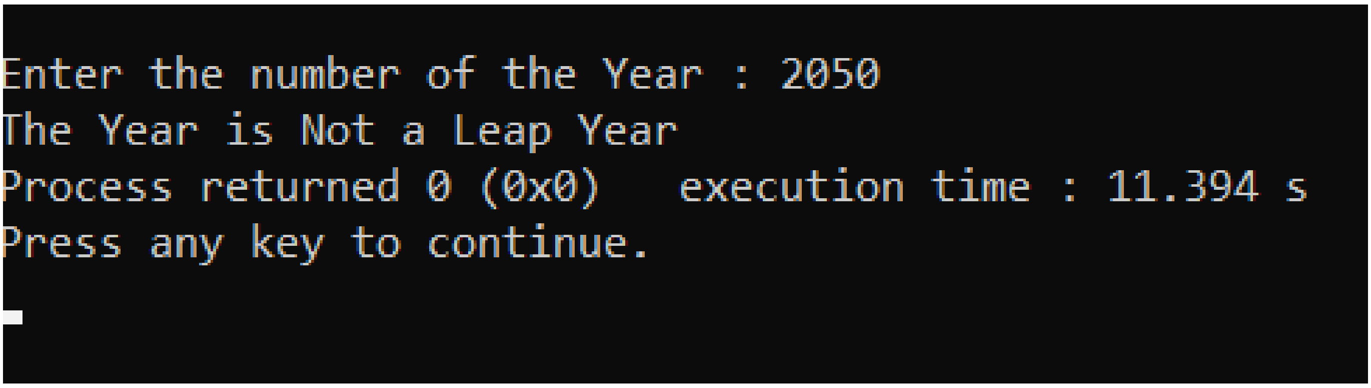
FLOWCHART:


Start
"Enter the number of the Year : "
GET year
Yes
year%4==0
No
Yes
year%100==0
No
PUT "The Year is Not a Leap Year "¶
Yes
year%400==0
No
PUT "The year is a Leap year "¶
PUT "The year entered is a Leap Year :"¶
End

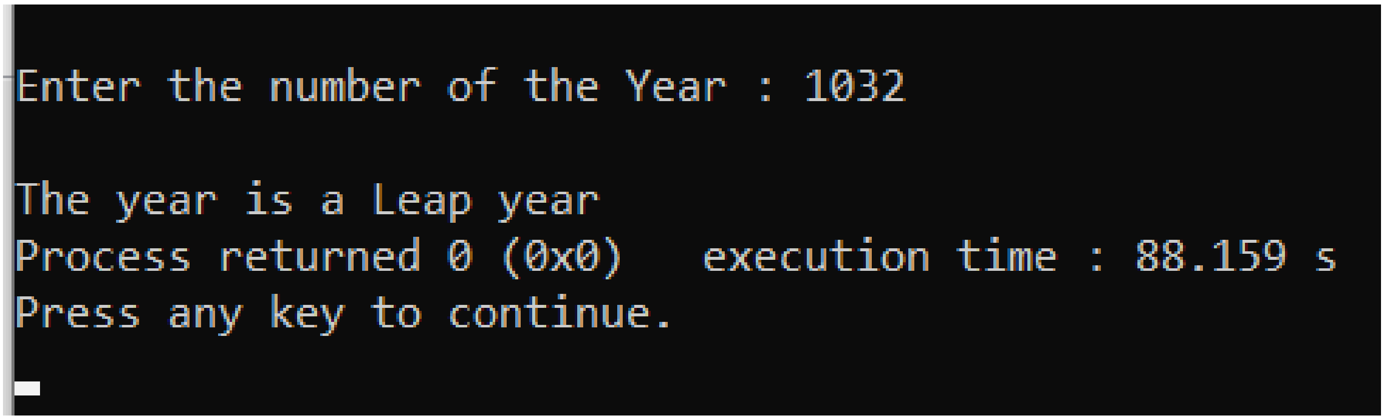
PROGRAM:

```
#include<stdio.h>
int main(){
int year;
printf("\nEnter the number of the Year : ");
scanf("%d",&year);
  if (year % 4==0)
  {
    if (year % 100==0)
    {
      if (year % 400==0)
      {
        printf("\nThe year entered is a Leap Year :");       }
      else
      {
      }
    }
    else
    {
      printf("\nThe year is a Leap year " );    }
  }
  else
  {
    printf("The Year is Not a Leap Year ");  }
return 0;
}
```

OUTPUT:



```
Enter the number of the Year : 2050
The Year is Not a Leap Year
Process returned 0 (0x0)   execution time : 11.394 s
Press any key to continue.
```



```

Enter the number of the Year : 1032

The year is a Leap year
Process returned 0 (0x0)   execution time : 88.159 s
Press any key to continue.
```

5.2)

AIM: Write a C program to find the roots of a Quadratic Equation.

DESCRIPTION:

- The algo developed here takes inputs for the coefficients of x^2 , x and constant values from the model equation a x^2+bx+c.
- Internal operations are performed to show the roots of the equation.
- The nature of the roots is also shown.

ALGORITHM:

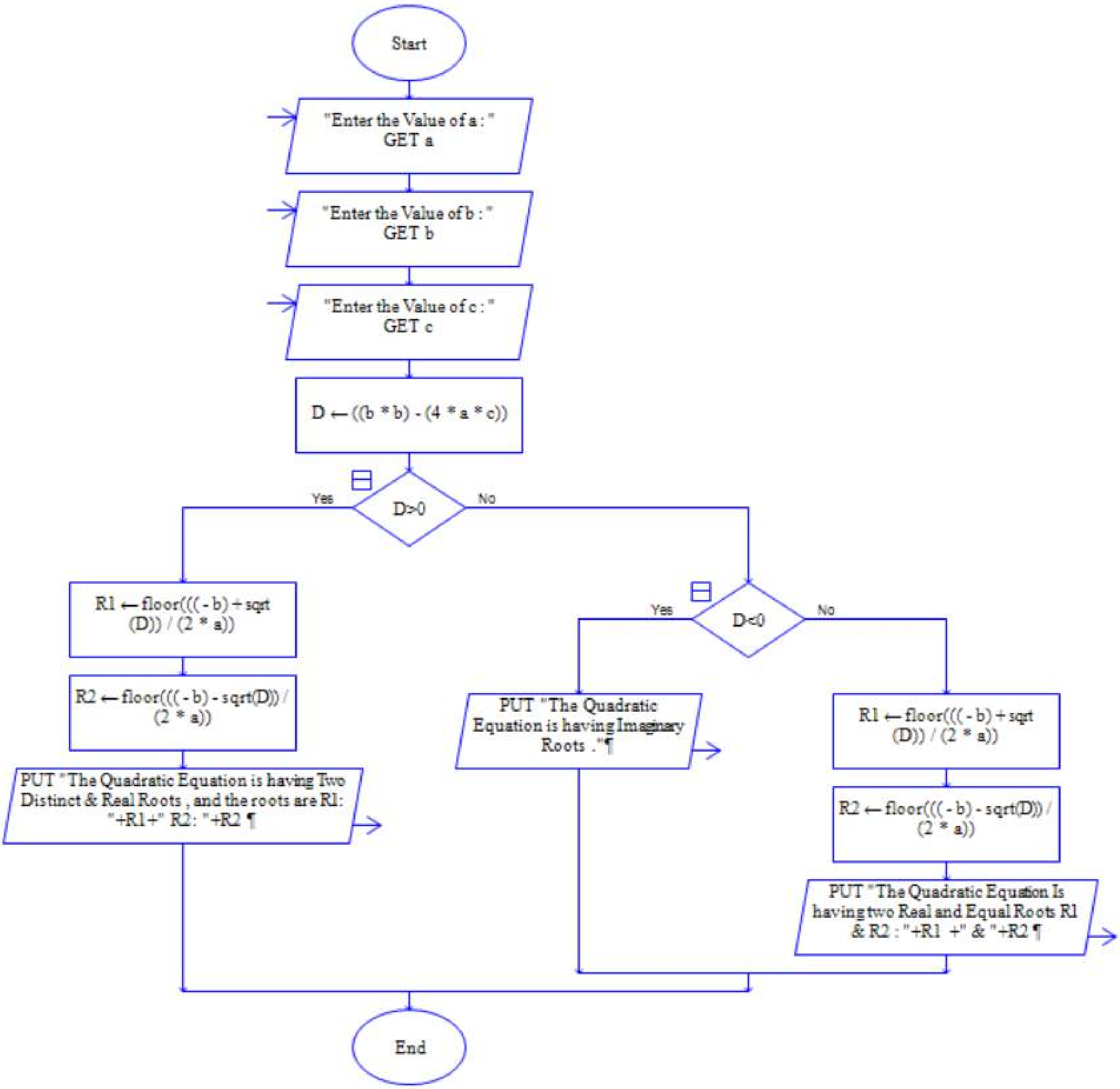
```
STEP 1: START.

STEP 2: CREATE VARIABLES a,b,c,R1,R2,D.

STEP 3: READ a.

STEP 4: READ b.

STEP 5: READ c.

STEP 7: CALC D = (b*b)-(4*a*c)

STEP 8: IF D>0 ,ASSIGN: R1 = ((-b)+sqrt(D))/(2*a).

                        R2 = ((-b)-sqrt(D))/(2*a).

DISPLAY ROOTS ARE REAL AND DISTINCT ALONG WITH R1 AND R2.

STEP 9: IF NO,

CHECK IF D<0.

IF YES,

STEP 10: PRINT ROOTS ARE IMAGINARY
```

STEP 11: IF NO,

DISPLAY ROOTS ARE REAL AND EQUAL ALONG WITH R1 AND R2.

STEP12: END.

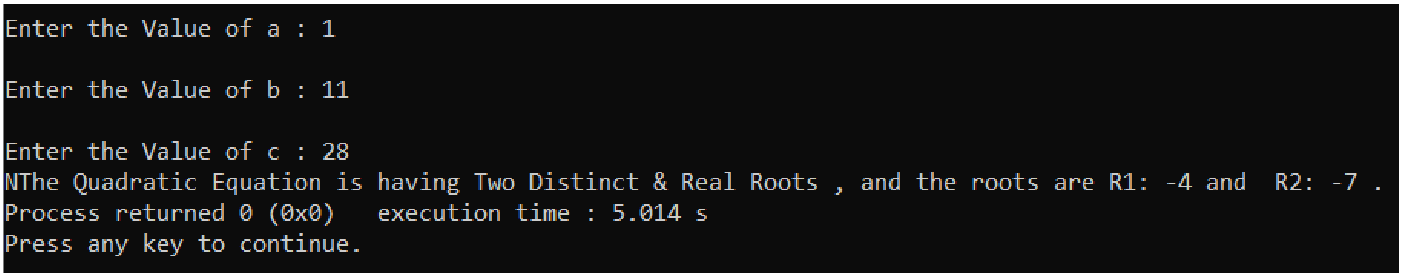
FLOWCHART:


Start
"Enter the Value of a : "
GET a
"Enter the Value of b : "
GET b
"Enter the Value of c : "
GET c
D ← ((b * b) - (4 * a * c))
Yes
D>0
No
R1 ← floor(((- b) + sqrt (D)) / (2 * a))
R2 ← floor(((- b) - sqrt(D)) / (2 * a))
PUT "The Quadratic Equation is having Two Distinct & Real Roots , and the roots are R1: "+R1+" R2: "+R2 ¶
Yes
D<0
No
PUT "The Quadratic Equation is having Imaginary Roots ."¶
R1 ← floor(((- b) + sqrt (D)) / (2 * a))
R2 ← floor(((- b) - sqrt(D)) / (2 * a))
PUT "The Quadratic Equation Is having two Real and Equal Roots R1 & R2 : "+R1 +" & "+R2 ¶
End

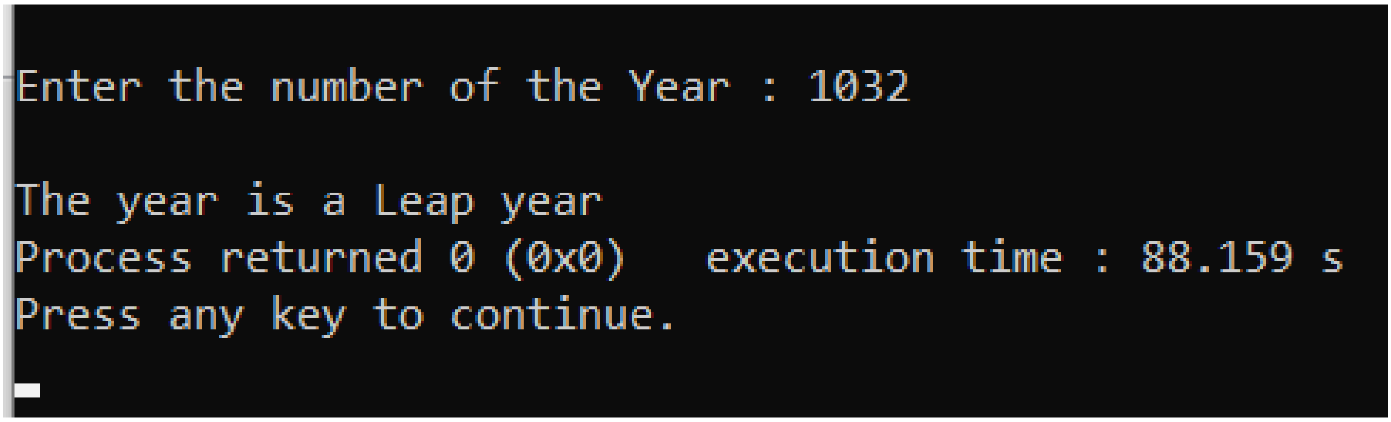
PROGRAM:

```
#include<stdio.h>

int main(){
 int a,b,c,R1,R2,D;
  printf("\nEnter the Value of a : ");
  scanf("%d",&a);
  printf("\nEnter the Value of b : ");
  scanf("%d",&b);
  printf("\nEnter the Value of c : ")
  scanf("%d",&c);
  D =((b*b)-(4*a*c));
  if (D>0)
  {
    R1 =floor(((-b)+sqrt(D))/(2*a));
    R2 =floor(((-b)-sqrt(D))/(2*a));
    printf("\NThe Quadratic Equation is having Two Distinct & Real
Roots , and the roots are R1: %d and  R2: %d .",R1,R2);
     }
  else
  {
    if (D<0)
    {
      printf("\nThe Quadratic Equation is having Imaginary Roots .");
     }
    else
    {
      R1 =floor(((-b)+sqrt(D))/(2*a));
      R2 =floor(((-b)-sqrt(D))/(2*a));
      printf("\nThe Quadratic Equation Is having two Real and Equal Roots R1 = %d & R2 = %d \n",R1,R2);
     }
  }
return 0;
}
```

OUTPUT:



```
Enter the Value of a : 1

Enter the Value of b : 11

Enter the Value of c : 28
NThe Quadratic Equation is having Two Distinct & Real Roots , and the roots are R1: -4 and  R2: -7 .
Process returned 0 (0x0)   execution time : 5.014 s
Press any key to continue.
```



```
Enter the number of the Year : 1032

The year is a Leap year
Process returned 0 (0x0)   execution time : 88.159 s
Press any key to continue.
```

5.3)

AIM: Write a C Program to make a simple Calculator to Add, Subtract, Multiply or Divide Using Switch...case.

DESCRIPTION:

- The algo takes input for two numeric values and another input for the operation to be performed.
- Internal calculations are performed and the desire operation with its result is displayed to the user.
- The code is developed with the help of a switch case statement.

ALGORITHM:

STEP 1: START.

STEP 2: CREATE VARIABLES a, b OF INT TYPE, value OF FLOAT TYPE, c OF CHAR TYPE.

STEP 3: READ c, a, b.

STEP 4: USING SWICTCH CASE STATEMENT COMPARE VALUE GIVEN BY USER AGAINST THE CASE VALUES AS SHOWN IN THE FLOW CHART OR PROGRAM.

STEP 5: DECLARE CASES +, -, * , /, %.

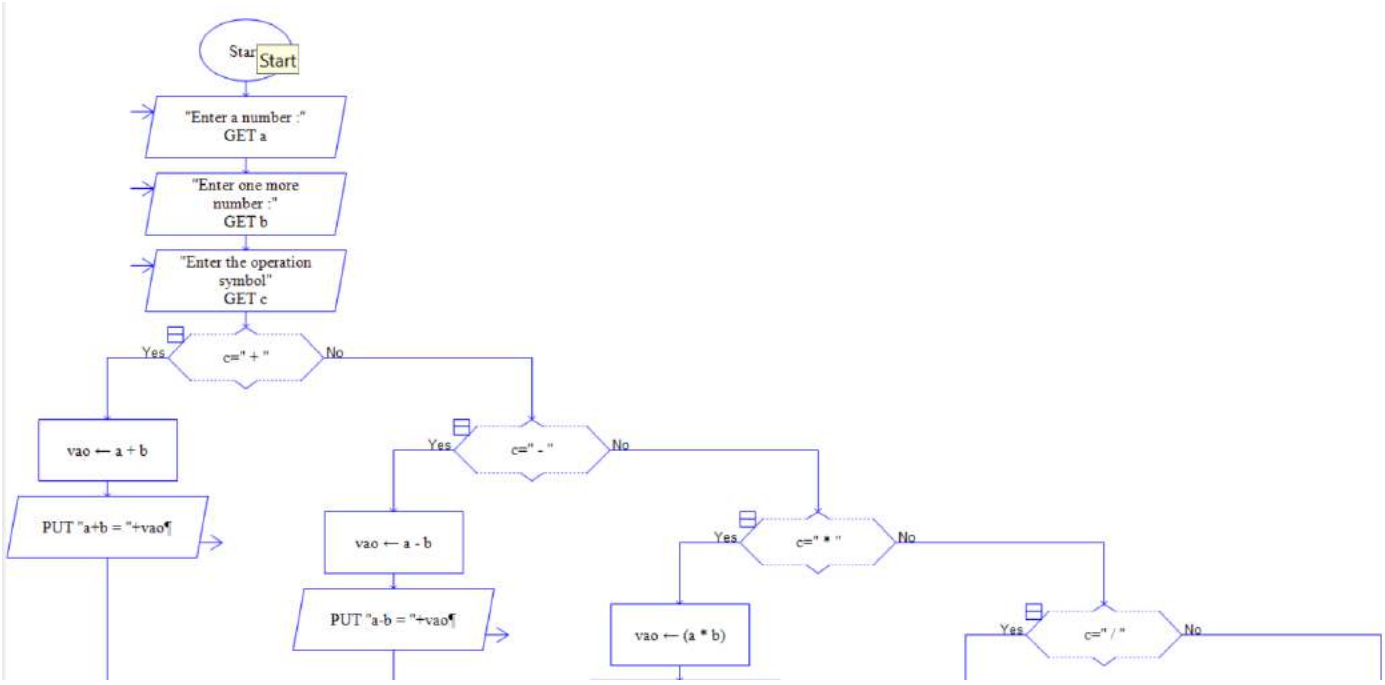
ALSO INCLUDE STATEMENTS FOR A default STATEMENT.

STEP 7: SPECIFY OPERATIONS TO BE PERFORMED UNDER EACH CASE ALONG WITH A break STATEMENT.

STEP 8: DISPLAY VALUE

STEP 9: END.

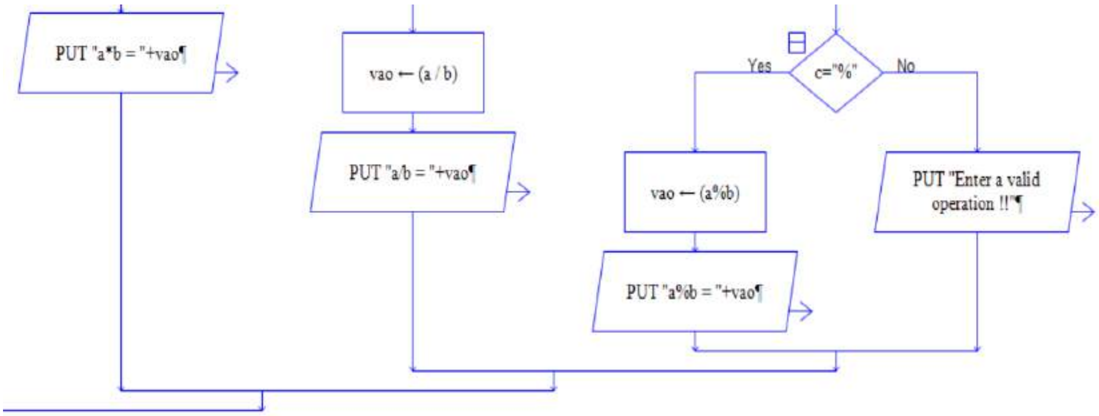
FLOWCHART:



1 2 3 4

5

3 4 5



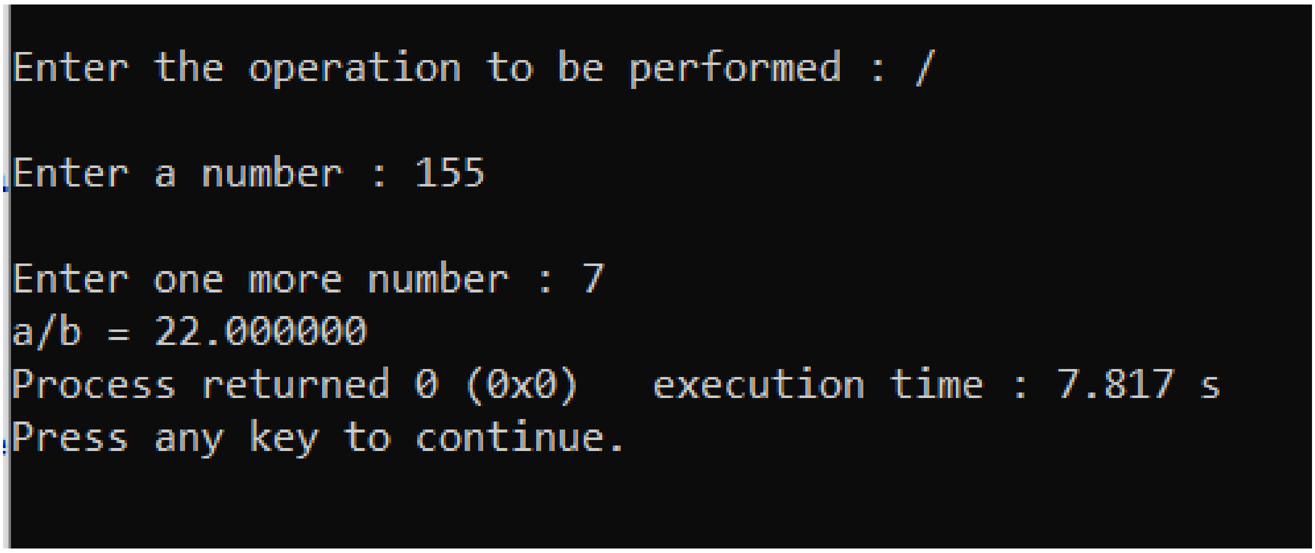
PROGRAM:

```
#include<stdio.h>

int main(){
```

```
int a,b;

float value;

char c;

  printf("\nEnter the operation to be performed : ");

  scanf("%c",&c);

  printf("\nEnter a number : ");

  scanf("%d",&a);

  printf("\nEnter one more number : ");

  scanf("%d",&b);

switch(c)

  {

  case '+':

          value=(a+b);

          printf("a+b = %f",value);

          break;

  case '-':

           value=(a-b);

           printf("a-b = %f",value);

           break;

  case '*':

           value=(a*b);

           printf("a*b = %f",value);
```

```
            break;
    case '/':
            value=(a/b);
            printf("a/b = %f",value);
            break;
    case '%':
           value=(a%b);
           printf("a%cb = %f",c,value);
           break;
    default :
           printf("Enter a valid operation !!.");
           break;
    }
return 0;
}
```

OUTPUT:



```
Enter the operation to be performed : /

Enter a number : 155

Enter one more number : 7
a/b = 22.000000
Process returned 0 (0x0)   execution time : 7.817 s
Press any key to continue.
```

EXPERIMENT NO 6

CONTROL FLOW - II

6.1)

AIM: Write a C Program to Find Whether the Given Number is Prime number or not.

DESCRIPTION:

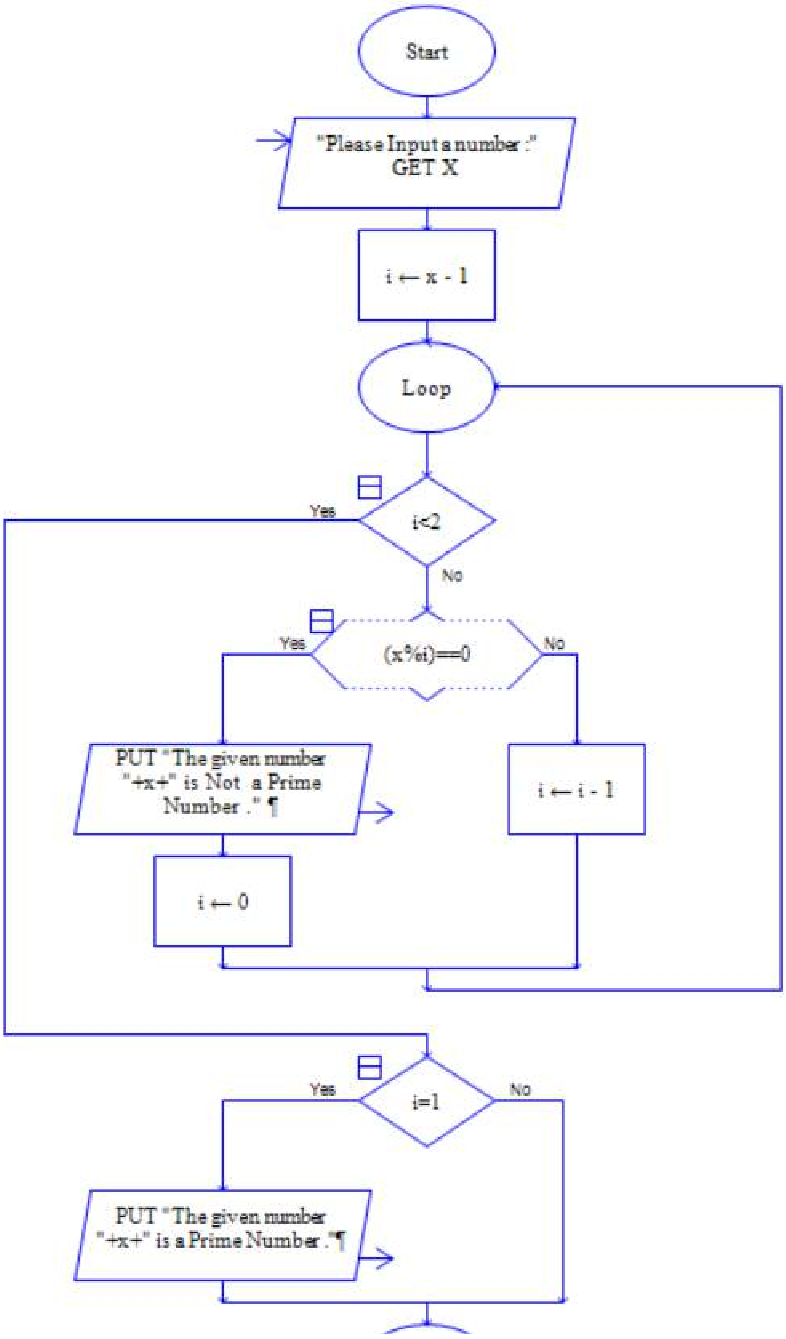
- **The** algo developed here takes input into variable declared int it.
- Internal operations are performed to check whether the given is prime or not.

ALGORITHM:

```
STEP 1: START.
STEP 2: CREATE VARIABLES i,x.
STEP 3: READ x.
STEP 4: SET i = x-1.
STEP 5: WHILE !(i<2)
IF CONITION IS SATISFIED.
CHECK if ((x % i)==0)
IF YES,
DISPLAY NOT PRIME.
SET i=0.
STEP 7: IF NO,
SET i=i-1.
STEP 8: IF CONDITION IS NOT SATISFIED.
STEP 9: DISPLAY PRIME.
```

STEP 10: END.

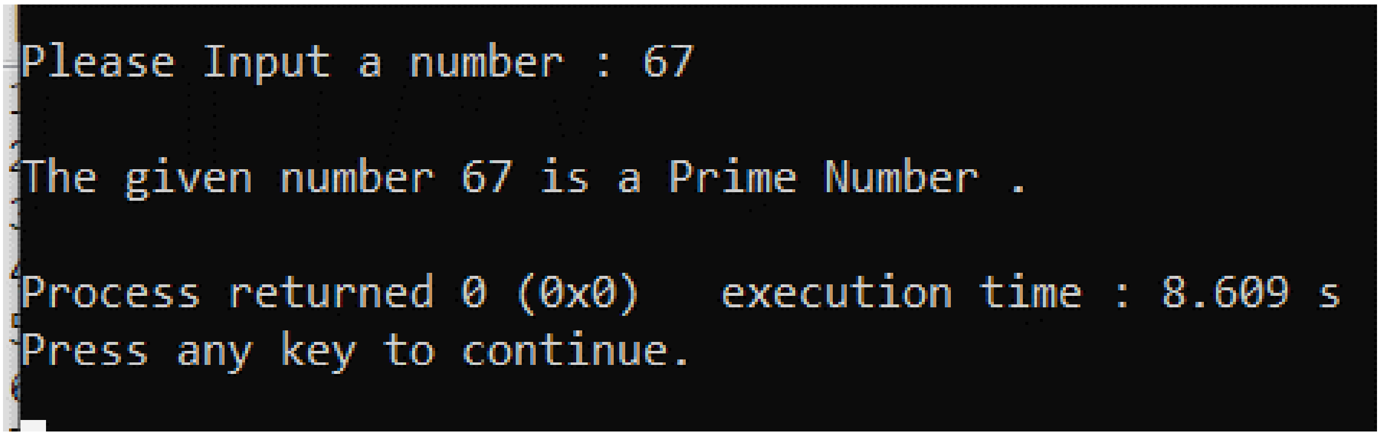
FLOWCHART:


Start
"Please Input a number :"
GET X
i ← x - 1
Loop
i<2
Yes
No
(x%i)==0
Yes
No
PUT "The given number "+x+" is Not a Prime Number ." ¶
i ← 0
i ← i - 1
i=1
Yes
No
PUT "The given number "+x+" is a Prime Number ."¶

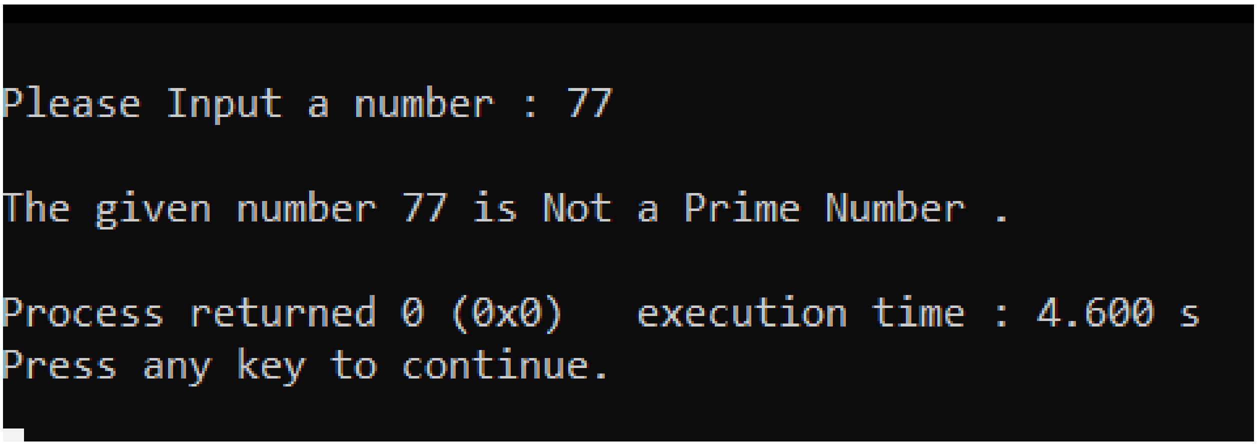
PROGRAM:

```
#include<stdio.h>

int main(){

  int i,x;

  printf("\nPlease Input a number : ");
  scanf("%d",&x)

  i =x-1;

  while (! (i<2))

  {

    if ((x % i)==0)

    {

      printf("\nThe given number %d is Not a Prime Number .\n",x);

      i=0;

    }

    else{

      i =i-1;

    }

  }

  if (i==1){

    printf("\nThe given number %d is a Prime Number .\n",x);


return 0;

}
```

OUTPUT:



```
Please Input a number : 67

The given number 67 is a Prime Number .

Process returned 0 (0x0)   execution time : 8.609 s
Press any key to continue.
```



```
Please Input a number : 77

The given number 77 is Not a Prime Number .

Process returned 0 (0x0)   execution time : 4.600 s
Press any key to continue.
```

6.2)

AIM: Write a C Program to Find Whether the Given Number is Armstrong Number or not.

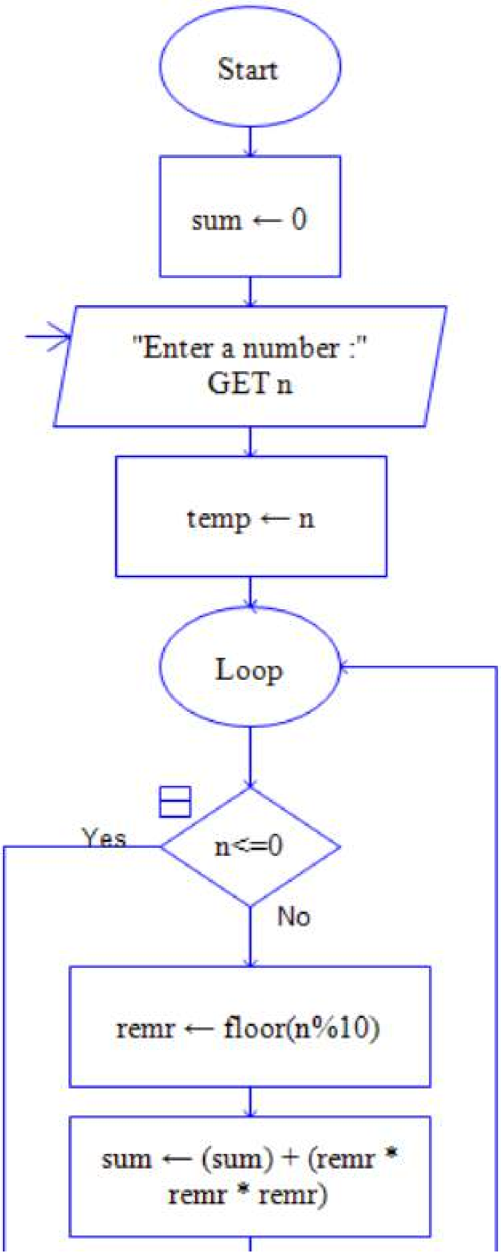
DESCRIPTION:

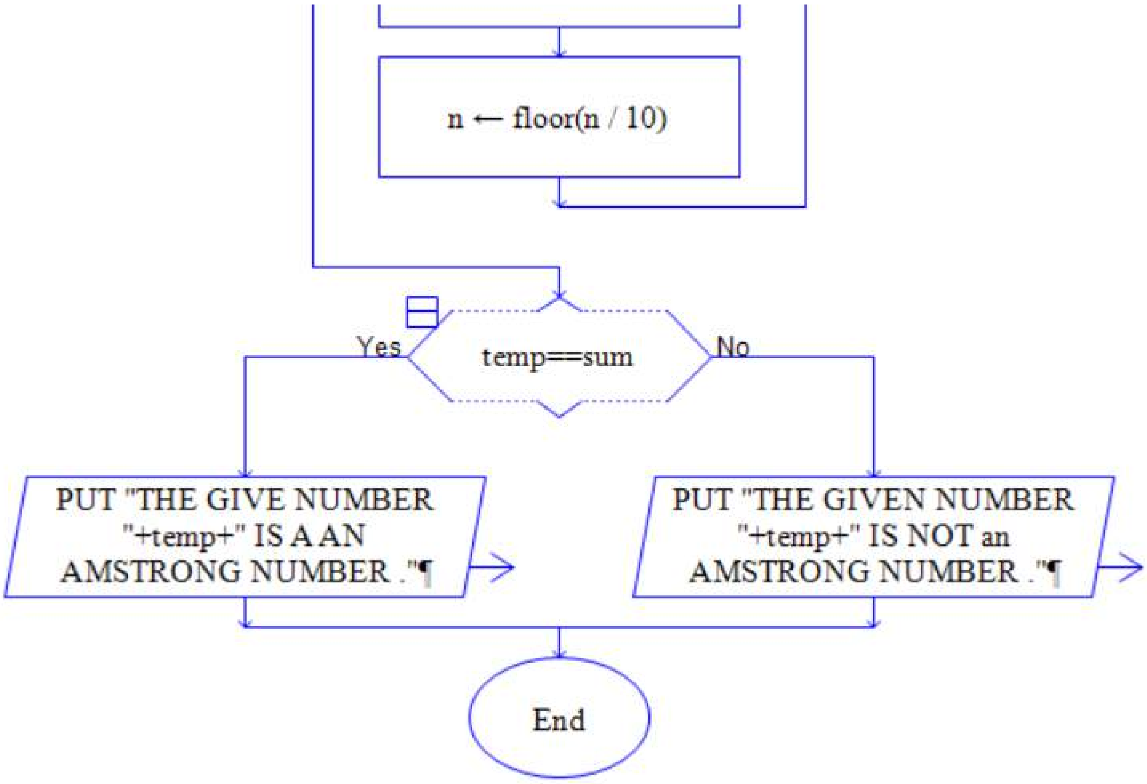
- The program takes input from the user a number.
- Internally given logic evaluates and shows the output as according to the logic of a Armstrong number.

ALGORITHM:

```
STEP 1: START.
STEP 2: CREATE VARIABLES r, t, s, n.
STEP 3: SET sum =0.
STEP 4: READ n.
STEP 5: SET t = n.
STEP 7: while (!(n<=0))
SET r=(n % 10);
    s=((s)+(r*r*r));
    n =(n/10);
STEP 8: if (temp==sum)
DISPLAY AMSTRONG
STEP 9: If NO,
DISPLAY not AMSTRONG.
STEP 10: END.
```

FLOWCHART:

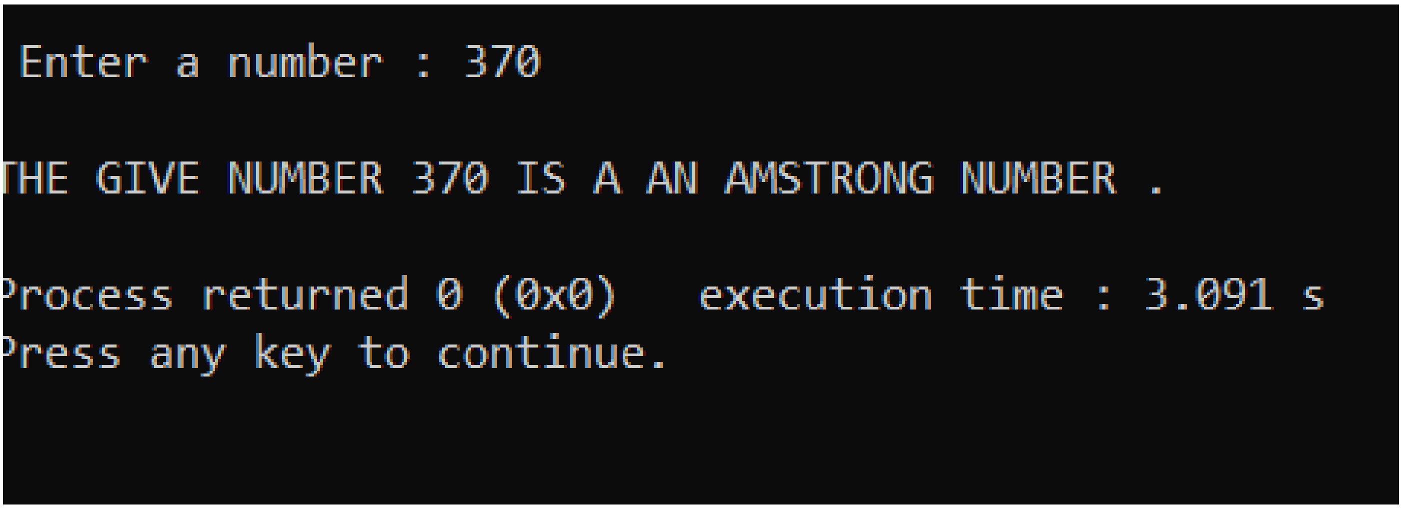

Start
sum ← 0
"Enter a number :"
GET n
temp ← n
Loop
n<=0
Yes
No
remr ← floor(n%10)
sum ← (sum) + (remr *
remr * remr)


n ← floor(n / 10)
Yes
temp==sum
No
PUT "THE GIVE NUMBER "+temp+" IS A AN AMSTRONG NUMBER ."¶
PUT "THE GIVEN NUMBER "+temp+" IS NOT an AMSTRONG NUMBER ."¶
End

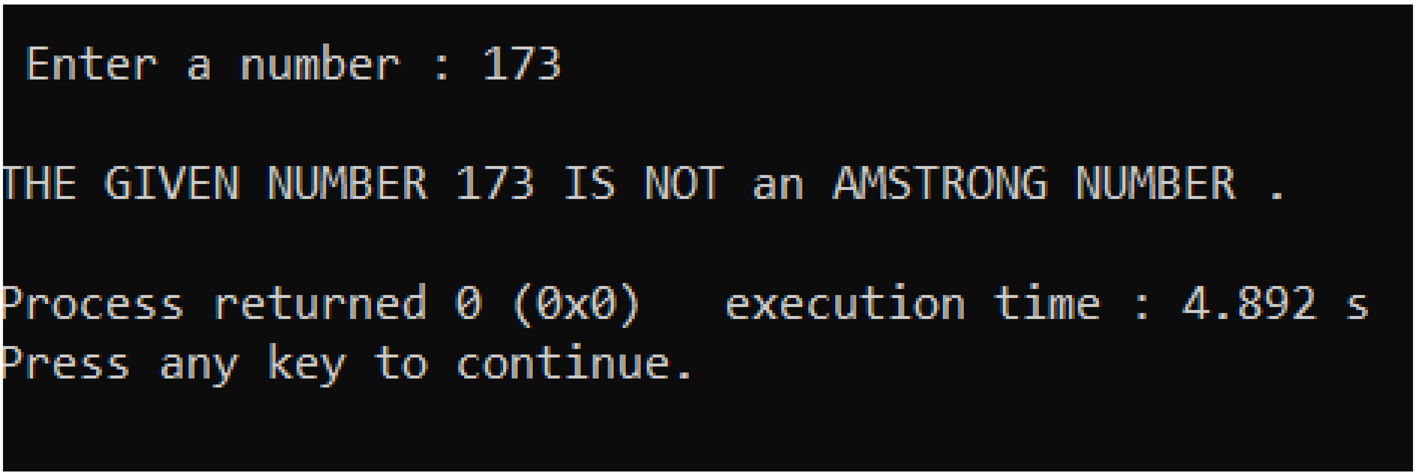
PROGRAM:

```
#include <stdio.h>

#include <string.h>

int main()

{

  int remr;

  int temp;

  int sum;

  int n

  sum =0;


  printf("\n Enter a number : ");

  scanf("%d",&n);
  temp =n;
  while (!(n<=0))
  {
    remr =(n % 10);
    sum =((sum)+(remr*remr*remr))
    n =floor(n/10);
  }
  if (temp==sum)
  {
    printf("\nTHE GIVE NUMBER %d IS A AN AMSTRONG NUMBER
.\n",temp);
  }
  else
  {
    printf("\nTHE GIVEN NUMBER %d IS NOT an AMSTRONG NUMBER
.\n",temp);
   }
  return 0;
}
```

OUTPUT:



```
 Enter a number : 370

THE GIVE NUMBER 370 IS A AN AMSTRONG NUMBER .

Process returned 0 (0x0)   execution time : 3.091 s
Press any key to continue.
```



```
 Enter a number : 173

THE GIVEN NUMBER 173 IS NOT an AMSTRONG NUMBER .

Process returned 0 (0x0)   execution time : 4.892 s
Press any key to continue.
```

6.3)

AIM: Write a C program to print Floyd Triangle.

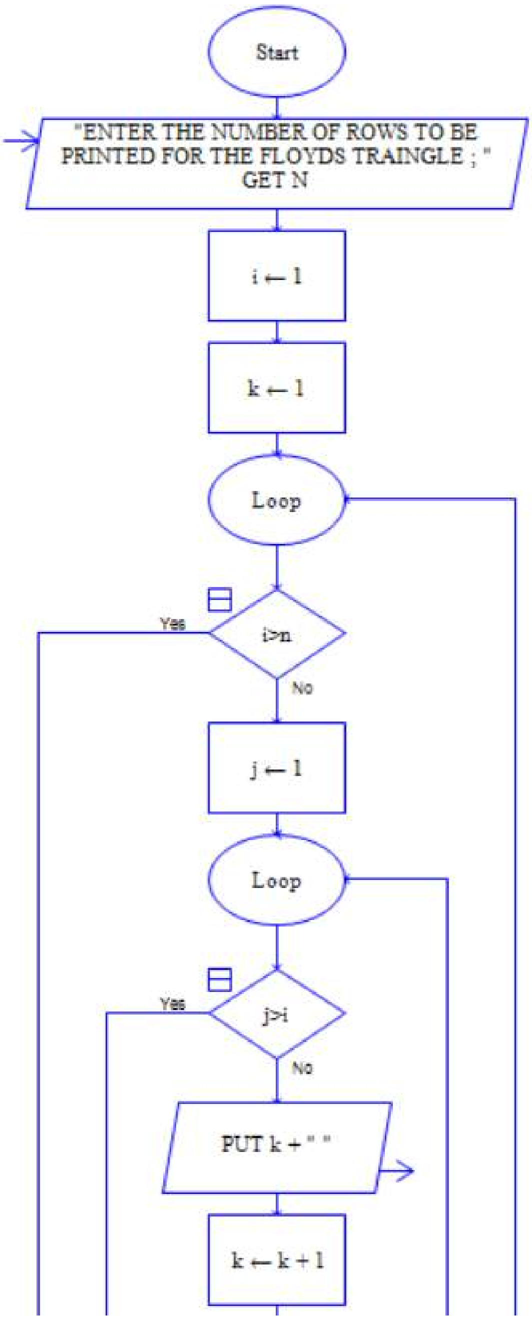
DESCRIPTION:

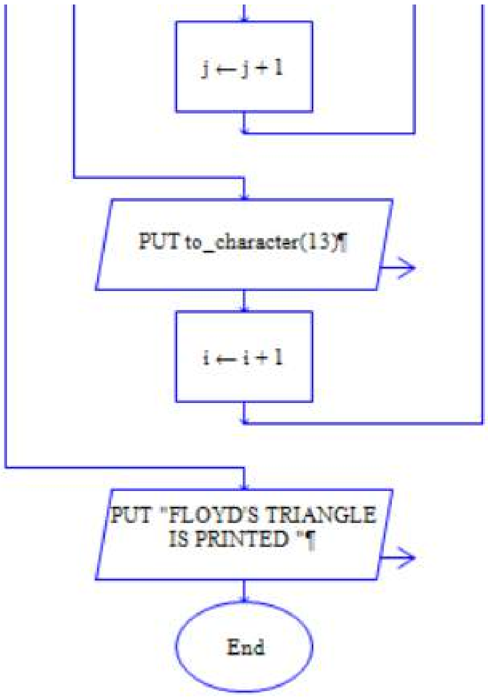
- **The** algorithm developed here prints a Floyd’s Triangle by taking input for the number of wanted by user / taken input from the user.

ALGORITHM:

```
STEP 1: START.

STEP 2: CREATE VARIABLE j, k, I, n.

STEP 3: READ n.

STEP 4: SET i=1.
             SET k=1.

STEP 5: while (!(i>n))

SET j=1.


STEP 7: while (!(j>i))

DISPLAY k,

SET k=k+1.

SET j=j+1.

STEP 8: IF CONDITION IN STEP 7 IS FALSE.

SET i= i+1.

STEP 9: DISPLAY THE TRANGLE

STEP 10:  END.
```

FLOWCHART:


Start
"ENTER THE NUMBER OF ROWS TO BE PRINTED FOR THE FLOYDS TRAINGLE ; " GET N
i ← 1
k ← 1
Loop
Yes
i>n
No
j ← 1
Loop
Yes
j>i
No
PUT k + " "
k ← k + 1

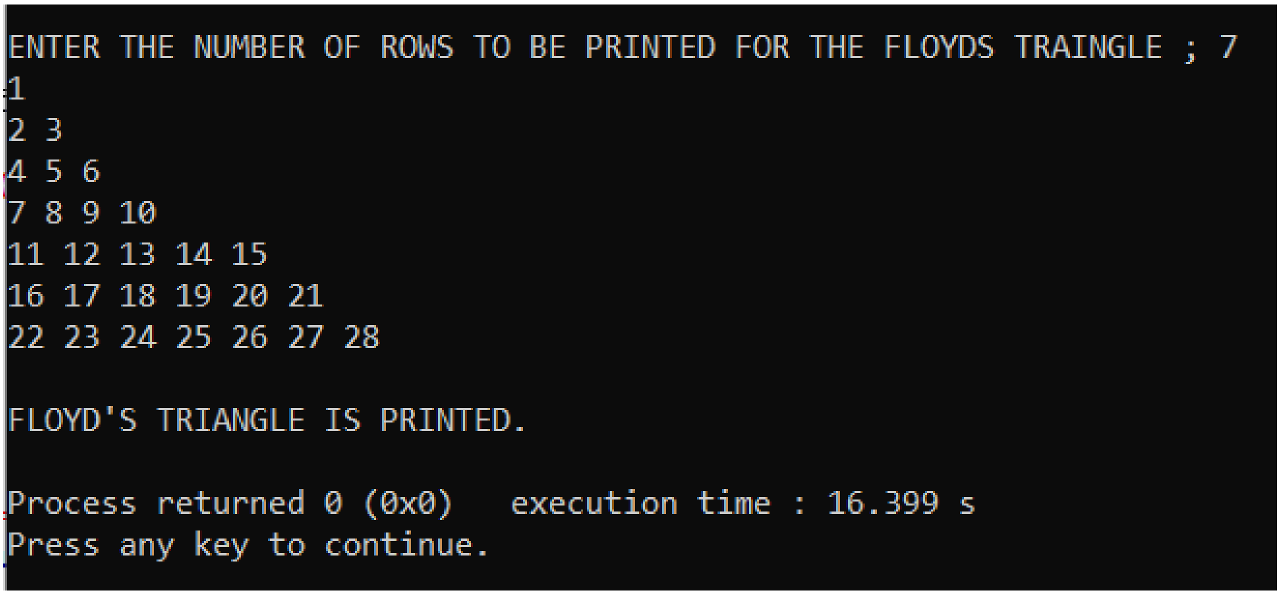


PROGRAM:

```
#include <stdio.h>
int main()
{
  int j,k,i,n;
  printf("\nENTER THE NUMBER OF ROWS TO BE PRINTED FOR THE
FLOYDS TRAINGLE ; ");
  scanf("%d",&n);
  i =1;
  k =1;
  while (! (i>n))
  {
    j =1;
```

```
     while (! (j>i))
     {
       printf("%d ",k);
       k =k+1;
       j =j+1;
     }
     printf("\n");
     i =i+1;
   }
   printf("\nFLOYD'S TRIANGLE IS PRINTED. \n");
   return 0;
}
```

OUTPUT:



```
ENTER THE NUMBER OF ROWS TO BE PRINTED FOR THE FLOYDS TRAINGLE ; 7
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
16 17 18 19 20 21
22 23 24 25 26 27 28

FLOYD'S TRIANGLE IS PRINTED.

Process returned 0 (0x0)   execution time : 16.399 s
Press any key to continue.
```

EXPERIMENT NO 7

CONTROL FLOW – III.

7.1)

AIM: Write a C program to find the sum of individual digits of a positive integer.

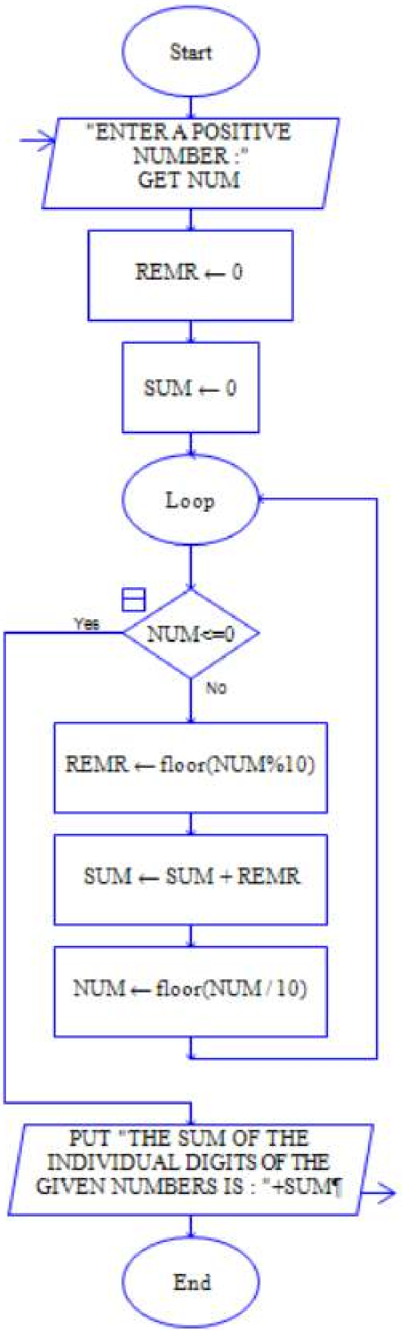
DESCRIPTION:

- **The** algo works by taking input from the user which are positive integers.
- Internal logical operations make sure that the output displayed, is the sum of the individual digits of a given number.

ALGORITHM:

```
STEP 1: START.
STEP 2: CRAETE NUM, REMR, SUM.
STEP 3: READ NUM.
STEP 4: SET R = 0.
STEP 5: SET SUM = 0.
STEP 7: while (!(NUM<=0))
SET : REMR =(NUM % 10);
    SUM =SUM+REMR;
    NUM =(NUM/10);
STEP 8: DISPLAY THE SUM.
STEP 9: END.
```

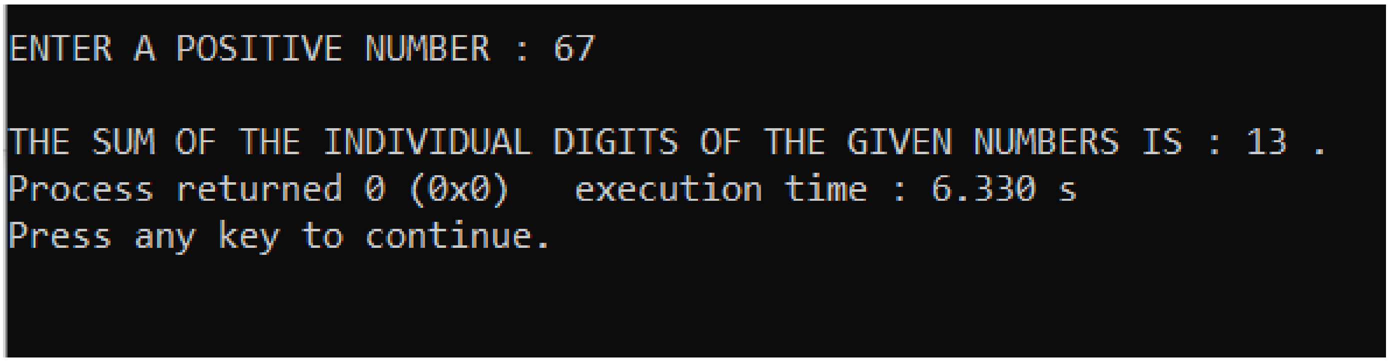
FLOWCHART:



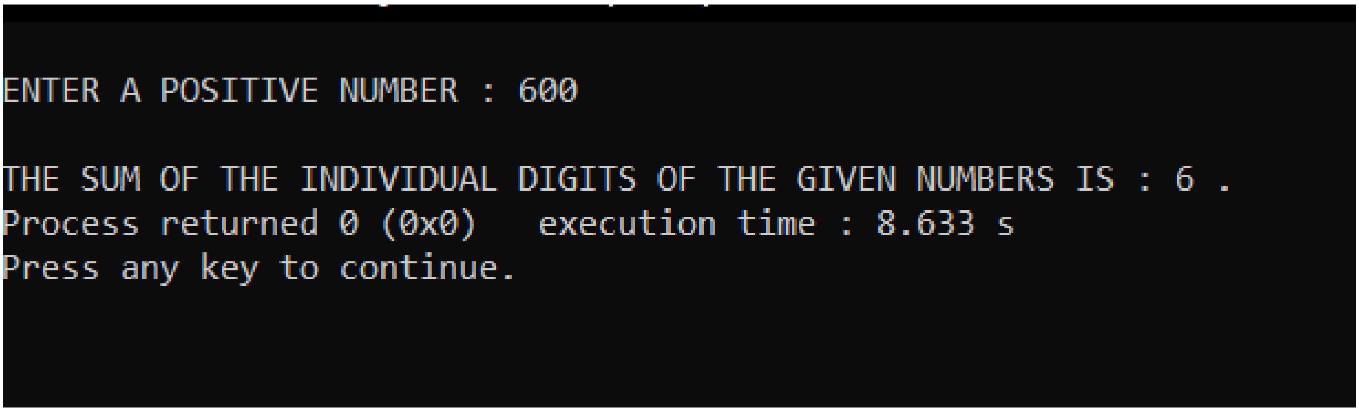
PROGRAM:

```
#include <stdio.h>

int main()

{

  int NUM,REMR,SUM;

  printf("\nENTER A POSITIVE NUMBER : ");

  scanf("%d",&NUM);
  REMR =0;

  SUM =0;

  while (! (NUM<=0))

  {

    REMR =(NUM % 10);

    SUM =SUM+REMR;

    NUM =(NUM/10);

  }

  printf("\nTHE SUM OF THE INDIVIDUAL DIGITS OF THE GIVEN
NUMBERS IS : %d .",SUM);

  return 0;

}
```

OUTPUT:



```
ENTER A POSITIVE NUMBER : 67

THE SUM OF THE INDIVIDUAL DIGITS OF THE GIVEN NUMBERS IS : 13 .
Process returned 0 (0x0)   execution time : 6.330 s
Press any key to continue.
```



```
ENTER A POSITIVE NUMBER : 600

THE SUM OF THE INDIVIDUAL DIGITS OF THE GIVEN NUMBERS IS : 6 .
Process returned 0 (0x0)   execution time : 8.633 s
Press any key to continue.
```

7.2)

AIM: Write a C program to check whether given number is palindrome or not

DESCRIPTION:

- The algo works by taking input from the user.
- Internal logic is developed to ensure that the right way of checking of whether the given number is a Palindrome or not.

ALGORITHM:

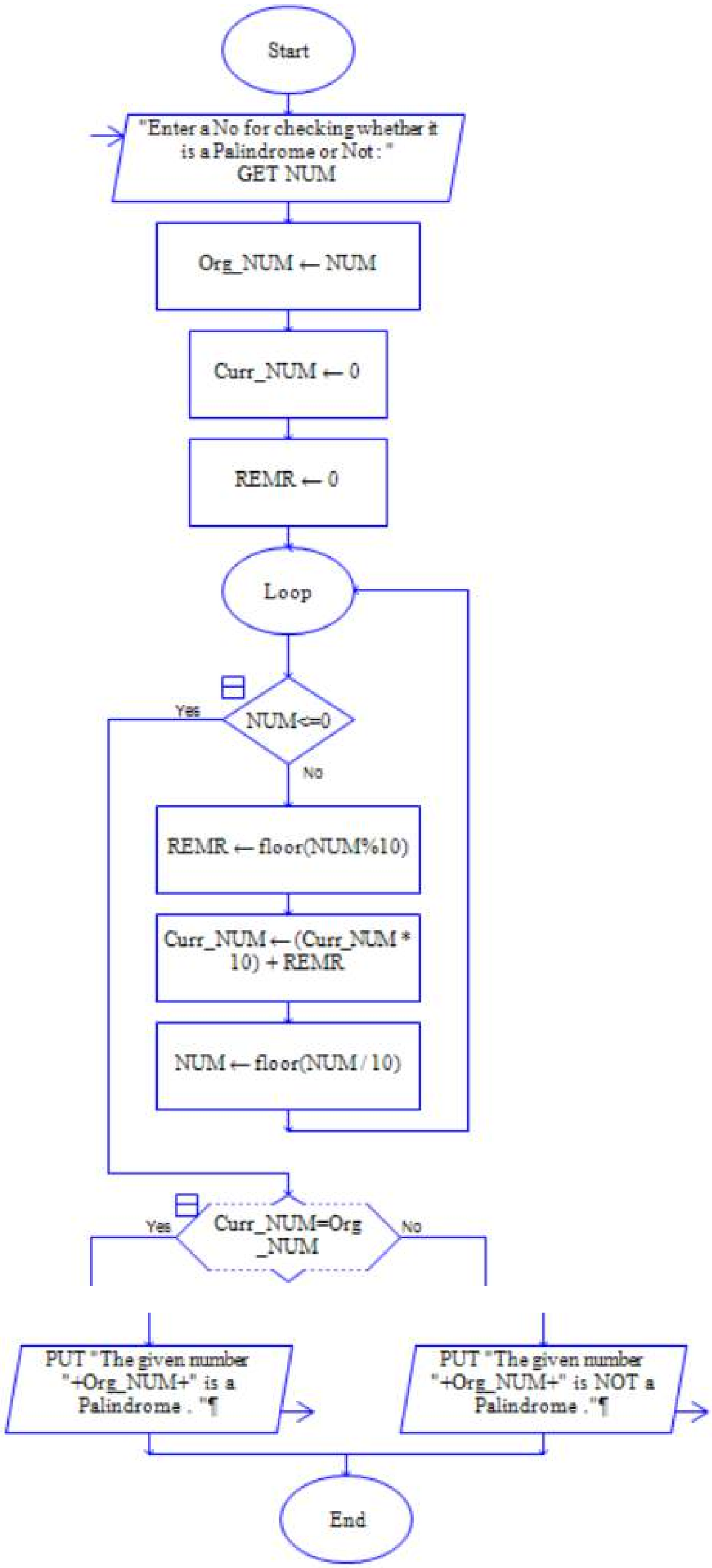
```
STEP 1: START.

STEP 2: DECLARE VARIABLES REMR, NUM, Org_NUM, Curr_NUM.

STEP 3: READ NUM.

STEP 4:

SET:

Org_NUM =NUM;

  Curr_NUM =0;

  REMR =0;

STEP 5: while (!(NUM<=0))

SET: REMR =floor(NUM % 10);

    Curr_NUM =(Curr_NUM*10)+REMR;

    NUM =floor(NUM/10);

STEP 7: if (Curr_NUM==Org_NUM)

STEP 8: DISPLAY PALINDROME.
```

STEP 9: ELSE

DISPLAY not A PANLINDROME.

STEP 10: END.

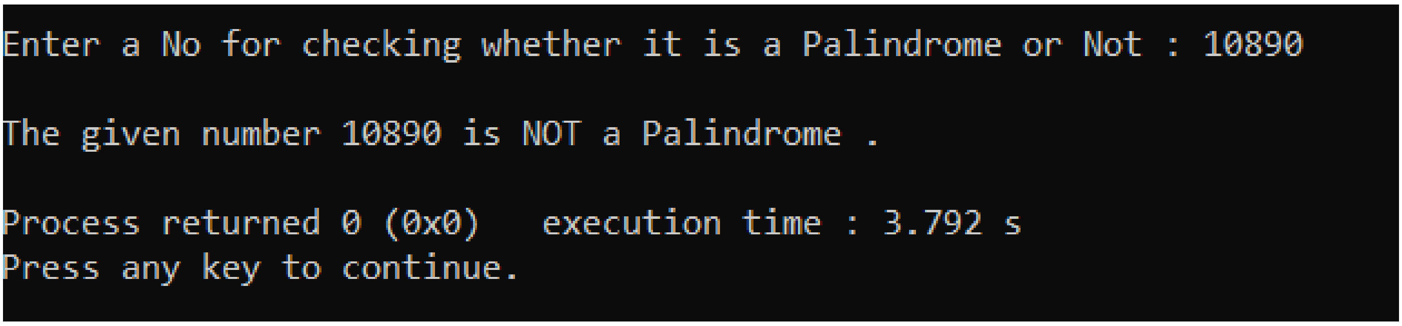
FLOWCHART:


Start
"Enter a No for checking whether it is a Palindrome or Not : "
GET NUM
Org_NUM ← NUM
Curr_NUM ← 0
REMR ← 0
Loop
NUM<=0
Yes
No
REMR ← floor(NUM%10)
Curr_NUM ← (Curr_NUM * 10) + REMR
NUM ← floor(NUM / 10)
Curr_NUM=Org_NUM
Yes
No
PUT "The given number "+Org_NUM+" is a Palindrome . "¶
PUT "The given number "+Org_NUM+" is NOT a Palindrome ."¶
End

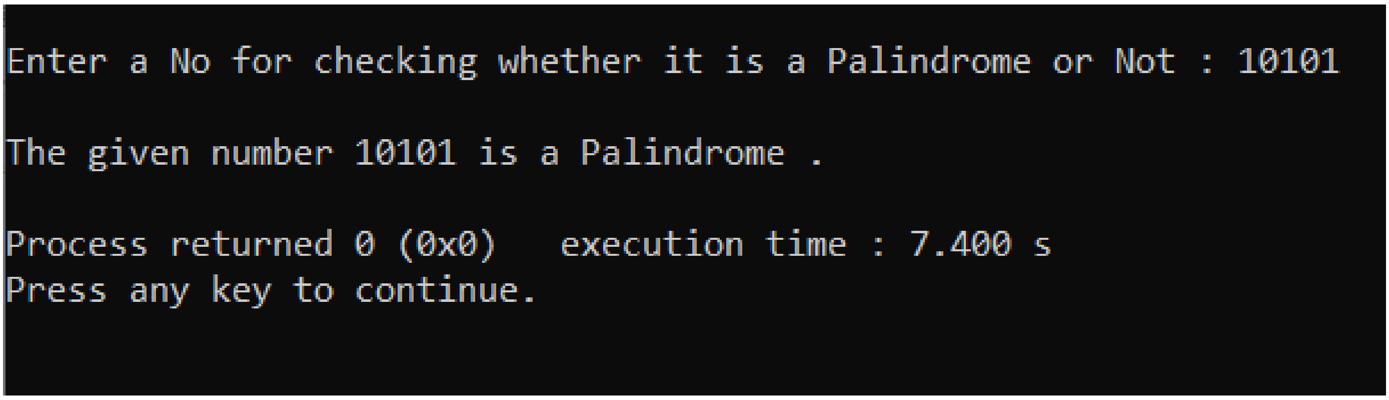
PROGRAM:

```
#include <stdio.h>
int main()
{
  int  REMR,NUM,Org_NUM,Curr_NUM;
  printf("\nEnter a No for checking whether it is a Palindrome or Not : ");
  scanf("%d",&NUM);
  Org_NUM =NUM;
  Curr_NUM =0;
  REMR =0;
  while (! (NUM<=0))
  {
    REMR =floor(NUM % 10);
    Curr_NUM =(Curr_NUM*10)+REMR;
    NUM =floor(NUM/10);
  }
  if (Curr_NUM==Org_NUM)
  {
    printf("\nThe given number %d is a Palindrome . \n",Org_NUM);
     }
  else
  {
    printf("\nThe given number %d is NOT a Palindrome .\n",Org_NUM);
     }
  return 0;
}
```

OUTPUT:



```
Enter a No for checking whether it is a Palindrome or Not : 10890

The given number 10890 is NOT a Palindrome .

Process returned 0 (0x0)   execution time : 3.792 s
Press any key to continue.
```



```
Enter a No for checking whether it is a Palindrome or Not : 10101

The given number 10101 is a Palindrome .

Process returned 0 (0x0)   execution time : 7.400 s
Press any key to continue.
```

7.3)

AIM: Write a C program to read two numbers, x and n, and then compute the sum of the geometric progression 1+x+x’+x^3……..+x^n.

DESCRIPTION:

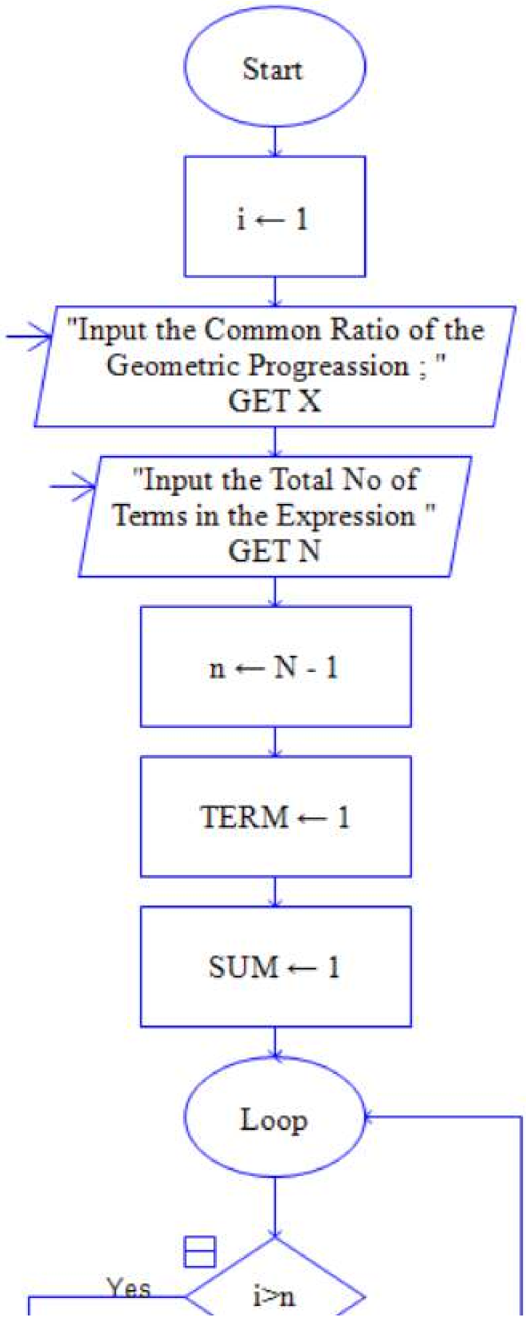
- The algorithm developed takes input from the user.
- Internal logic ensures of evaluating whether the entered numeric value is a palindrome or not.

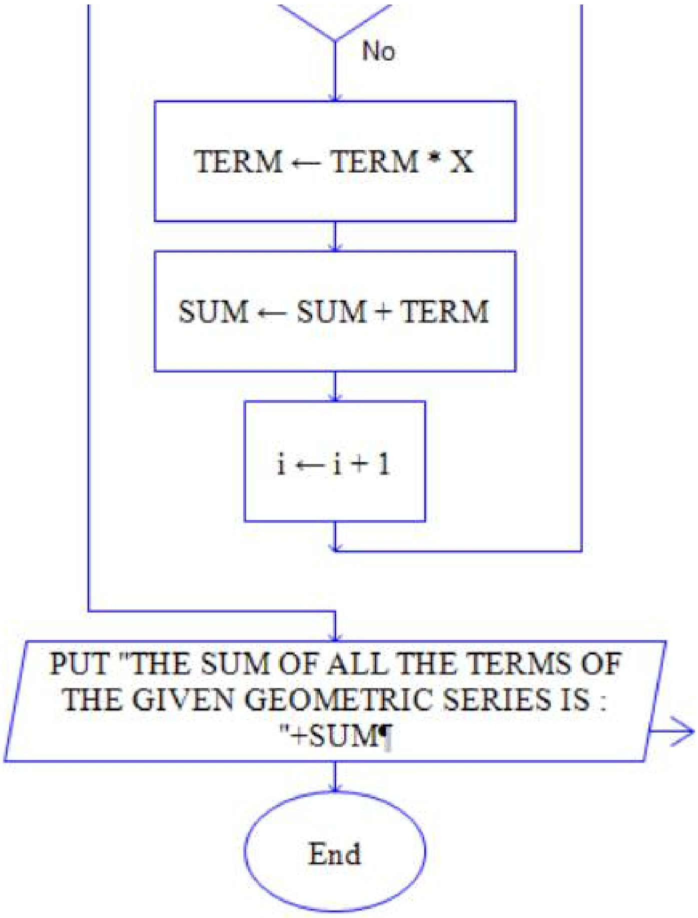
ALGORITHM:

```
STEP 1: START.

STEP 2: CREATE VARIABLES x, i, n, term ,sum.

STEP 3: SET i=1.

STEP 4: RAED x.

STEP 5: READ n.

STEP 7:

SET:  n =n-1.

        term =1.

        sum =1.

STEP 8: while (!(i>n))

SET: term =term*x.

         sum =sum+term.

          i =i+1.

STEP 9: DISPLAY SUM.
```

STEP 10: END.

FLOWCHART:

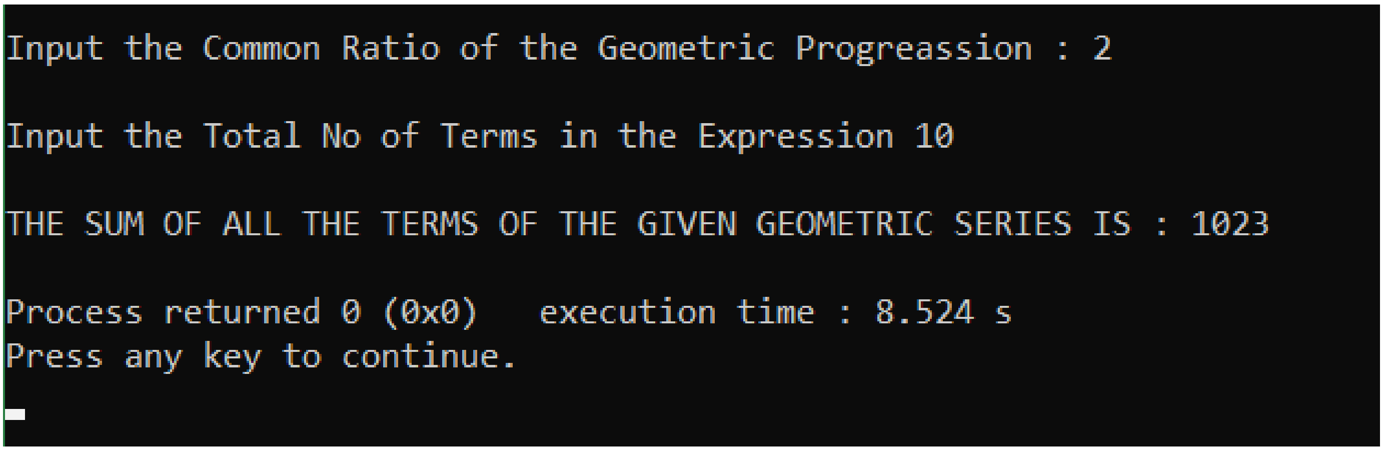

Start
i ← 1
"Input the Common Ratio of the Geometric Progreassion ; "
GET X
"Input the Total No of Terms in the Expression "
GET N
n ← N - 1
TERM ← 1
SUM ← 1
Loop
Yes
i>n


No
TERM ← TERM * X
SUM ← SUM + TERM
i ← i + 1
PUT "THE SUM OF ALL THE TERMS OF THE GIVEN GEOMETRIC SERIES IS : "+SUM¶
End

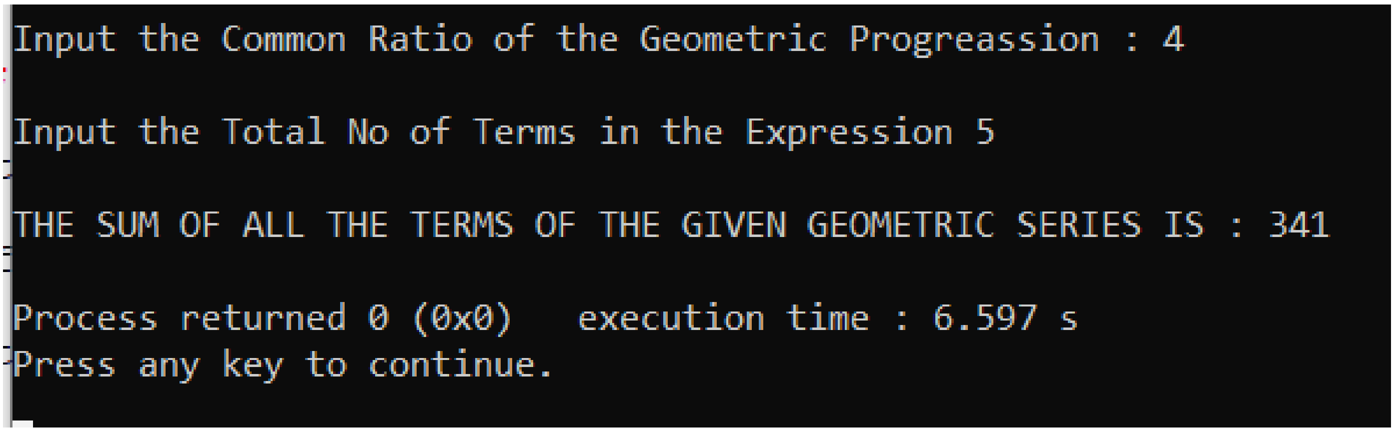
PROGRAM:

```
#include <stdio.h>
int main()
{
  int x,i,n,term,sum;
  i =1;
  printf("\nInput the Common Ratio of the Geometric Progreassion : ");
  scanf("%d",&x);
  printf("\nInput the Total No of Terms in the Expression ");

  scanf("%d",&n);
  n =n-1;
  term =1;
  sum =1;
  while (! (i>n))
  {
    term =term*x;
    sum =sum+term;
    i =i+1;
  }
  printf("\nTHE SUM OF ALL THE TERMS OF THE GIVEN GEOMETRIC SERIES IS : %d \n",sum);
  return 0;
}
```

<u>OUTPUT:</u>



```
Input the Common Ratio of the Geometric Progreassion : 2

Input the Total No of Terms in the Expression 10

THE SUM OF ALL THE TERMS OF THE GIVEN GEOMETRIC SERIES IS : 1023

Process returned 0 (0x0)   execution time : 8.524 s
Press any key to continue.
```



```
Input the Common Ratio of the Geometric Progreassion : 4

Input the Total No of Terms in the Expression 5

THE SUM OF ALL THE TERMS OF THE GIVEN GEOMETRIC SERIES IS : 341

Process returned 0 (0x0)   execution time : 6.597 s
Press any key to continue.
```

7.4)

AIM: Sorting the elements of an array using selection sort or linear searching algorithm.

DESCRIPTION:

The algorithm developed here takes input from the user for the size of the array and the elements into it.

Internal code ensures that the given array is sorted in ascending order.

ALGORITHM:

STEP 1: CREATE AN ARRAY a[] OF ANY SIZE , INTERGER n , I, j, temp.

STEP 2: READ ELEEMENTS OF THE ARRAY a[n].

STEP 3: RUN A FOR LOOP ITERATING FROM i=0 TO i<n.

STEP 4: RUN ANOTHER FOR LOOP INSIDE THE ABOVE LOOP INTERATING FROM j=i+1 TO j<n.

STEP 5: IF ANY WHERE a[j] < a[i] ,

STORE a[j] IN temp.

STEP 6: UPDATE a[j] WITH VALUE OF a[i].

STEP 7: UPDATE VALUE OF a[i] WITH temp.

STEP 8: USE ANOTHER FOOR LOOP OUTSIDE OF ALL THE ABOVE LOOPS, DISPLAY THE SORTED ARRAY.

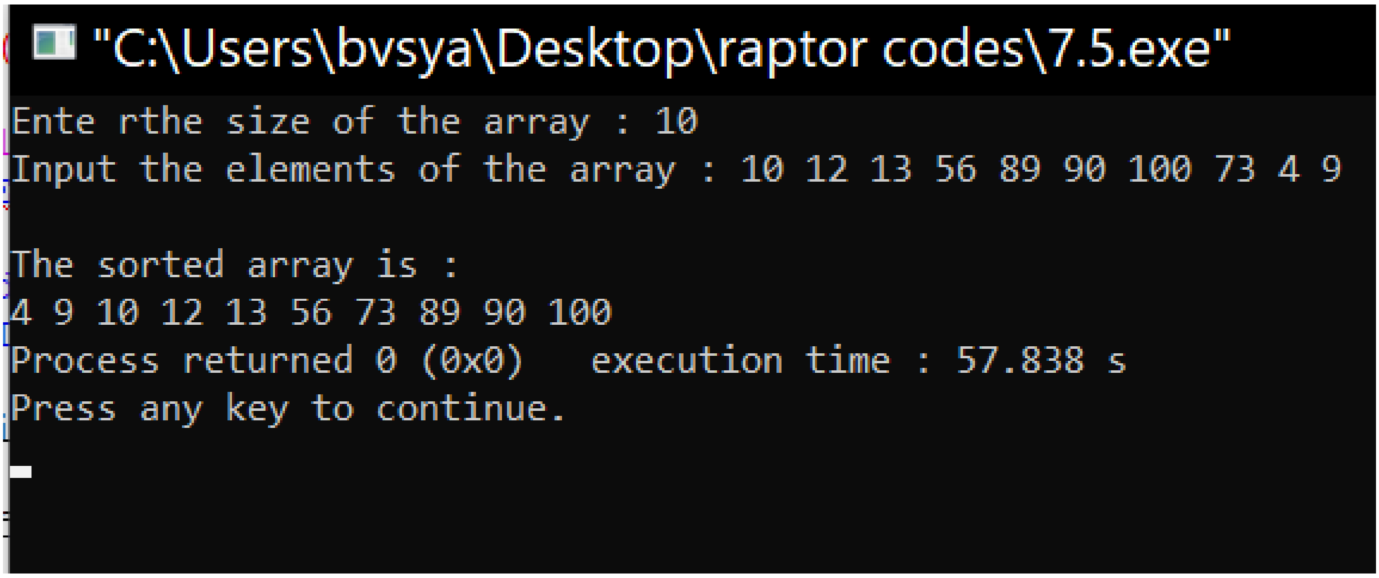
STEP 9: END.

PROGRAMME:

```
#include<stdio.h>
```

```
int main(){
int n,a[10],i,j,temp;
printf("Enter the size of the array : ");
scanf("%d",&n);
printf("Input the elements of the array : ");
for(int i=0;i<n;i++)
{
    scanf("%d",&a[i]);
}
for(i=0;i<n-1;i++){
    for(j=i+1;j<n;j++){
        if(a[j]<a[i]){
            temp = a[j];
            a[j]=a[i];
            a[i]=temp;
        }
    }
}
printf("\nThe sorted array is : \n");
for(int i=0;i<n;i++){
    printf("%d ",a[i]);
}
}
```

OUTPUT:



"C:\Users\bvsya\Desktop\raptor codes\7.5.exe"

```
Ente rthe size of the array : 10
Input the elements of the array : 10 12 13 56 89 90 100 73 4 9

The sorted array is :
4 9 10 12 13 56 73 89 90 100
Process returned 0 (0x0)   execution time : 57.838 s
Press any key to continue.
```

7.5)

AIM: WRITE AN ALGORITHM TO DISPLAY THE FREQUENCY OF EACH ELEMENT IN AN ARRAY.

DESCRIPTION:

- The algorithm developed here takes input from the user an array of elements whose size is given as 20.
- The program displays the frequency o each element in the array as the output.

ALGORITHM:

Step 1:START.

Step 2: declare a , b two arrays of size required , variables count,I,j all of in type.

Step 3: read elements of the array 'a' using a for loop.

Step 4: use a for loop iterating from I = 0 to I < 20 , increment I by 1 at each iteration. Include the following steps in it.

Step 5: in the loop initilaise count = 1.

Step 6: if a[i] ! = INT_MAX, i.e a normal value is found at that index.

Step 7: then inside the above statement place a for loop that starts iterating from j= i+1 and goes upto last index of the array.

Step 8: inside the above for loop if at any iteration element in a[i] is found equal to a[j], then increment count by 1. And assign INT_MAX at those indexe's value fields.

Step 9: end if, end for.

Step 10: assign b[i] = count after all iterations of for loop. This array keeps track of the frequency of a number in the given array a.

Step 11: end for.

Step 12: using a for loops print the the elements and their frequency if they satisfy the condition a[i] ! = INT_MAX (this ensures a value is occurred once in the array a).

Step 13: END.

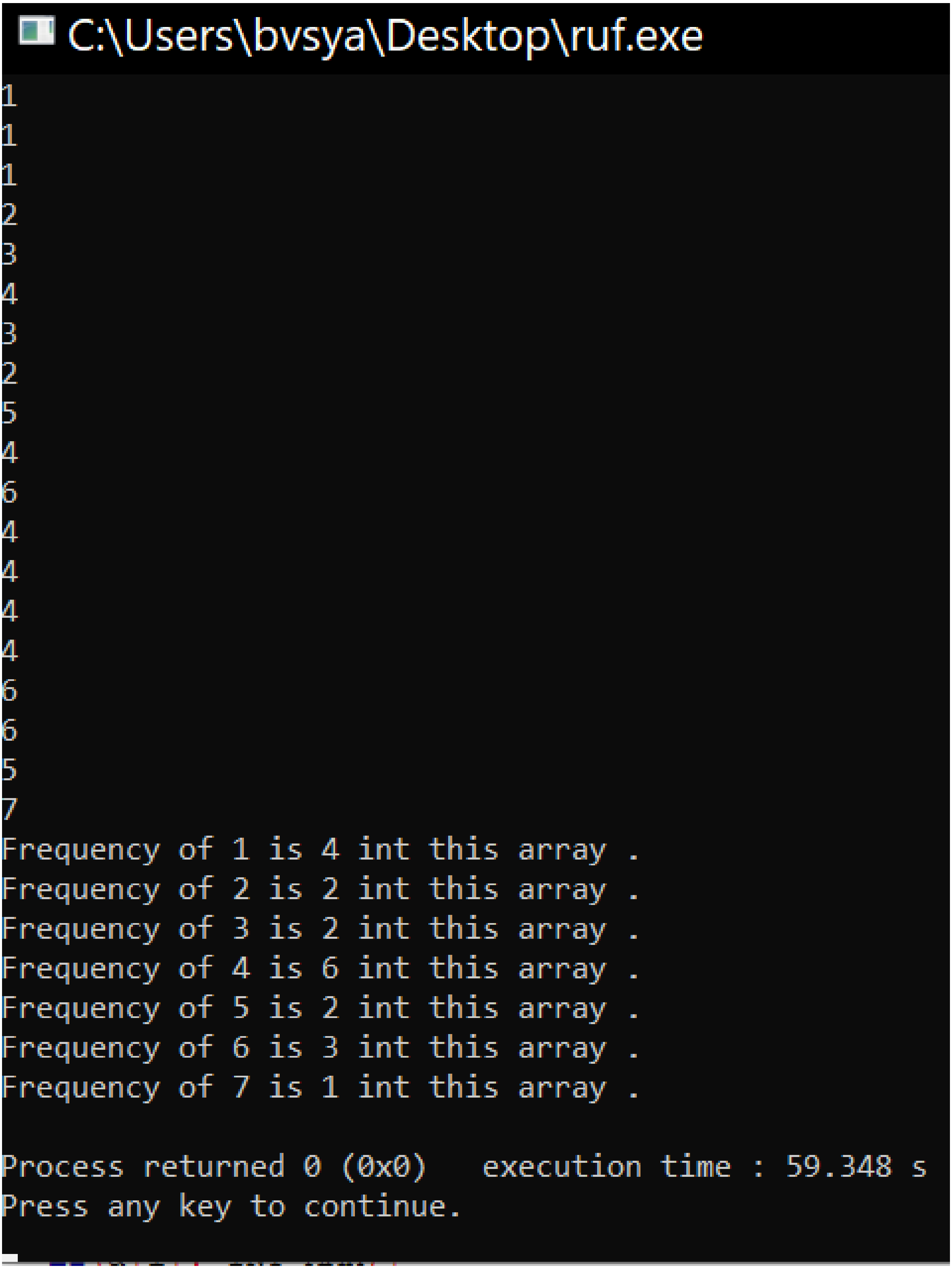
FLOWCHART:

PROGRAMME:

```
#include<stdio.h>

#include<limits.h>

int main(){

  int a[20],b[20],count,i,j;

  for(i=0;i<20;i++){

    scanf("%d",&a[i]);

  }

  for(i=0;i<20;i++){

    count=1;

    if(a[i]! =INT_MAX){

    for(j=i+1;j<20;j++){

      if(a[i]==a[j]){

        count++;

        a[j]=INT_MAX;
```

```
            }
        }
        b[i]=count;
        }
    }
    for(i=0;i<20;i++){
        if(a[i]! =INT_MAX){
            printf("Frequency of %d is %d int this array .\n",a[i],b[i]);
        }
    }
    return 0;
}
```

OUTPUT:



```
1
1
1
2
3
4
3
2
5
4
6
4
4
4
4
6
6
5
7
Frequency of 1 is 4 int this array .
Frequency of 2 is 2 int this array .
Frequency of 3 is 2 int this array .
Frequency of 4 is 6 int this array .
Frequency of 5 is 2 int this array .
Frequency of 6 is 3 int this array .
Frequency of 7 is 1 int this array .

Process returned 0 (0x0)   execution time : 59.348 s
Press any key to continue.
```

EXPERIMENT NO 8: ARRAYS

8.1)

AIM: Write a C program to search an element in the given array (Linear Search.

DESCRIPTION:

The algorithm works by taking inputs for the size, elements of the array and the element to be found. Linear Search is performed to find the index of the given number in the array.

ALGORITHM:

STEP 1: START.

STEP 2: Read size of the array.

STEP 3: Read the Elements of the array by using a for loop.

STEP 4: Read the key number to be searched.

STEP 5: Using a for loop iterate over the array and search for the key given.

STEP6: If key value = element in the array

Print the index of the element int the array and break the loop.

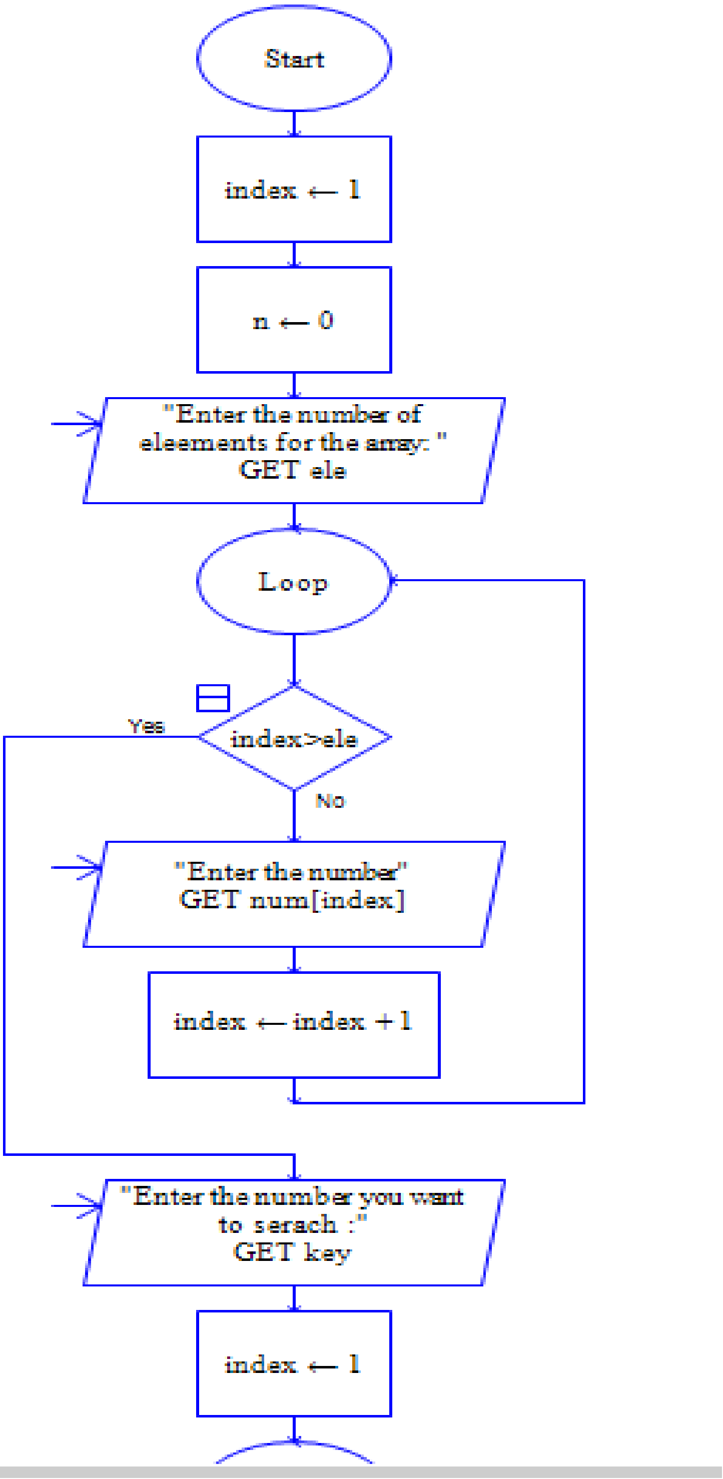
STEP 7: If no, continue for the next iteration.

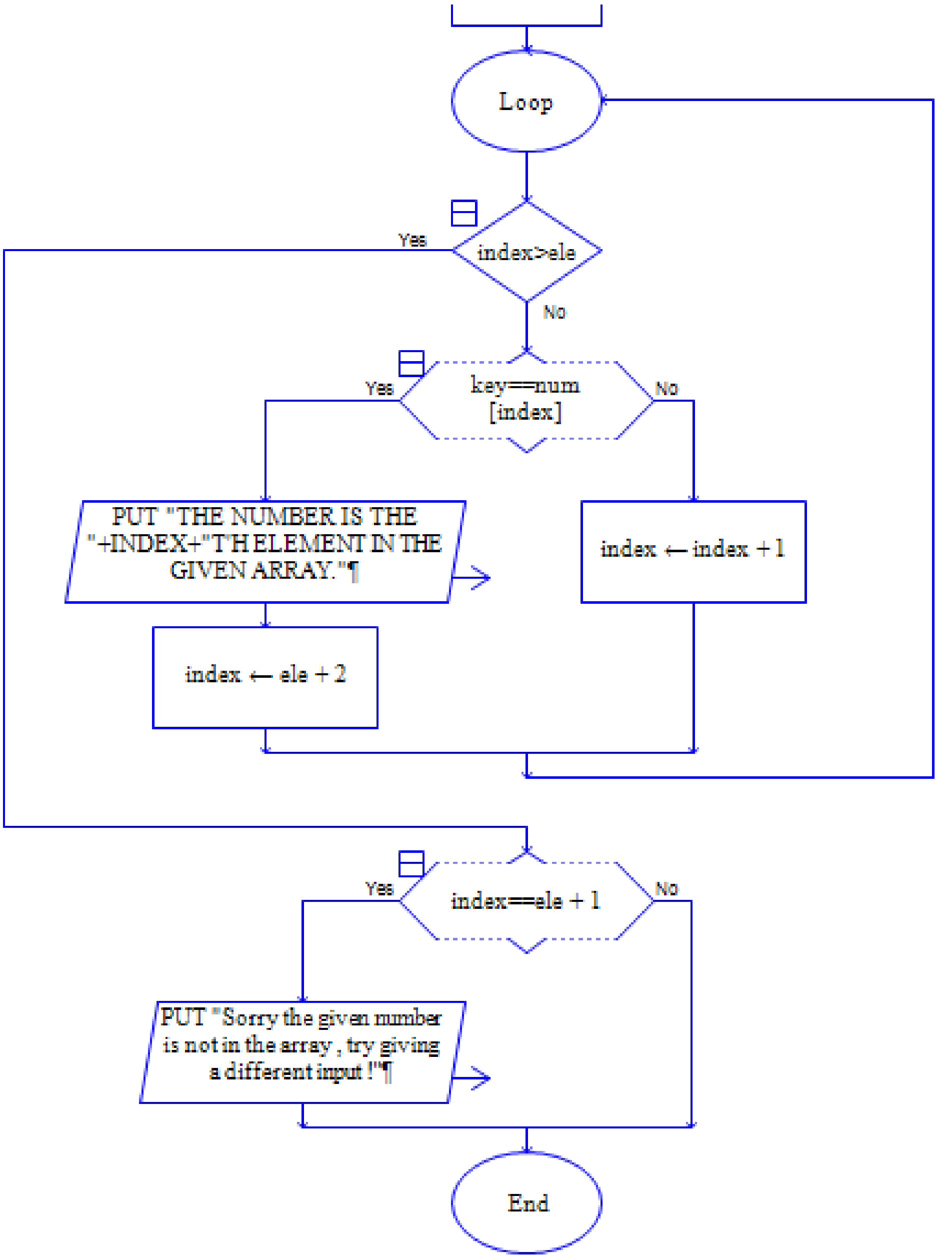
STEP 8: After all the iterations, still if no element matches.

STEP 9: print the element is not found.

STEP 10: End.

FLOWCHART:


Start
index ← 1
n ← 0
"Enter the number of eleements for the array: "
GET ele
Loop
index>ele
Yes
No
"Enter the number"
GET num[index]
index ← index + 1
"Enter the number you want to serach :"
GET key
index ← 1


Loop
index>ele
Yes
No
key==num [index]
Yes
No
PUT "THE NUMBER IS THE "+INDEX+"TH ELEMENT IN THE GIVEN ARRAY."¶
index ← index + 1
index ← ele + 2
index==ele + 1
Yes
No
PUT "Sorry the given number is not in the array , try giving a different input !"¶
End

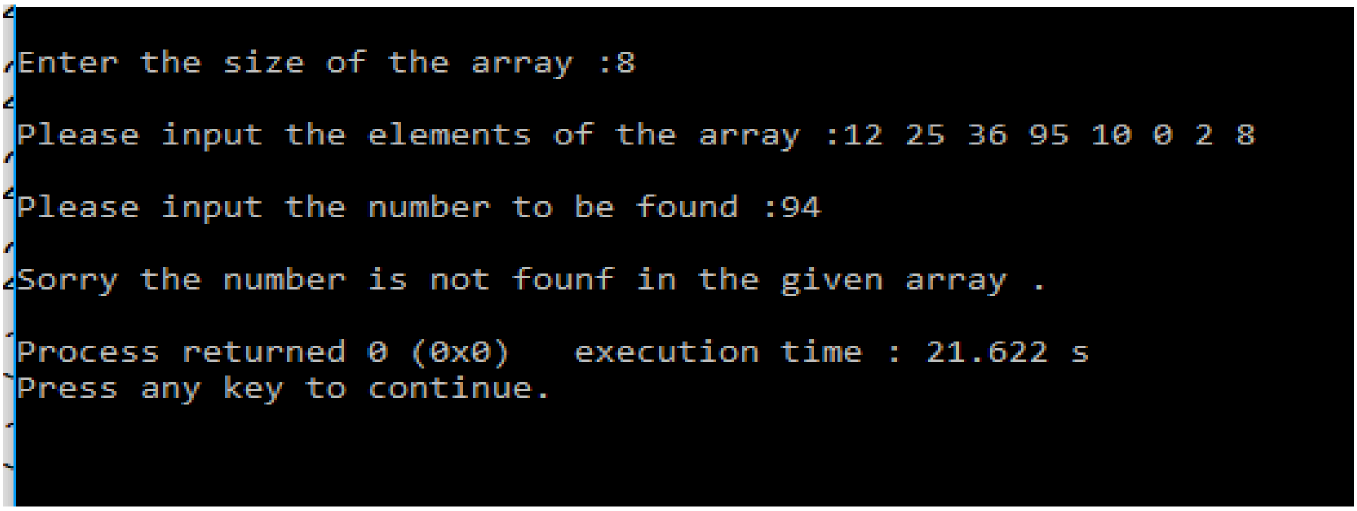
PROGRAM:

```
#include <stdio.h>

void main()

{

  int key,i;

  int n;

  int arr[50];
```

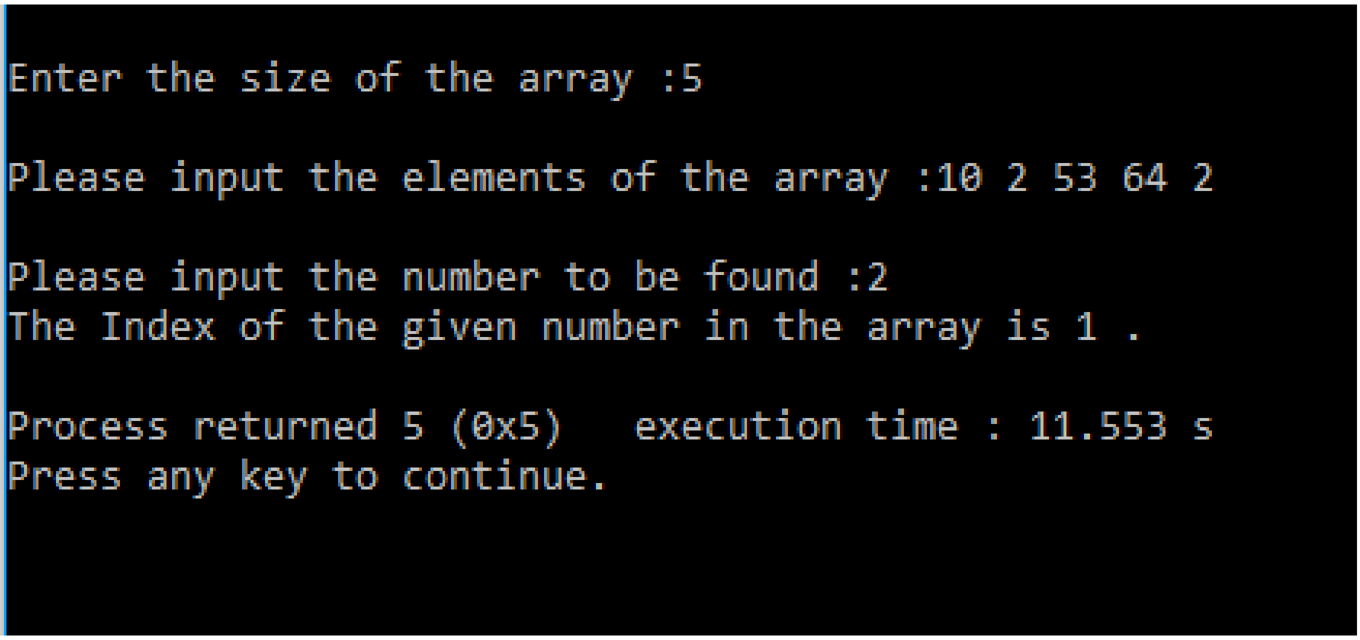
```
printf("\nEnter the size of the array :");
scanf("%d",&n);
printf("\nPlease input the elements of the array :");
for(i=0;i<n;i++)
{
    scanf("%d",&arr[i]);
}
printf("\nPlease input the number to be found :");
scanf("%d",&key);
for(i=0;i<n;i++)
{
    if(key==arr[i])
    {
        printf("The Index of the given number in the array is %d .\n",i);
        break;
    }
    else
    {       continue;
    }
```

```
  }

  if(i==n)

  {

     printf("\nSorry the number is not founf in the given array .\n");

  }

}
```

OUTPUT:



```
Enter the size of the array :8

Please input the elements of the array :12 25 36 95 10 0 2 8

Please input the number to be found :94

Sorry the number is not founf in the given array .

Process returned 0 (0x0)   execution time : 21.622 s
Press any key to continue.
```



```
Enter the size of the array :5

Please input the elements of the array :10 2 53 64 2

Please input the number to be found :2
The Index of the given number in the array is 1 .

Process returned 5 (0x5)   execution time : 11.553 s
Press any key to continue.
```

8.2)

AIM: Write a C program to perform matrix addition.

DESCRIPTION:

The program works by taking input from the user the sizes of the two – two dimensional arrays.

Performs addition only if the dimensions of both the arrays are same.

ALGORITHM:

STEP 1: START.

STEP 2: Read the dimensions of matrix 1.

STEP 3: Read the dimensions of matrix 2.

STEP 4: if Dimensions of both the matrices are equal, perform matrix addition.

STEP 5: Scan the elements of the two matrices using nested for loops.

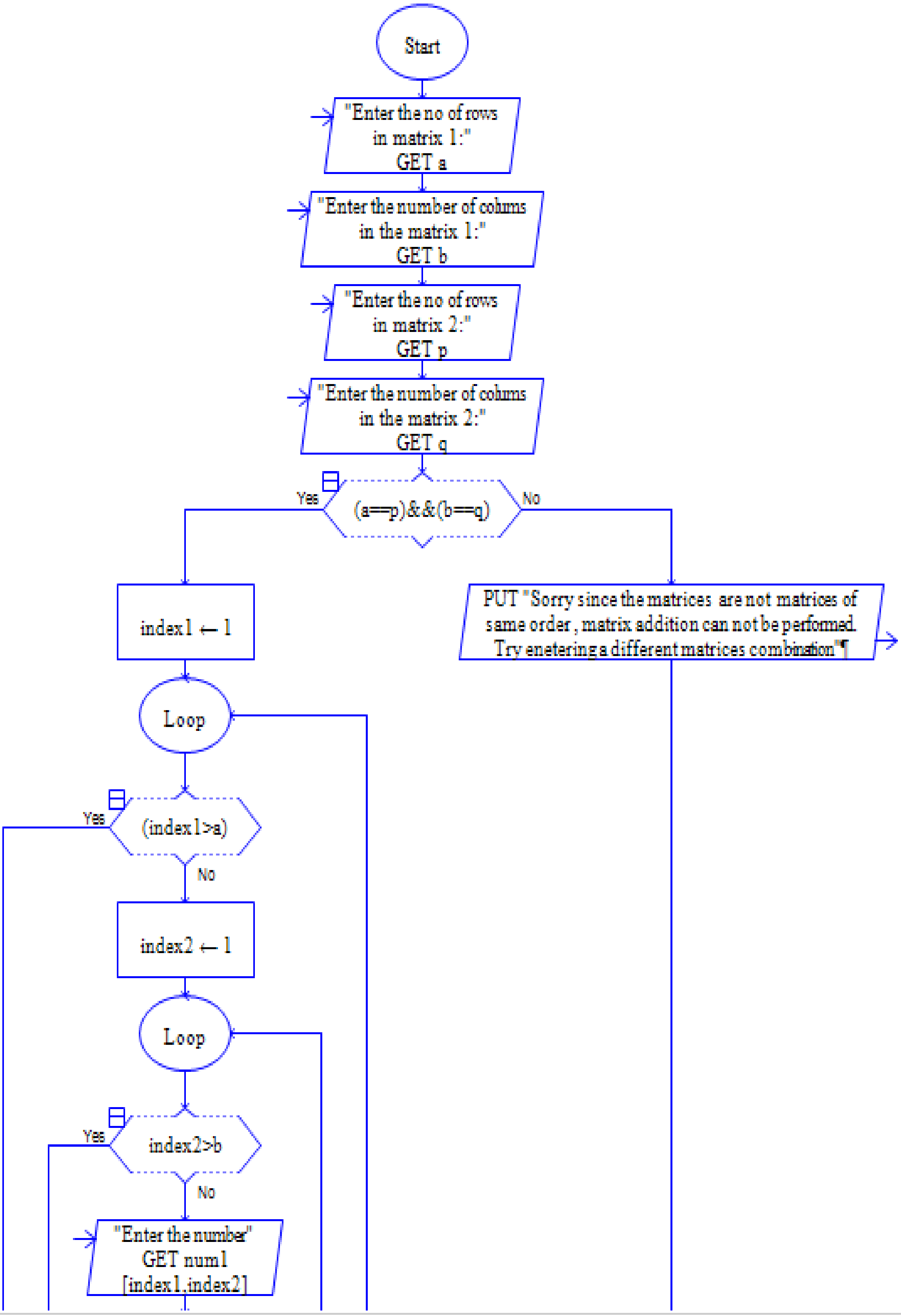
STEP6: Using nested for loops perform matrix addition and store the elements the new array.

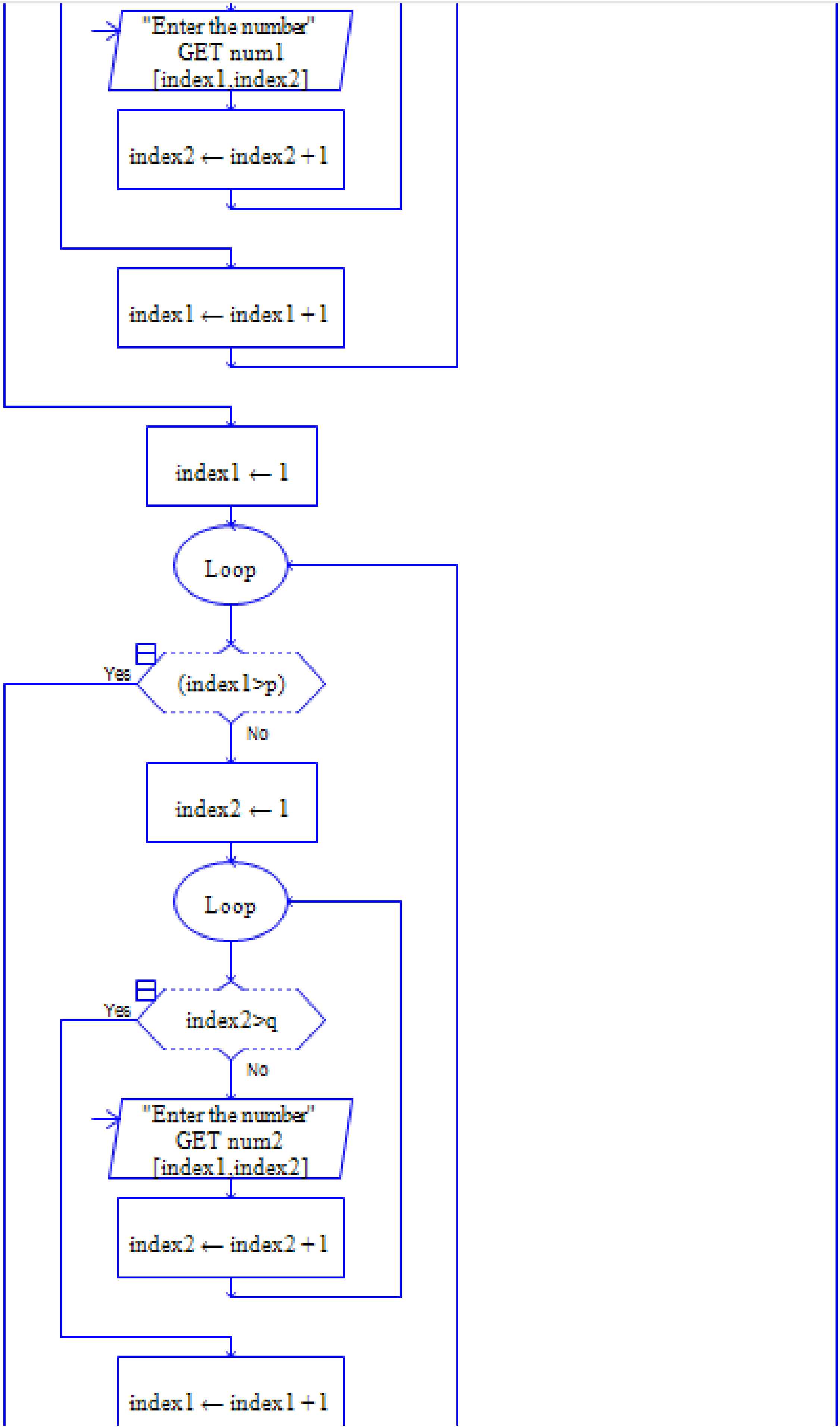
STEP 7: Print The array obtained array after addition.

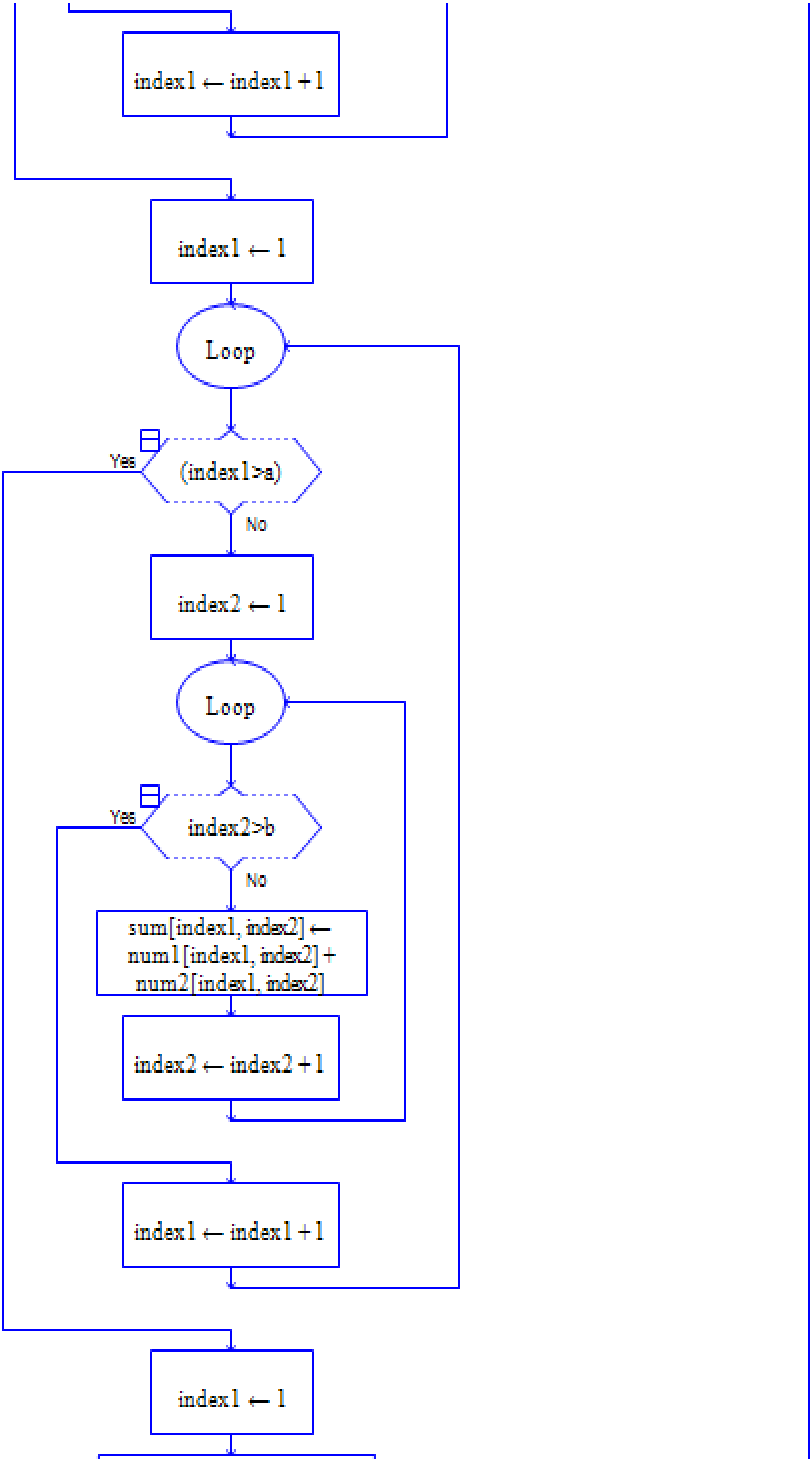
STEP 8: if no, display the dimensions of matrices should be equal to perform matrix addition.

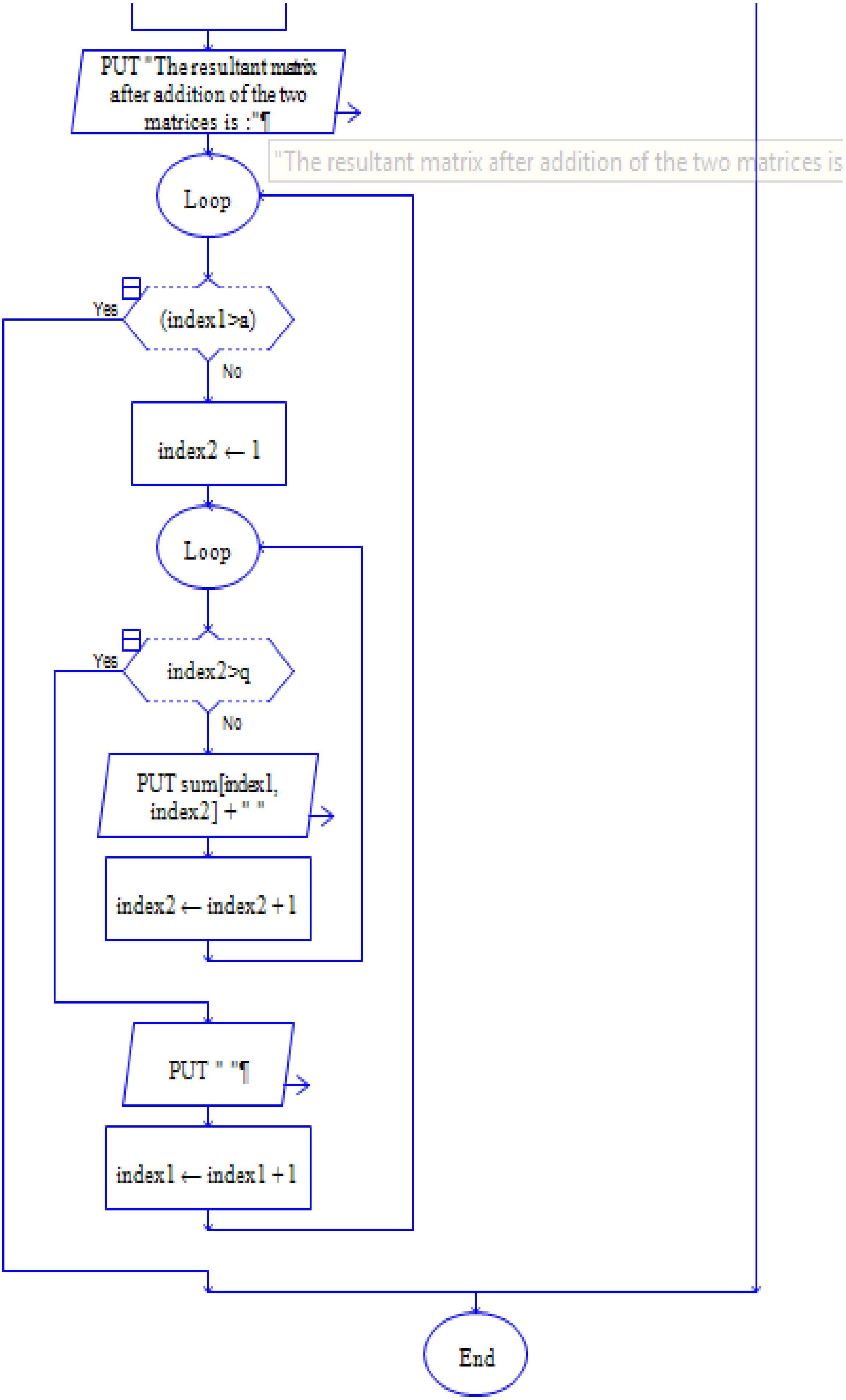
STEP 9: END.

FLOWCHART:


Start
"Enter the no of rows in matrix 1:" GET a
"Enter the number of colums in the matrix 1:" GET b
"Enter the no of rows in matrix 2:" GET p
"Enter the number of colums in the matrix 2:" GET q
Yes
(a==p)&&(b==q)
No
index1 ← 1
PUT "Sorry since the matrices are not matrices of same order , matrix addition can not be performed. Try eneteringa different matrices combination"¶
Loop
Yes
(index1>a)
No
index2 ← 1
Loop
Yes
index2>b
No
"Enter the number" GET num1 [index1,index2]


"Enter the number"
GET num1
[index1,index2]
index2 ← index2 + 1
index1 ← index1 + 1
index1 ← 1
Loop
Yes
(index1>p)
No
index2 ← 1
Loop
Yes
index2>q
No
"Enter the number"
GET num2
[index1,index2]
index2 ← index2 + 1
index1 ← index1 + 1

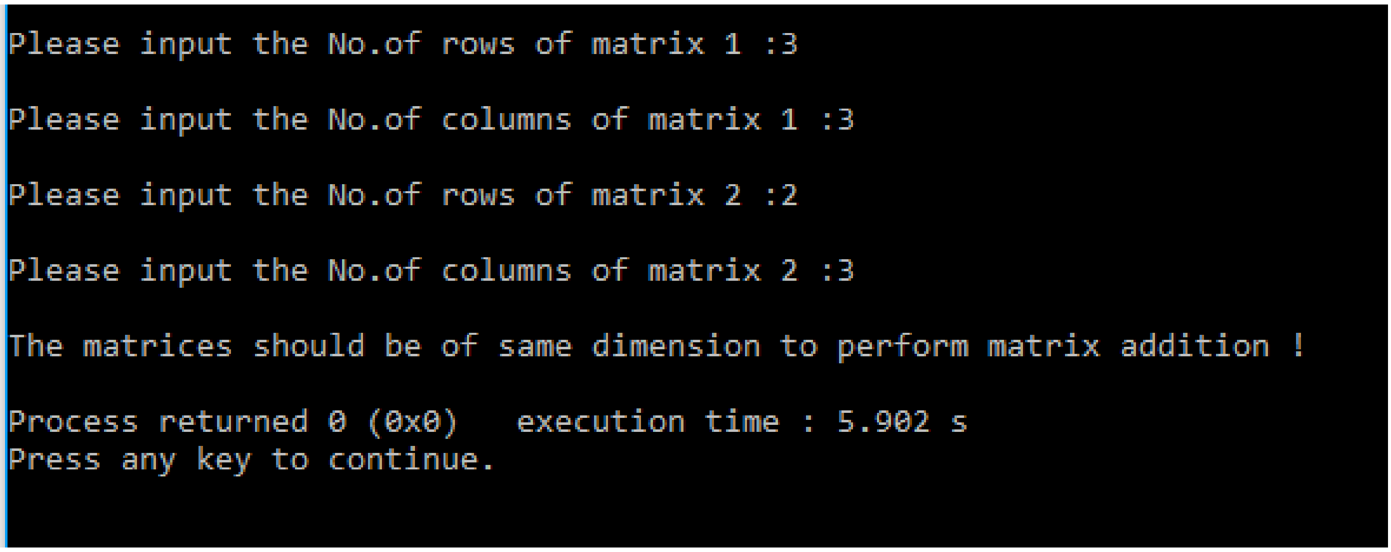

index1 ← index1 + 1
index1 ← 1
Loop
Yes
(index1>a)
No
index2 ← 1
Loop
Yes
index2>b
No
sum[index1, index2] ← num1[index1, index2] + num2[index1, index2]
index2 ← index2 + 1
index1 ← index1 + 1
index1 ← 1


PUT "The resultant matrix after addition of the two matrices is :"¶
"The resultant matrix after addition of the two matrices is
Loop
(index1>a)
Yes
No
index2 ← 1
Loop
index2>q
Yes
No
PUT sum[index1, index2] + " "
index2 ← index2 + 1
PUT " "¶
index1 ← index1 + 1
End

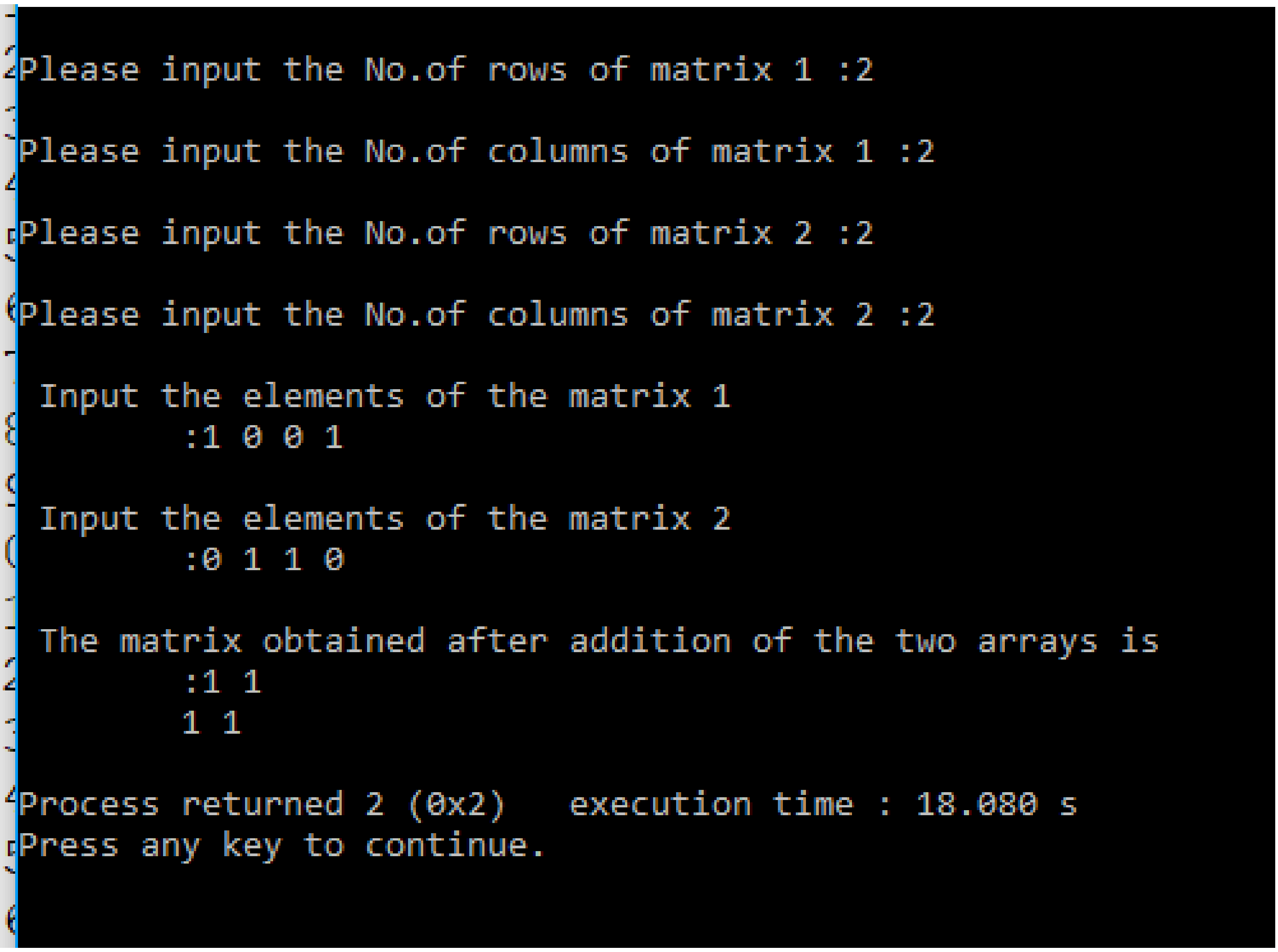
PROGRAM:

```
#include <stdio.h>

void main()

{

  int a[20][20],b[20][20],c[20][20],i,j,x,y,p,q;

  printf("\nPlease input the No.of rows of matrix 1 :");

  scanf("%d",&x);

  printf("\nPlease input the No.of columns of matrix 1 :");

  scanf("%d",&y);

  printf("\nPlease input the No.of rows of matrix 2 :");

  scanf("%d",&p);

  printf("\nPlease input the No.of columns of matrix 2 :");

  scanf("%d",&q);

  if((x==p)&&(y==q))

  {

     printf("\n Input the elements of the matrix 1\n\t:");

     for(i=0;i<x;i++)

     {

        for(j=0;j<y;j++)

        {

           scanf("%d",&a[i][j]);
  }
}
printf("\n Input the elements of the matrix 2\n\t:");
for(i=0;i<x;i++)
{
  for(j=0;j<y;j++)
  {
    scanf("%d",&b[i][j]);
  }
}
for(i=0;i<x;i++)
{
  for(j=0;j<y;j++)
  {
    c[i][j] = a[i][j]+b[i][j];
  }
}
printf("\n The matrix obtained after addition of the two arrays is\n\t:");
for(i=0;i<x;i++)
{
      for(j=0;j<y;j++)
      {
        printf("%d ",c[i][j]);
      }
      printf(" \n\t");
    }
  }
  else
  {
    printf("\nThe matrices should be of same dimension to perform matrix addition !\n");
  }
}
```

OUTPUT:



```
Please input the No.of rows of matrix 1 :3

Please input the No.of columns of matrix 1 :3

Please input the No.of rows of matrix 2 :2

Please input the No.of columns of matrix 2 :3

The matrices should be of same dimension to perform matrix addition !

Process returned 0 (0x0)   execution time : 5.902 s
Press any key to continue.
```



```
Please input the No.of rows of matrix 1 :2

Please input the No.of columns of matrix 1 :2

Please input the No.of rows of matrix 2 :2

Please input the No.of columns of matrix 2 :2

 Input the elements of the matrix 1
        :1 0 0 1

 Input the elements of the matrix 2
        :0 1 1 0

 The matrix obtained after addition of the two arrays is
        :1 1
        1 1

Process returned 2 (0x2)   execution time : 18.080 s
Press any key to continue.
```

8.3)

AIM: Write a C program to perform matrix multiplication.

DESCRIPTION:

The program works by taking input of two – two dimensional arrays.

Performs matrix multiplication only if the no of columns of first matrix are equal to no of rows of second matrix.

Prints the matrix obtained after matrix multiplication.

ALGORITHM:

STEP 1: START.

STEP 2: Read the dimensions of matrix 1.

STEP 3: Read the dimensions of matrix 2.

STEP 4: if columns of the matrix 1 is equal to rows of matrix 2, we perform matrix multiplication.

STEP 5: Scan the elements of the two matrices using nested for loops.

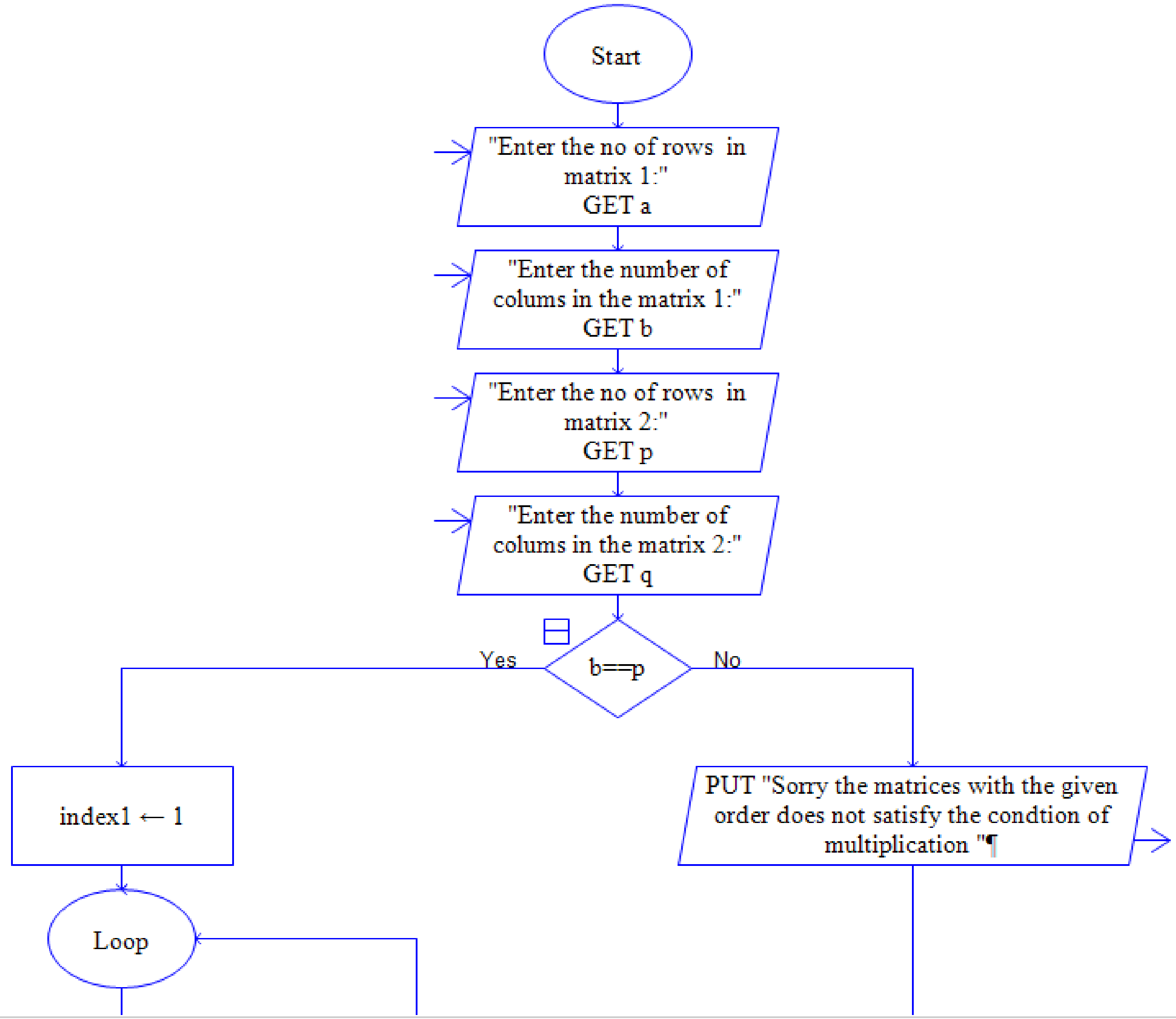
STEP6: Using nested for loops perform matrix multiplication and store the elements the new array.

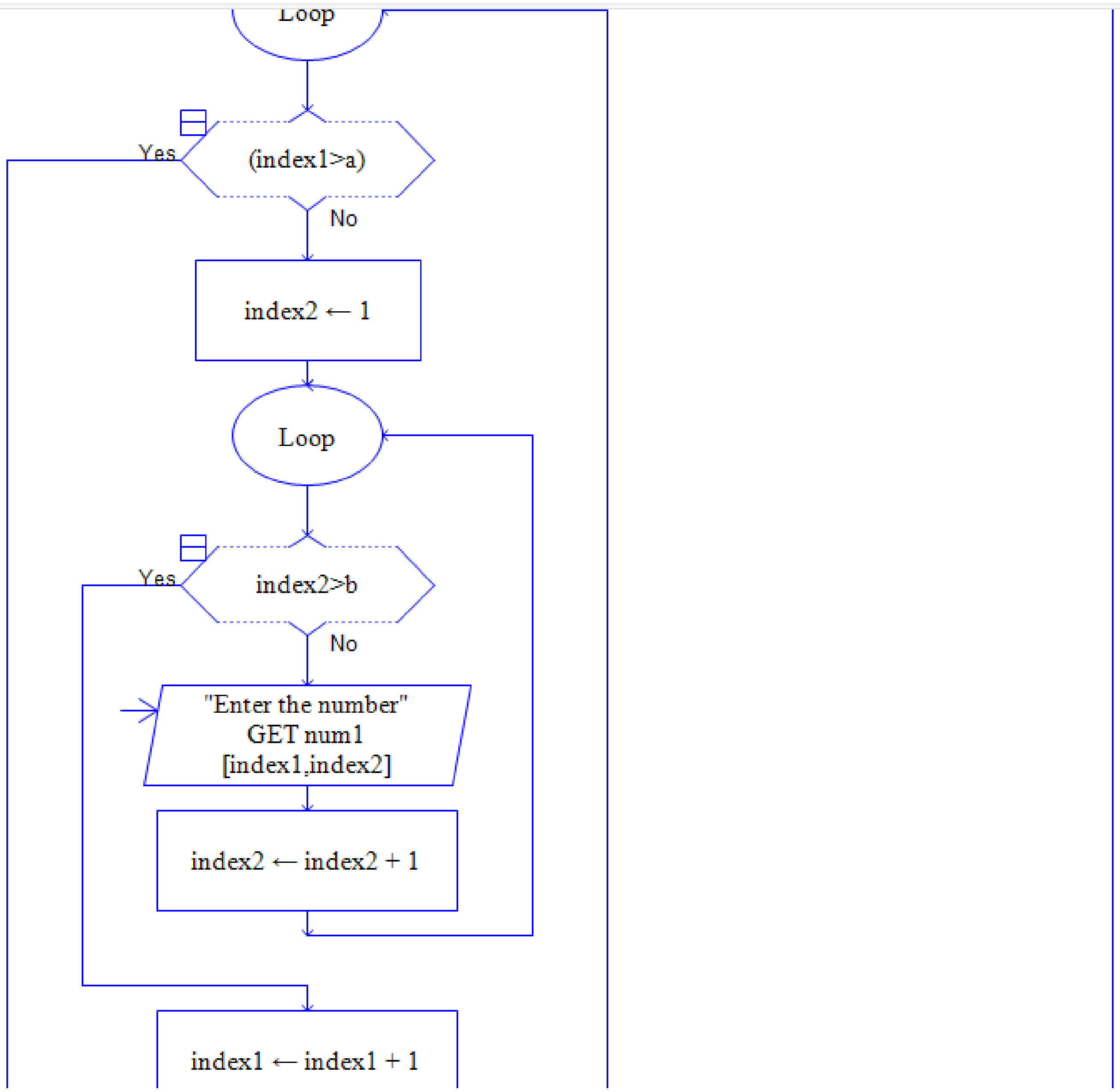
STEP 7: Print The array obtained array after multiplication.

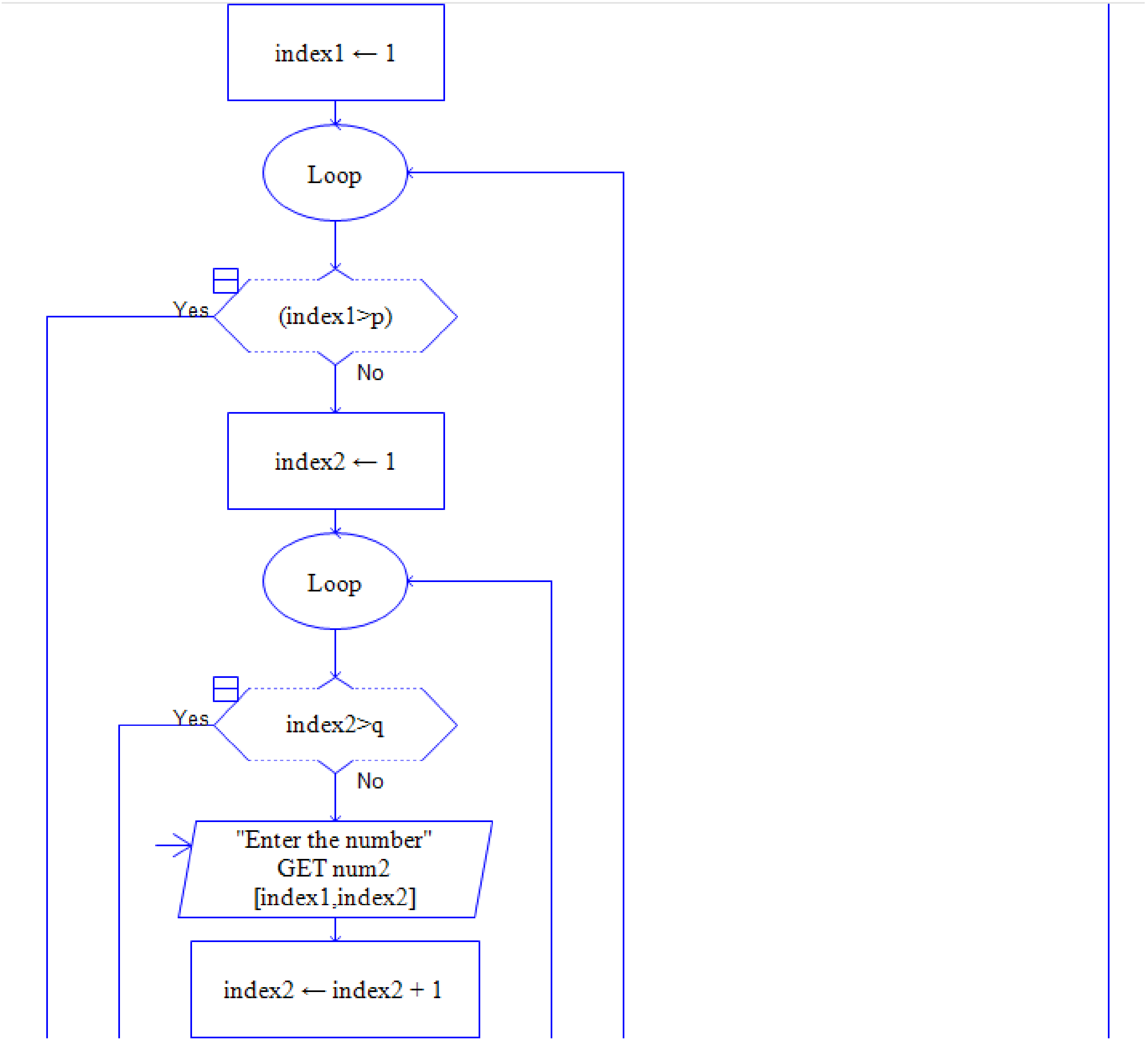
STEP 8: if no, display the dimensions of matrices are not valid for performing matrix multiplication.

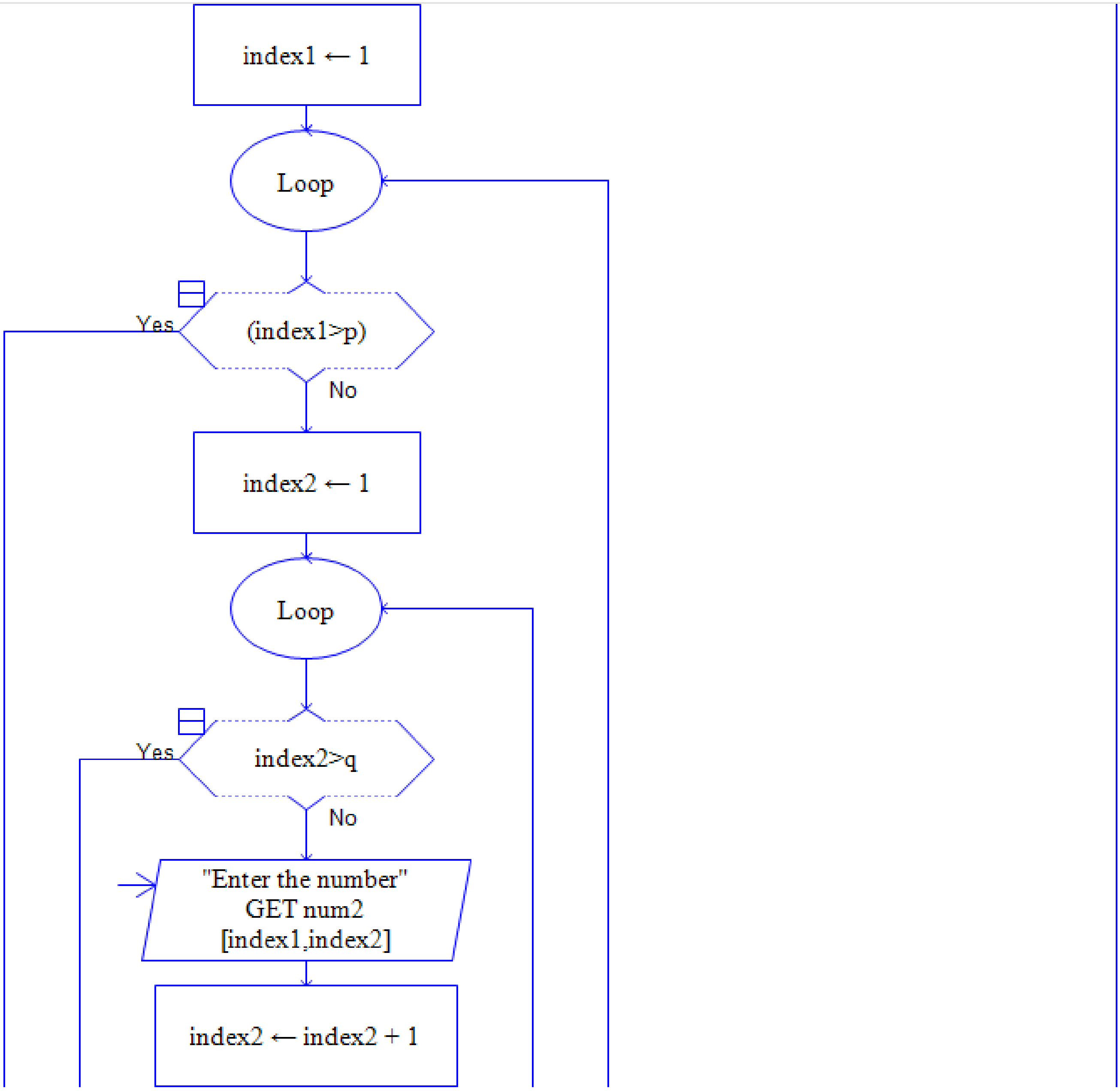
STEP 9: END.

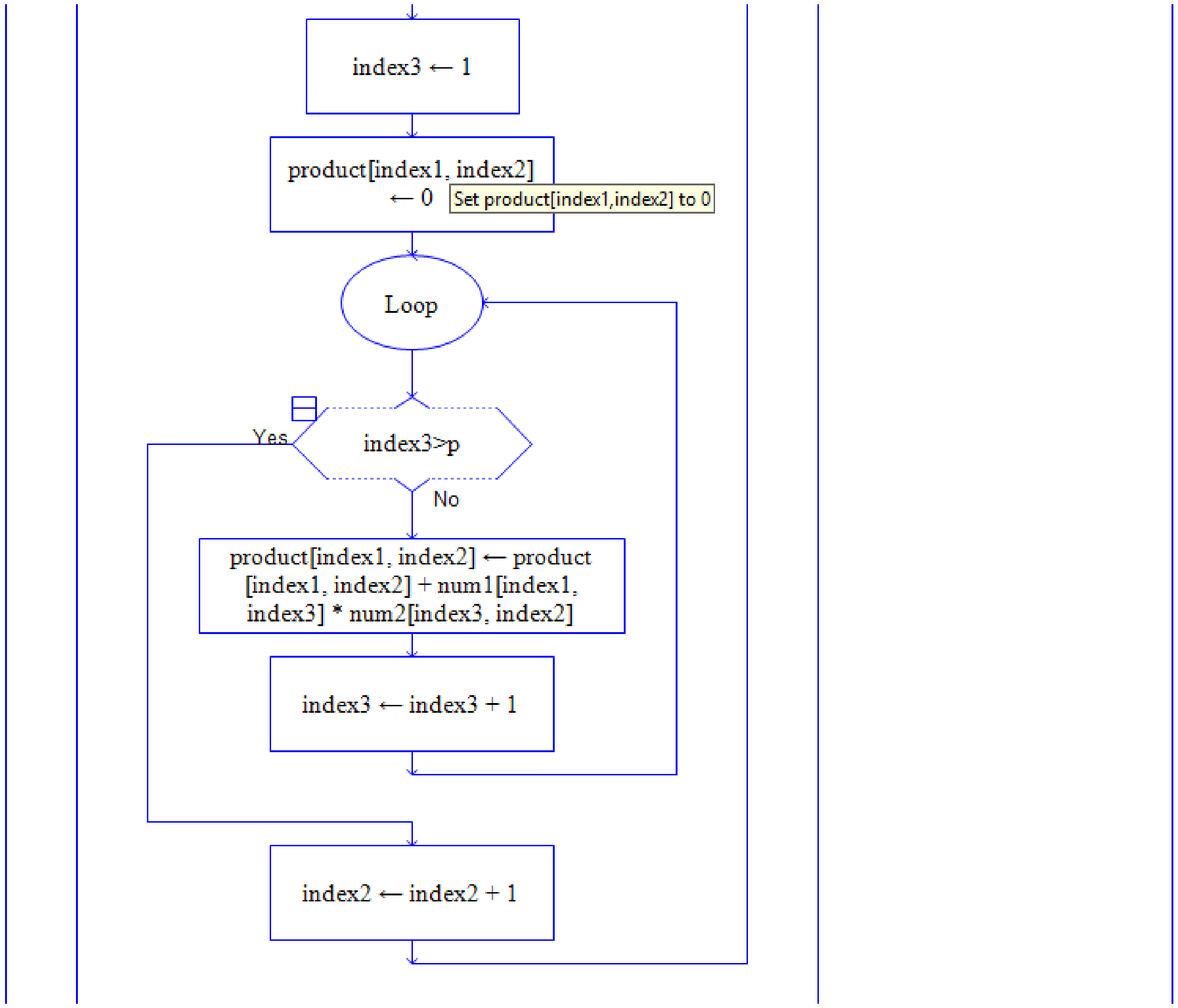
FLOWCHART:

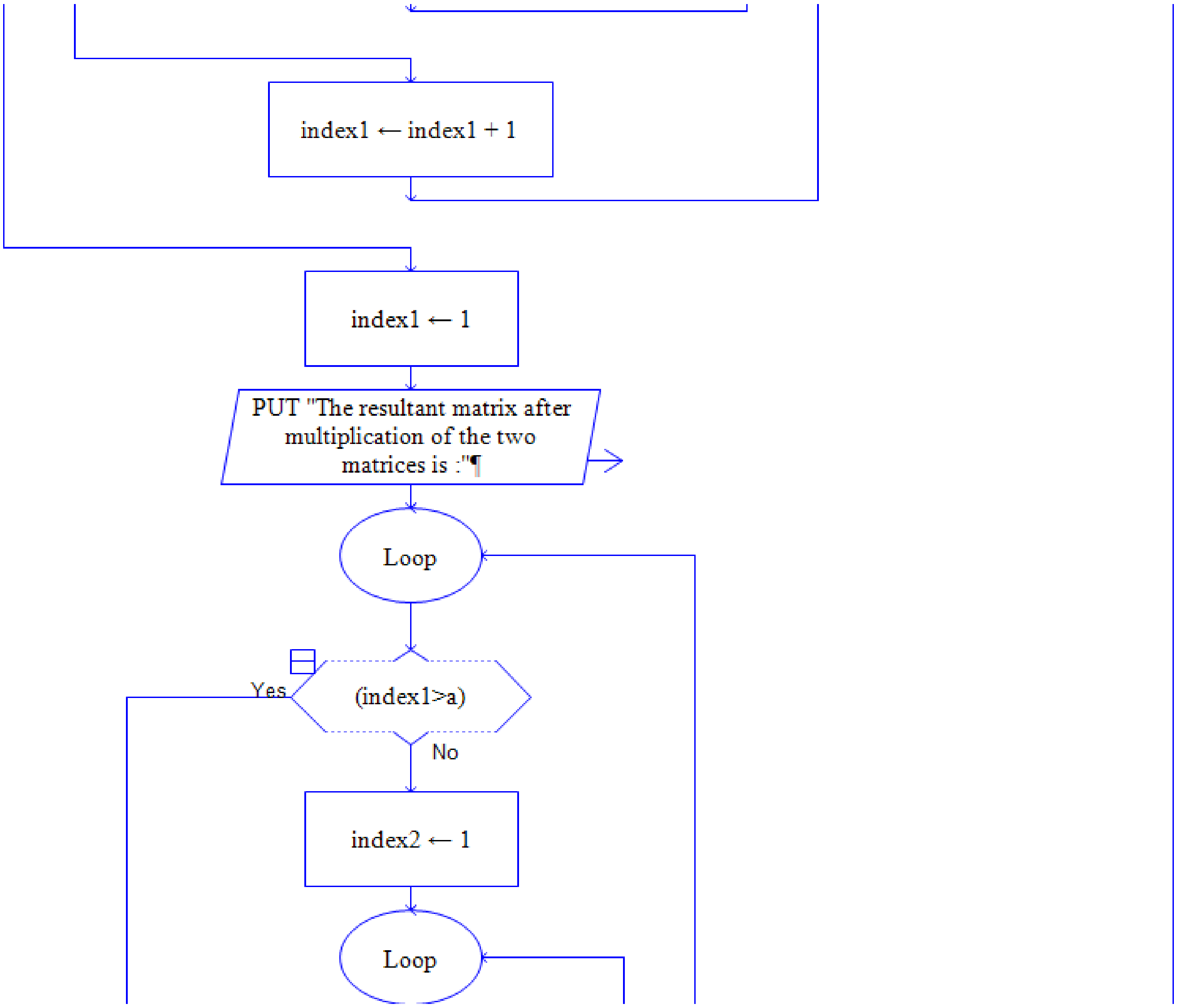

Start
"Enter the no of rows in matrix 1:"
GET a
"Enter the number of colums in the matrix 1:"
GET b
"Enter the no of rows in matrix 2:"
GET p
"Enter the number of colums in the matrix 2:"
GET q
Yes
b==p
No
index1 ← 1
PUT "Sorry the matrices with the given order does not satisfy the condtion of multiplication "¶
Loop

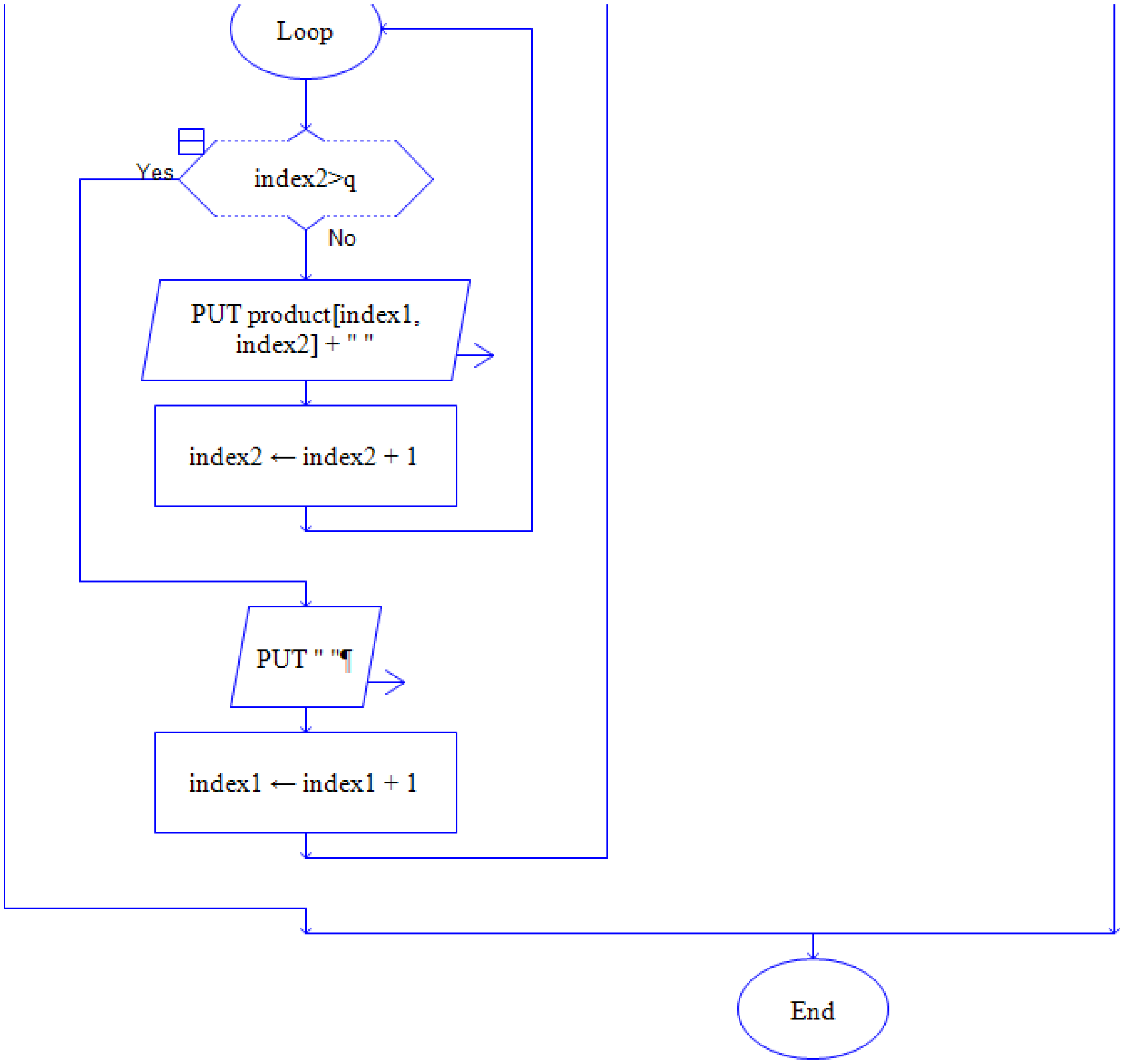

Loop
Yes
(index1>a)
No
index2 ← 1
Loop
Yes
index2>b
No
"Enter the number"
GET num1
[index1,index2]
index2 ← index2 + 1
index1 ← index1 + 1


index1 ← 1
Loop
(index1>p)
Yes
No
index2 ← 1
Loop
index2>q
Yes
No
"Enter the number"
GET num2
[index1,index2]
index2 ← index2 + 1


index1 ← 1
Loop
Yes
(index1>p)
No
index2 ← 1
Loop
Yes
index2>q
No
"Enter the number"
GET num2
[index1,index2]
index2 ← index2 + 1


index3 ← 1
product[index1, index2] ← 0
Set product[index1,index2] to 0
Loop
index3>p
Yes
No
product[index1, index2] ← product [index1, index2] + num1[index1, index3] * num2[index3, index2]
index3 ← index3 + 1
index2 ← index2 + 1


index1 ← index1 + 1
index1 ← 1
PUT "The resultant matrix after multiplication of the two matrices is :"¶
Loop
Yes
(index1>a)
No
index2 ← 1
Loop



PROGRAM:

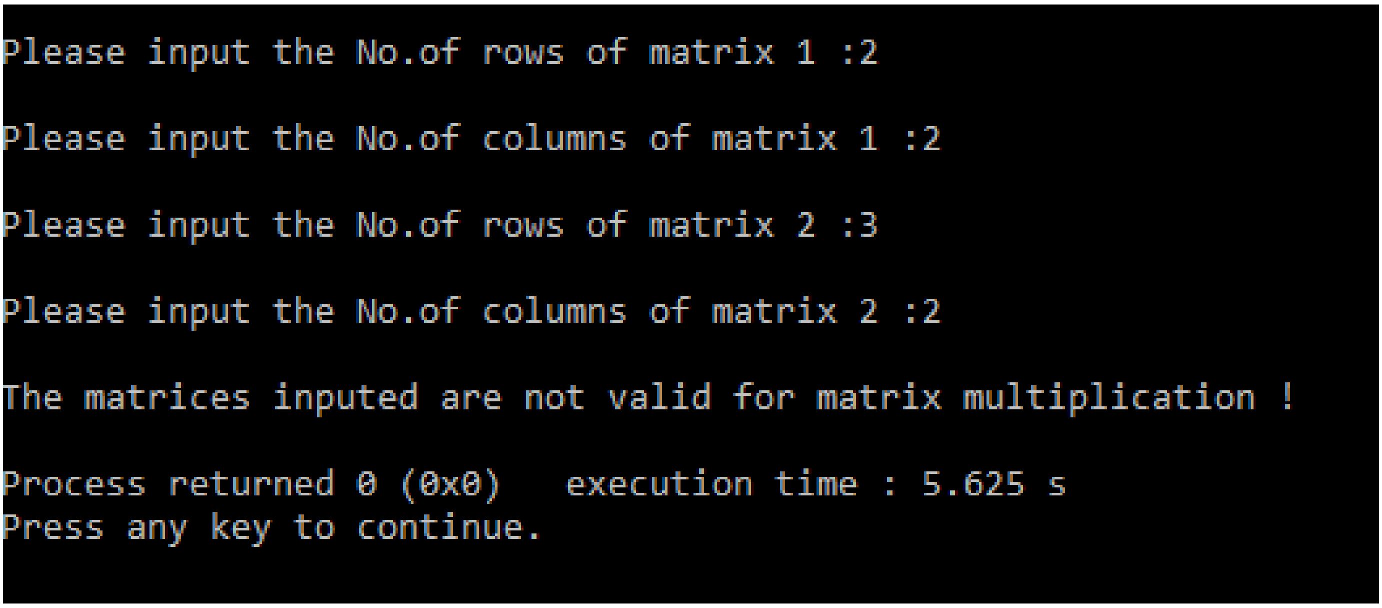
```
#include <stdio.h>

void main()

{

  int a[20][20],b[20][20],c[20][20],i,j,x,y,p,q,k;

  printf("\nPlease input the No.of rows of matrix 1 :");

  scanf("%d",&x);

  printf("\nPlease input the No.of columns of matrix 1 :");

  scanf("%d",&y);
```

```
printf("\nPlease input the No.of rows of matrix 2 :");

scanf("%d",&p);

printf("\nPlease input the No.of columns of matrix 2 :");

scanf("%d",&q);

if((x==p)&&(y==q))

{

    printf("\n Input the elements of the matrix 1\n\t:");

    for(i=0;i<x;i++)

    {

        for(j=0;j<y;j++)

        {

            scanf("%d",&a[i][j]);

        }

    }

    printf("\n Input the elements of the matrix 2\n\t:");

    for(i=0;i<x;i++)

    {

        for(j=0;j<y;j++)

        {
```

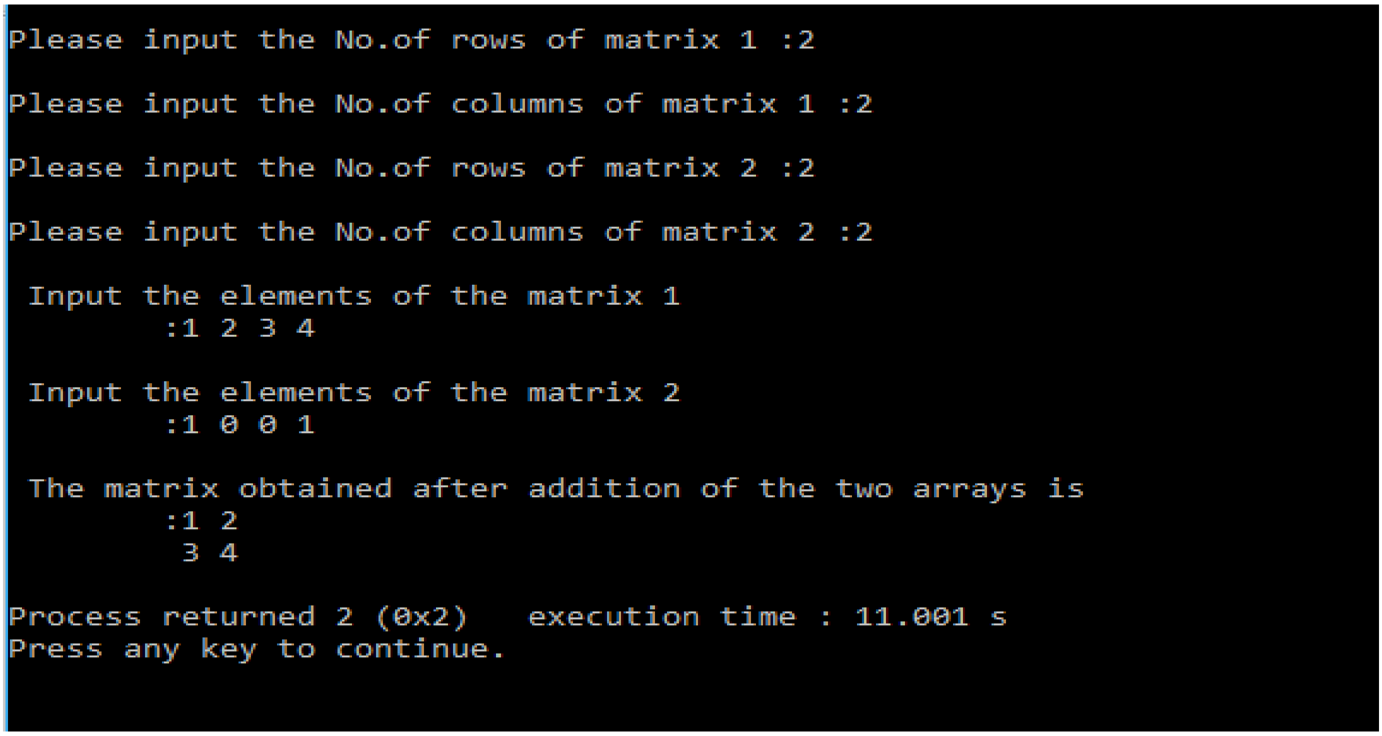
```
        scanf("%d",&b[i][j]);
    }
}
for(i=0;i<x;i++)
{
    for(j=0;j<y;j++)
    {
        c[i][j]=0;
        for(k=0;k<y;k++)
        {
            c[i][j] = c[i][j]+a[i][k]*b[k][j];
        }
    }
}
printf("\n The matrix obtained after addition of the two arrays is\n\t:");
for(i=0;i<x;i++)
{
    for(j=0;j<y;j++)
    {
        printf("%d ",c[i][j]);
```

```
          }

          printf("\n\t ");

      }

  }

  else

  {

      printf("\nThe matrices inputed are not valid for matrix multiplication
!\n");

  }

}
```

OUTPUT:



```
Please input the No.of rows of matrix 1 :2

Please input the No.of columns of matrix 1 :2

Please input the No.of rows of matrix 2 :3

Please input the No.of columns of matrix 2 :2

The matrices inputed are not valid for matrix multiplication !

Process returned 0 (0x0)   execution time : 5.625 s
Press any key to continue.
```



```
Please input the No.of rows of matrix 1 :2

Please input the No.of columns of matrix 1 :2

Please input the No.of rows of matrix 2 :2

Please input the No.of columns of matrix 2 :2

 Input the elements of the matrix 1
        :1 2 3 4

 Input the elements of the matrix 2
        :1 0 0 1

 The matrix obtained after addition of the two arrays is
        :1 2
         3 4

Process returned 2 (0x2)   execution time : 11.001 s
Press any key to continue.
```

EXPERIMENT NO 9: POINTERS

9.1)

AIM: Write a C Program to Perform Addition. Subtraction, Multiplication and Division of two numbers using Command line arguments.

DESCRIPTION:

The program developed is pre compiled in an C supported and executed using the exe file of it from the command prompt of the operating system and internal logic ensures the operations specified are performed as per the question.

ALGORITHM:

STEP 1: START.

STEP 2: in order to perform the specified operation we need to include the stdlib.h and string.h libraries in addition to the stdio.h librarary to our program.

STEP 3: in the paranthesis of the main ,method declare arg of int type to keep count of the arguments passed from the command line and character array pointer argv[] which holds and points to the argument values passed from the command line.

STEP 4: Note by default argc value exists as 1 because the default value stored in argv is the exe file of the current running program.

STEP 5: using a switch case statement as our iput consists of the operation specified in the 2nd index of the argv (argv[2][0]—ensuring only character is taken but not a string), place the case statements value with the operator as a character and the statements to be executed and output statements and break statement.

STEP6: Note in order to perform multiplication “*” has to given in input instead of * .(MS OS recognises * as a different command by default). Rest of them works fine.

STEP 7: in order to convert the numeric strings into numeric values use atoi() metho defined in stdlib.h library. And perform the operation on the numberic values.

STEP 8: numeric values can be accessed using their index in the argv.

STEP 9: our input should be in this format :

program_name num1 op num2.

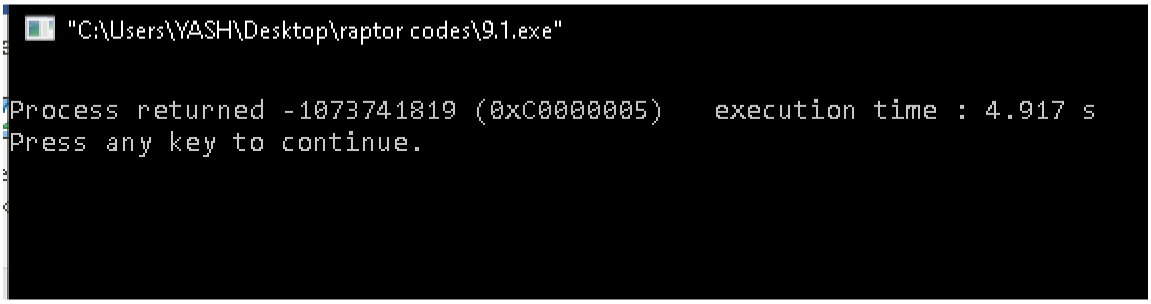
STEP 10: from the above we can see first numeric is present at index 1 and another at index 3. Pass these values to atoi to convert the string literals into the numeric data type. Ex: atoi(argv[1]).

Step 11: END.

PROGRAM:

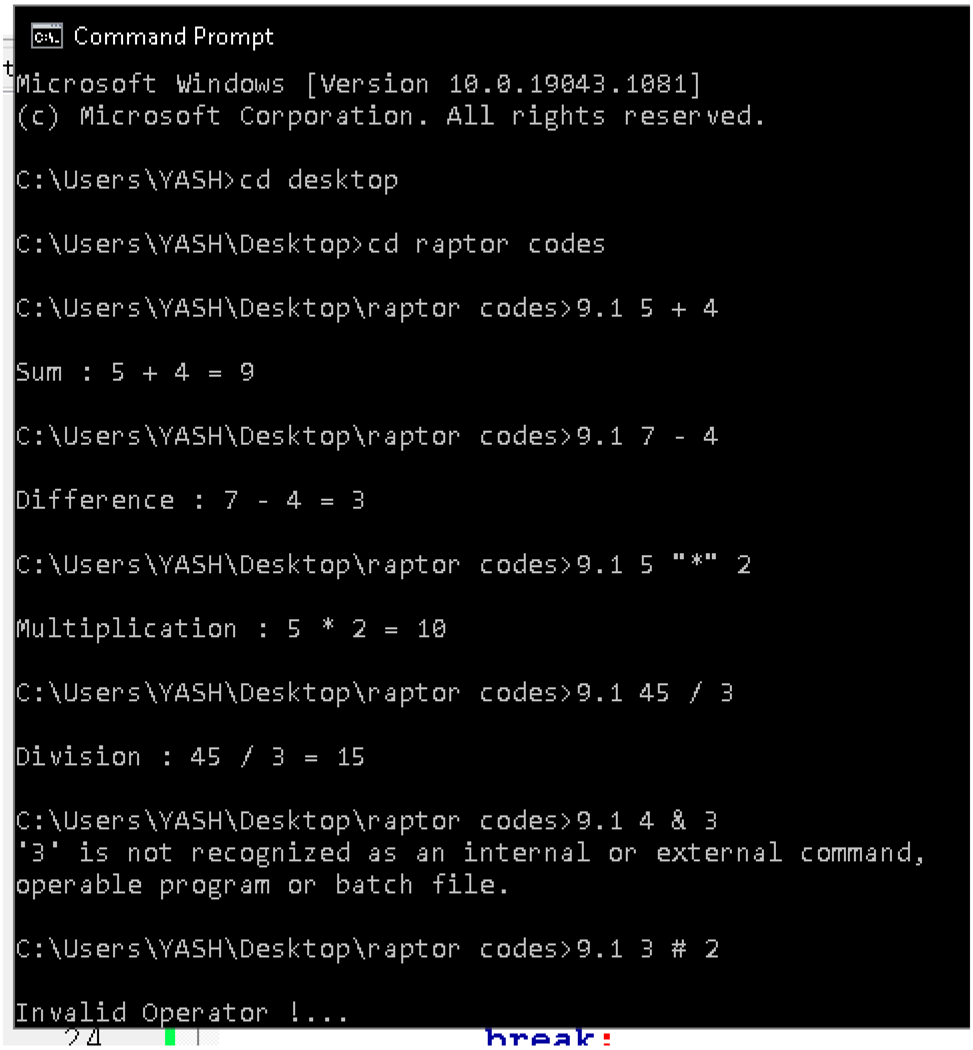
```
#include<stdio.h>

#include<stdlib.h>

#include<string.h>

void main(int argc,char *argv[])

{

   switch(argv[2][0])

   {

      case '+':
          printf("\nSum : %s + %s =
%d\n",argv[1],argv[3],(atoi(argv[1])+atoi(argv[3])));
          break;
       case '-':
          printf("\nDifference : %s - %s = %d\n",argv[1],argv[3],(atoi(argv[1])-
atoi(argv[3])));
          break;
       case '*':
          printf("\nMultiplication : %s * %s =
%d\n",argv[1],argv[3],(atoi(argv[1])*atoi(argv[3])));
          break;
       case '/':
          printf("\nDivision : %s / %s =
%d\n",argv[1],argv[3],(atoi(argv[1])/atoi(argv[3])));
          break;
       default:
          printf("\nInvalid Operator !...");
          break;
    }
```

OUTPUT:



"C:\Users\YASH\Desktop\raptor codes\9.1.exe"

```
Process returned -1073741819 (0xC0000005)   execution time : 4.917 s
Press any key to continue.
```



Command Prompt

```
Microsoft Windows [Version 10.0.19043.1081]
(c) Microsoft Corporation. All rights reserved.

C:\Users\YASH>cd desktop

C:\Users\YASH\Desktop>cd raptor codes

C:\Users\YASH\Desktop\raptor codes>9.1 5 + 4

Sum : 5 + 4 = 9

C:\Users\YASH\Desktop\raptor codes>9.1 7 - 4

Difference : 7 - 4 = 3

C:\Users\YASH\Desktop\raptor codes>9.1 5 "*" 2

Multiplication : 5 * 2 = 10

C:\Users\YASH\Desktop\raptor codes>9.1 45 / 3

Division : 45 / 3 = 15

C:\Users\YASH\Desktop\raptor codes>9.1 4 & 3
'3' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\YASH\Desktop\raptor codes>9.1 3 # 2

Invalid Operator !...
```

9.2)

AIM: Write a C program to find sum of n elements entered by user. To perform this program, allocate memory dynamically using malloc () function.

DESCRIPTION:

Program works by taking input for the numbers from user and allocating them in heap memory using malloc method.

Using dynamic memory allocation.

ALGORITHM:

STEP 1: START.

STEP 2: Declare variables n for no of inputs, I,‘sum’ for storing sum , intilaly initialised to 0,

A pointer of integer type *ptr.

STEP 3: using malloc method allocate memory in heap equal to sizeof(int) * number of numbers as input.

STEP 4: type caste the pointer to integer type as assign the memory address to pinter that is previously declared.

STEP 5: using a for loop iterate from i=1 to i<=n as test condition and i++ as step position.

STEP6: in the for loop using scanf the input from user into ptr variable and next update sum variable with sum + value inside ptr using dereference operator (*). Update ptr to next respective memory location.

STEP 7: end the for loop

STEP 8: print sum.

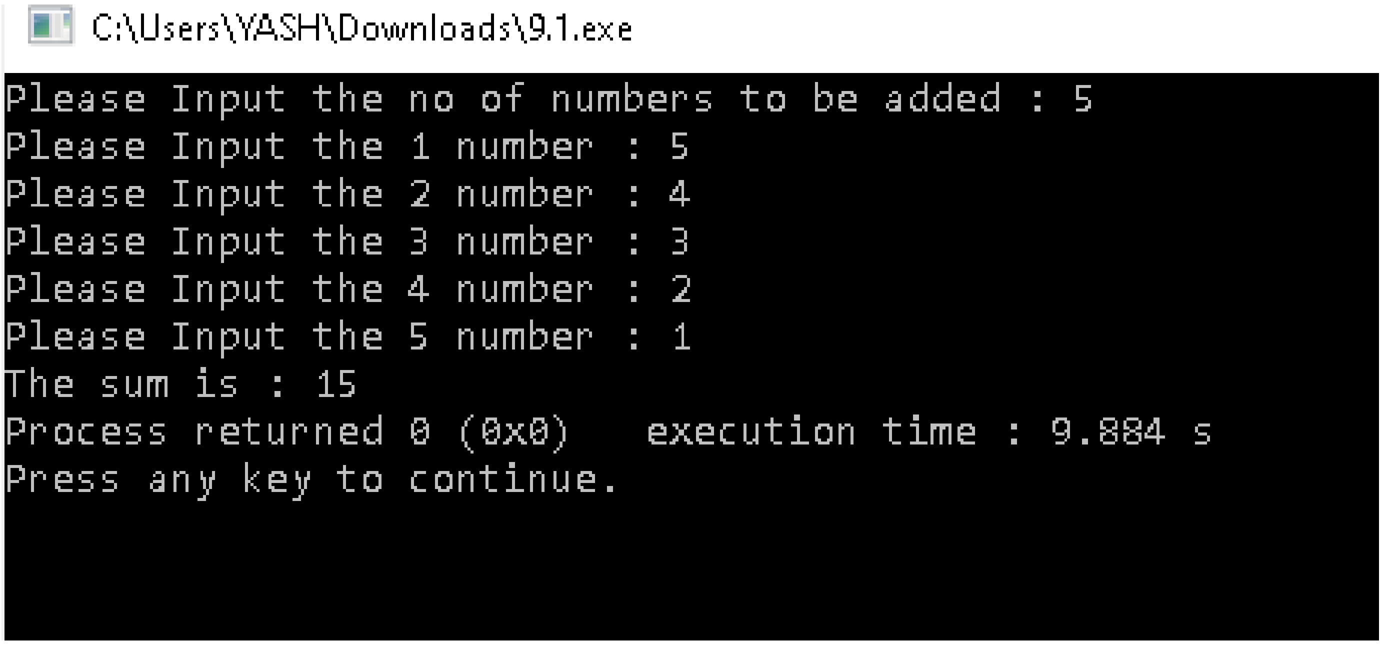
STEP 9: END.

PROGRAM:

```
#include<stdio.h>

#include<stdlib.h>

int main()

{

   int n,sum=0,*ptr,i;

   printf("Please Input the no of numbers to be added : ");

   scanf("%d",&n);

   ptr = (int*)malloc(n*sizeof(int));

   for(i=1;i<=n;i++)

   {

      printf("Please Input the %d number : ",i);

      scanf("%d",ptr);

      sum+= *ptr;

      ptr++;

   }

   printf("The sum is : %d",sum);
```

```
    return 0;

}
```

OUTPUT:



C:\Users\YASH\Downloads\9.1.exe

```
Please Input the no of numbers to be added : 5
Please Input the 1 number : 5
Please Input the 2 number : 4
Please Input the 3 number : 3
Please Input the 4 number : 2
Please Input the 5 number : 1
The sum is : 15
Process returned 0 (0x0)   execution time : 9.884 s
Press any key to continue.
```

9.3)

AIM: Write a C program to find sum of n elements entered by user. To perform this program, allocate memory dynamically using calloc () function.

DESCRIPTION:

Program works by taking input for the numbers from user and allocating them in heap memory using calloc method.

Using dynamic memory allocation.

ALGORITHM:

STEP 1: START.

STEP 2: Declare variables n for no of inputs, I,'sum' for storing sum , intilaly initialised to 0,

A pointer of integer type *ptr.

STEP 3: using calloc method allocate memory in heap equal to (n,sizeof(int)0 as argument . n = number of numbers as input.

STEP 4: type caste the pointer to integer type as assign the memory address to pinter that is previously declared.

STEP 5: using a for loop iterate from i=1 to i<=n as test condition and i++ as step position.

STEP6: in the for loop using scanf the input from user into ptr variable and next update sum variable with sum + value inside ptr using dereference operator (*). Update ptr to next respective memory location.

STEP 7: end the for loop

STEP 8: print sum.

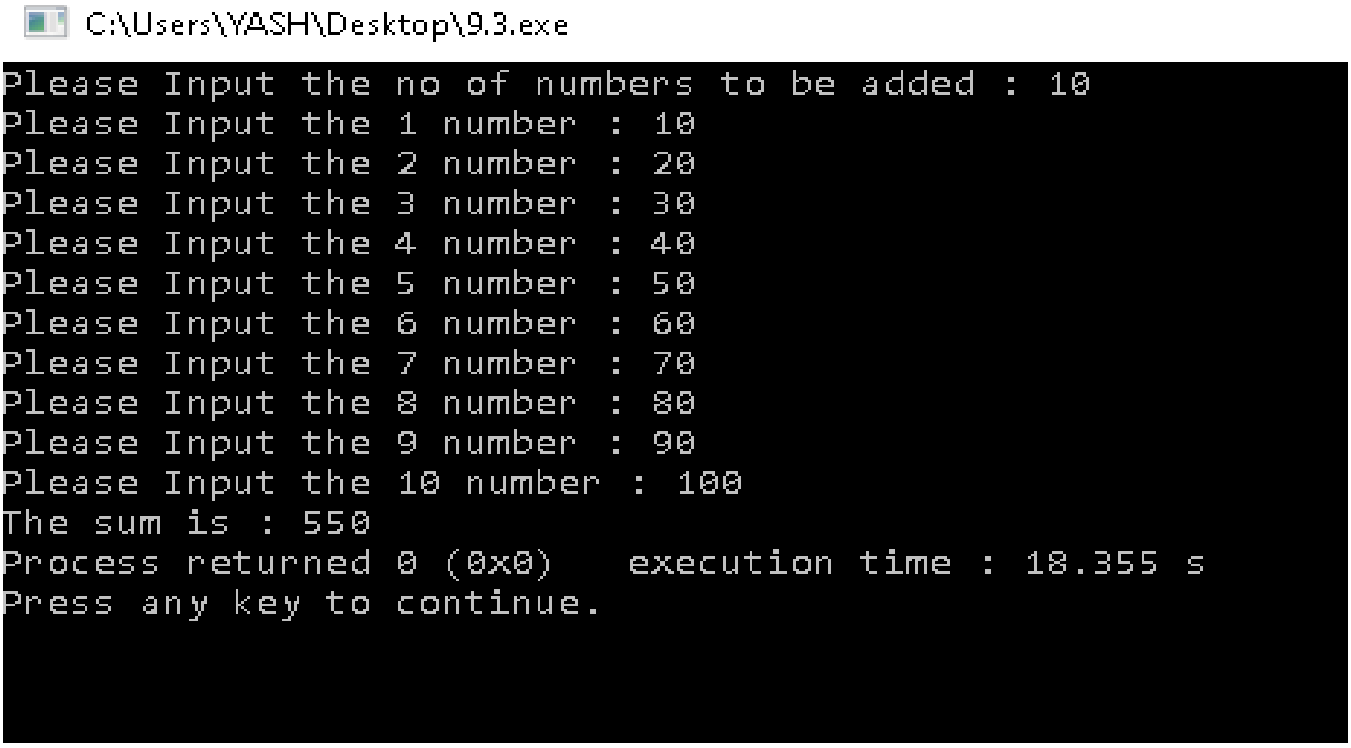
STEP 9: END.

PROGRAM:

```
#include<stdio.h>

#include<stdlib.h>

int main()

{

   int n,sum=0,*ptr,i;

   printf("Please Input the no of numbers to be added : ");

   scanf("%d",&n);

   ptr = (int*)calloc(n,sizeof(int));

   for(i=1;i<=n;i++)

   {

      printf("Please Input the %d number : ",i);

      scanf("%d",ptr);

      sum+= *ptr;
```

```
        ptr++;
    }
    printf("The sum is : %d",sum);
    return 0;
}
```

OUTPUT:



```
Please Input the no of numbers to be added : 10
Please Input the 1 number : 10
Please Input the 2 number : 20
Please Input the 3 number : 30
Please Input the 4 number : 40
Please Input the 5 number : 50
Please Input the 6 number : 60
Please Input the 7 number : 70
Please Input the 8 number : 80
Please Input the 9 number : 90
Please Input the 10 number : 100
The sum is : 550
Process returned 0 (0x0)   execution time : 18.355 s
Press any key to continue.
```

EXPERIMENT NO 10: FUNCTIONS, ARRAY & POINTERS

10.1)

AIM: Write a C Program to demonstrate parameter passing in Functions.

DESCRIPTION:

The program uses a function which was previously declared and defined.

Take two arguments and passes them to the function.

The function performs the operations and returns If any.

ALGORITHM:

STEP 1: START.

STEP 2: Read two value a and b using scanf statement.

STEP 3: declare the method for the operation liked to perform.

STEP 4: define the method with required parameters and variables.

STEP 5: In the main method call the function or prototype created.

STEP6: Pass the read values as arguments to the function.

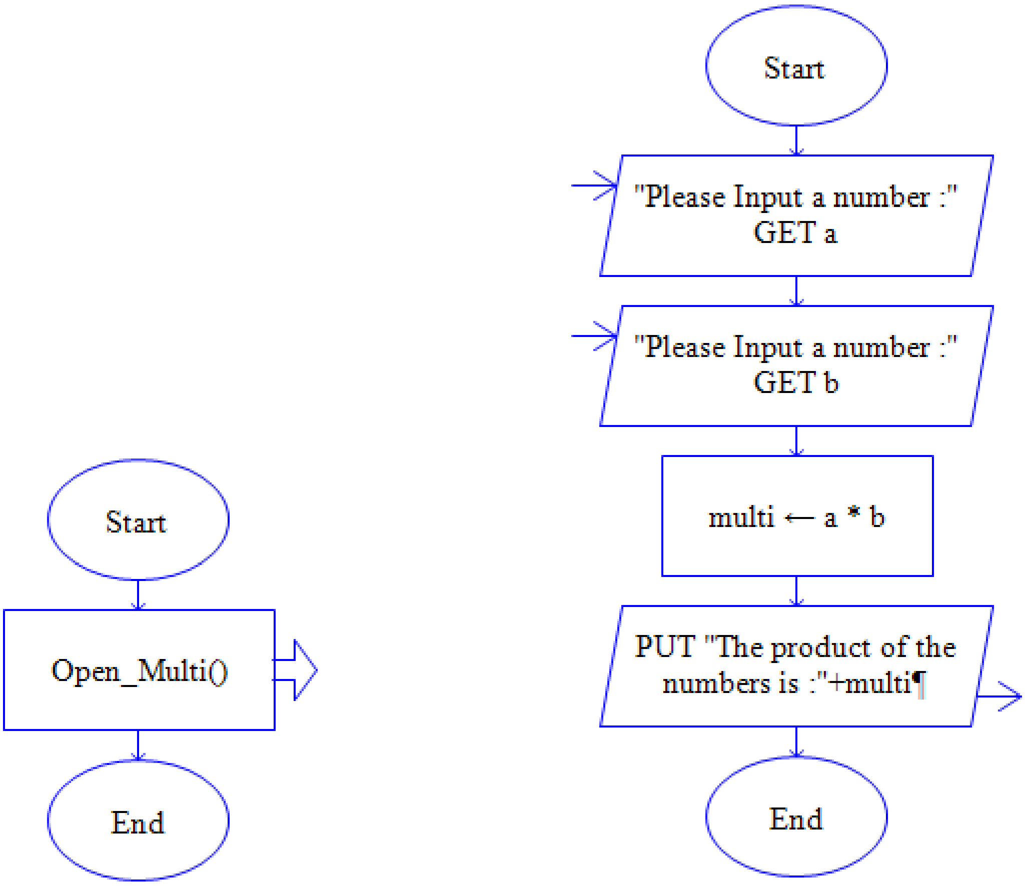
STEP 7: Using * operator perform multiplication after receiving values as parameters.

STEP 8: Display the output, using printf statement.

STEP 9: End.

Note : There are multiple ways to pass parameters to a function by values or by references and return also varies.

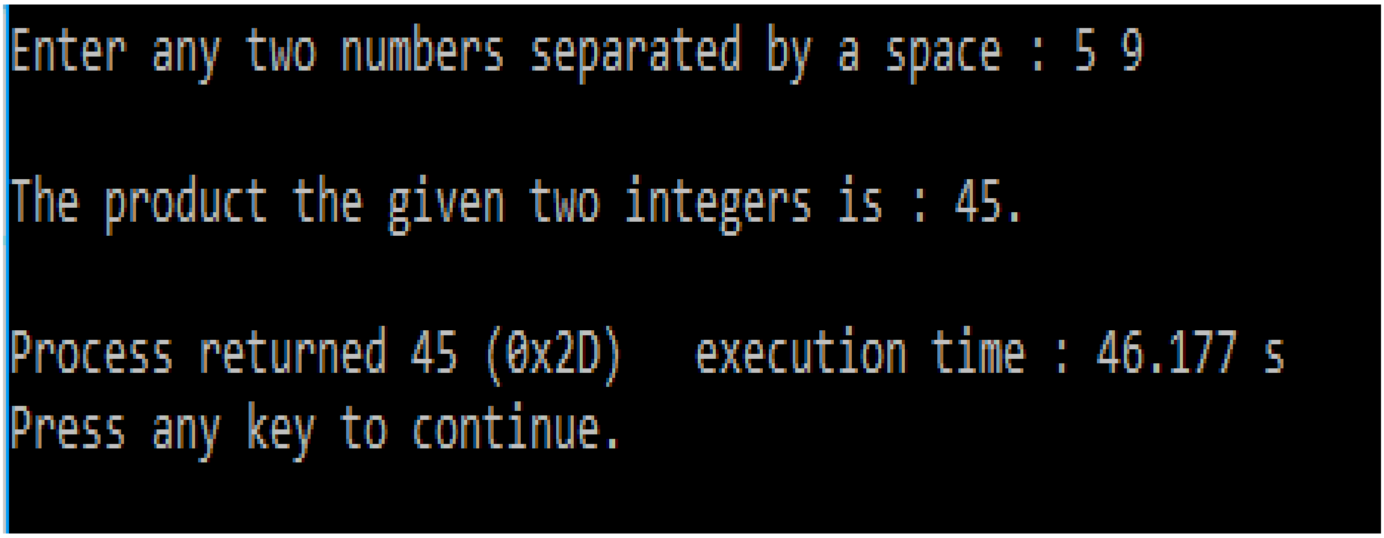
FLOWCHART:


Start
Open_Multi()
End
Start
"Please Input a number :"
GET a
"Please Input a number :"
GET b
multi ← a * b
PUT "The product of the numbers is :"+multi¶
End

PROGRAM:

```
//Demonstration of parameter passing using a FUCNTION

#include<stdio.h>

void milti(int x,int y); //declaration of a function

void main()

{
    int a,b;

    printf("\nEnter any two numbers separated by a space : ");

    scanf("%d %d",&a,&b);

    multi(a,b); // Passing a , b as arguments

}

//definition of the function multi

void multi(int x,int y) // x,y are parameters of this function

// values of a , b are passed into these parameters.

{

    int product;

    product = x*y;

    printf("\nThe product the given two integers is : %d.\n",product);

}
```

OUTPUT:



```
Enter any two numbers separated by a space : 5 9

The product the given two integers is : 45.

Process returned 45 (0x2D)   execution time : 46.177 s
Press any key to continue.
```

10.2)

AIM: Write a C Program to find Fibonacci. Factorial of a number with recursion and without recursion.

DESCRIPTION:

The code developed works by taking two numbers, As a input and finds the Fibonacci and factorial of the desired numbers by using the recursion method.

Another code is developed to do the same without using recursion.

ALGORITHM:

Part 1: (With Recursion)

```
STEP 1: START.

STEP 2: READ m

STEP 3: PASS m to fib function

STEP 4: fib(int x)

   if(x==0 or x==1)

      return x;

   else

      return fib(x-1)+fib(x-2);

STEP 5: Read n.

STEP6: Pass n to fib function.
```

```
STEP 7: fact(int n)
    if(n==0)
        return 1;
     else
       return n*fact(n-1);
STEP 8: END.
```

For part 2: (Without Recursion)

STEP 1: START.

STEP 2: Take input for the Nth Fibonacci to be found.

STEP 3: The input is passed to a function where Fibonacci is calculated.

STEP 4: if input 1 it returns 1 or if 0 it returns 0.

STEP 5: If argument is anything else Fibonacci is calculated.

STEP6: Using three variable a, b, fibb.

Initializing a=0, b=1, we calculate fibonnaci using for loop

STEP 7: start i=3 and iterate upto i<=parameter.

STEP 8: At each step make fib=a+b , a=b and b=fib.

STEP 9: when loop terminates return fib.

STEP 10: END.

STEP 1: Read umber

STEP 2: pass number as argument

STEP 3: in the function declared using a variable i initialised = parameter to >=1.

STEP 4: using fact =1, using for loop cal factorial by the logic fact *=i.

STEP 5: After completion of the loop.

STEP6: Return fact.

STEP 7: END.

PROGRAM:

Part 1: Using Recursion.

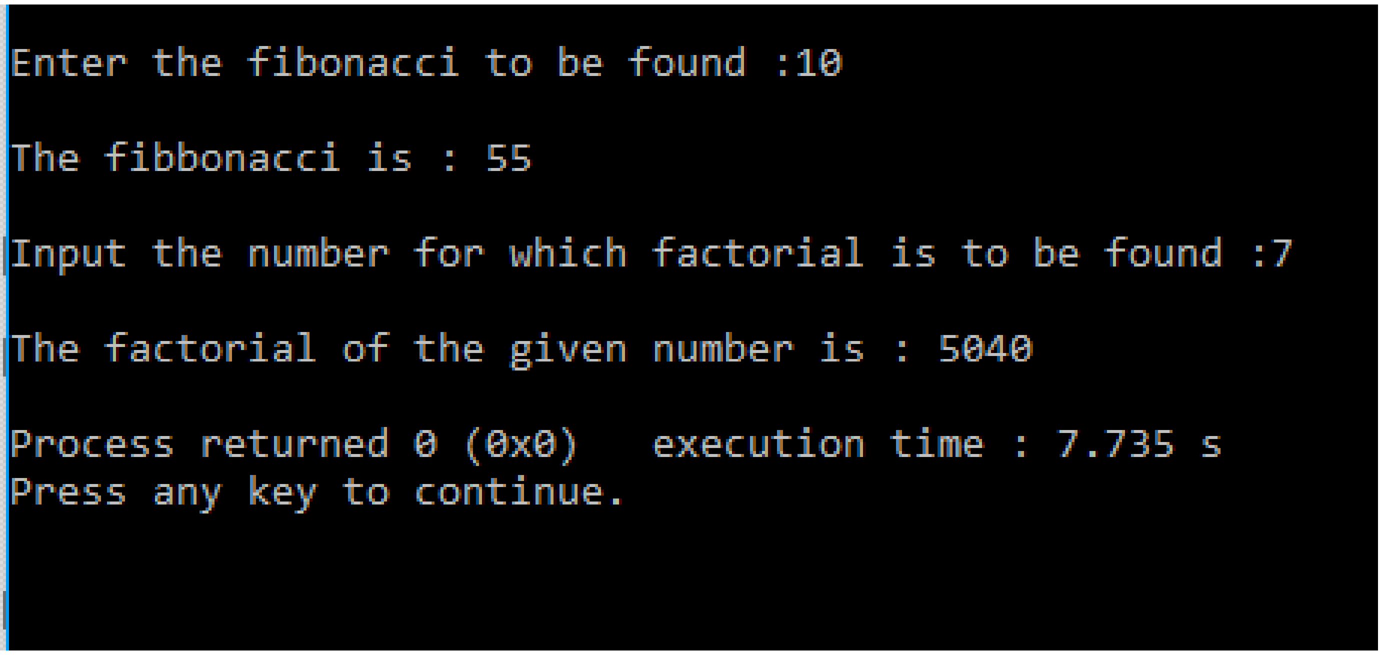
```
#include<stdio.h>
// c program for finding fibonacci , factorial of a number using recursion.
int fib(int x);
int fact(int n);
int main()
{
    int n,m;
    printf("\nEnter the fibonacci to be found :");
    scanf("%d",&n);
    printf("\nThe fibbonacci is : %d\n",fib(n));
    printf("\nInput the number for which factorial is to be found :");
    scanf("%d",&m);
    printf("\nThe factorial of the given number is : %d\n",fact(m));
    return 0;
}
int fib(int x)
{
    if(x==0||x==1)
    {
        return x;
    }
    else
    {
        return fib(x-1)+fib(x-2);
    }
}
int fact(int n)
{
    if(n==0){
        return 1;
    }
    else{
      return n*fact(n-1);
    }
}
```

Part 2:

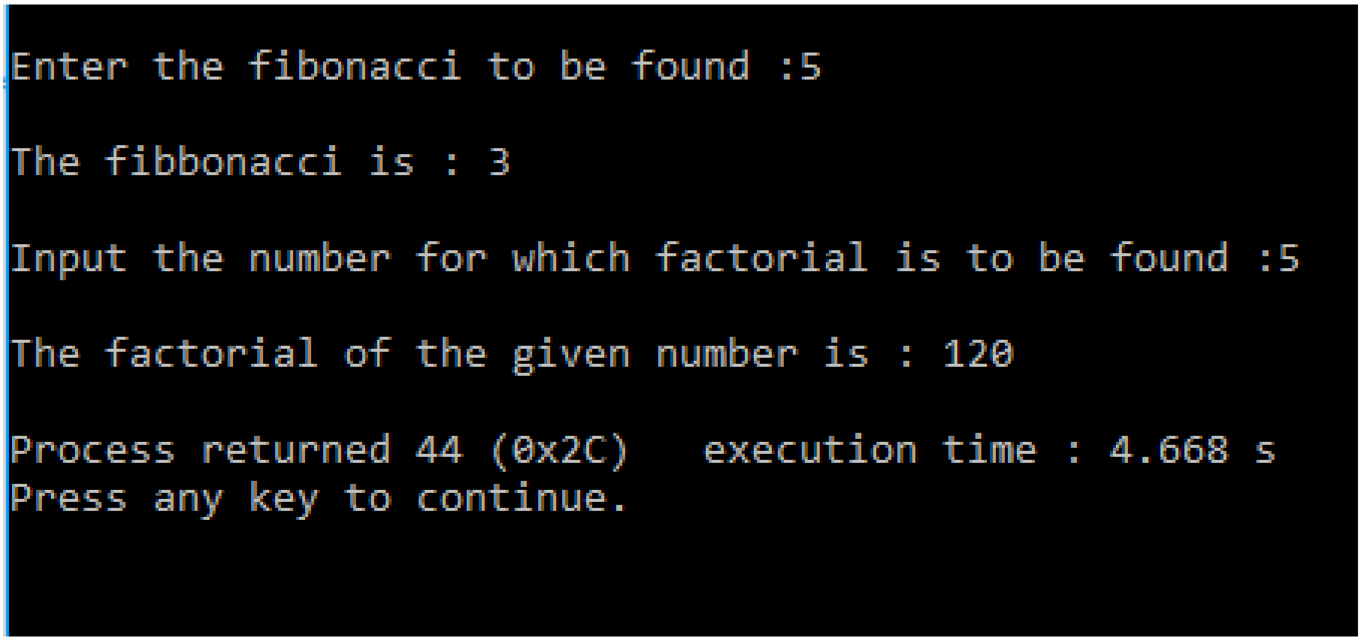
```
#include<stdio.h>
int fib(int x)
{
    int i,a,b,fibb;
    if(x==0||x==1)
    {
        return x;
    }
    else
    {   a=0;
        b=1;
        for(i=3;i<=x;i++)
        {
            fibb = a+b;
            a = b;
            b = fibb;
        }
        return fibb;
    }
}
int fact(int m)
{
    int i,fact=1;
    for(i=m;i>=1;i--)
    {
        fact = fact*i;
    }
    return fact;
}
void main()
{
     int n,m;

    printf("\nEnter the fibonacci to be found :");
    scanf("%d",&n);
```

```
   printf("\nThe fibbonacci is : %d\n",fib(n));

   printf("\nInput the number for which factorial is to be found :");

   scanf("%d",&m);

   printf("\nThe factorial of the given number is : %d\n",fact(m));

}
```

OUTPUT:



```
Enter the fibonacci to be found :10

The fibbonacci is : 55

Input the number for which factorial is to be found :7

The factorial of the given number is : 5040

Process returned 0 (0x0)   execution time : 7.735 s
Press any key to continue.
```



```
Enter the fibonacci to be found :5

The fibbonacci is : 3

Input the number for which factorial is to be found :5

The factorial of the given number is : 120

Process returned 44 (0x2C)   execution time : 4.668 s
Press any key to continue.
```

10.3)

AIM: Write a C Program to find the sum of given numbers with arrays and pointers.

DESCRIPTION:

The program takes input for the size and elements of the array from the user.

Using pointer which points to the address of the address of array's elements.

Increments it's address over a for loop after every iteration .

Stores value of sum in a vaiable and displays to the user.

ALGORITHM:

STEP 1: Start.

STEP 2: Read size of array.

STEP 3: read input for the elements of the array using for loop and scabnf statement.

STEP 4: use a pointer ptr pointing to the memory address of each elements with incrementing over the for loop.

STEP 5: using a for loop

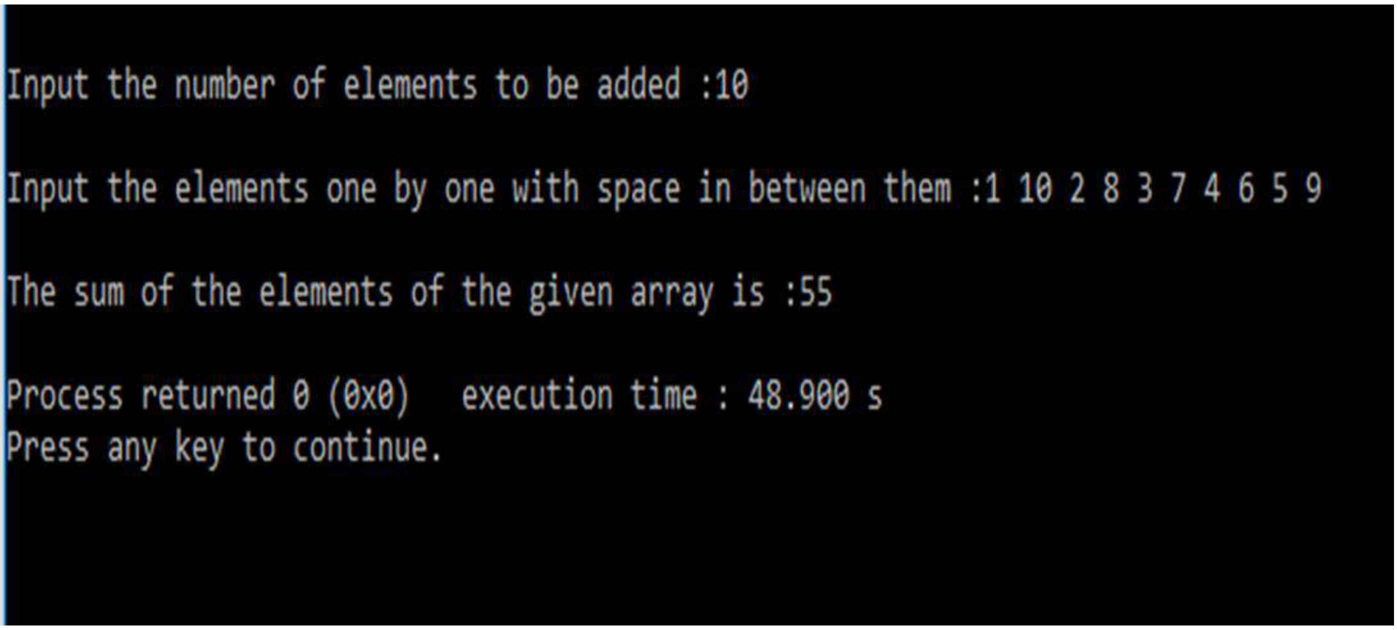
Initiliase sum variable = 0 outside and before the loop.

```
STEP6: in the loop

STEP 7:

for(ptr=&arr[0];ptr<=&arr[n-1];ptr++)

      sum+=*ptr

STEP 8: Ater all the iterations Display sum.

STEP 9: END.
```

PROGRAM:

```
#include<stdio.h>

int main()

{

   int arr[100],*ptr,n,sum=0;

   printf("\nInput the number of elements to be added :");

   scanf("%d",&n);

   printf("\nInput the elements one by one with space in between them :");

   for(int i=0;i<n;i++)

   {

      scanf("%d",&arr[i]);

   }

   for(ptr=&arr[0];ptr<=&arr[n-1];ptr++)

   {
      sum+=*ptr;

   }



   printf("\nThe sum of the elements of the given array is :%d\n",sum)

   return 0;

}
```

OUTPUT:



```
Input the number of elements to be added :10

Input the elements one by one with space in between them :1 10 2 8 3 7 4 6 5 9

The sum of the elements of the given array is :55

Process returned 0 (0x0)   execution time : 48.900 s
Press any key to continue.
```

EXPERIMENT NO 11: STRINGS

11.1)

AIM:)Implementation of string manipulation operations with library function:

a.copy

b. concatenate

c.length

d. compare

DESCRIPTION:

The program performs the operations mentioned under a,b,c,d using string operation functions strcpy, strcat, strlen, strcmp.

ALGORITHM:

STEP 1: START.

STEP 2: Include string.h library in order to perform string operations.

STEP 3: declare and define two arrays of character data type names s1 and s2 each of size 100.

STEP 4: assign values to the string arrays or take input from the user for the strings.

STEP 5: use strcpy(s3,s2) to copy string in s2 into s3.

STEP6: use strcat(s1,s2) to concatenate or to join string in s2 with string in s1.

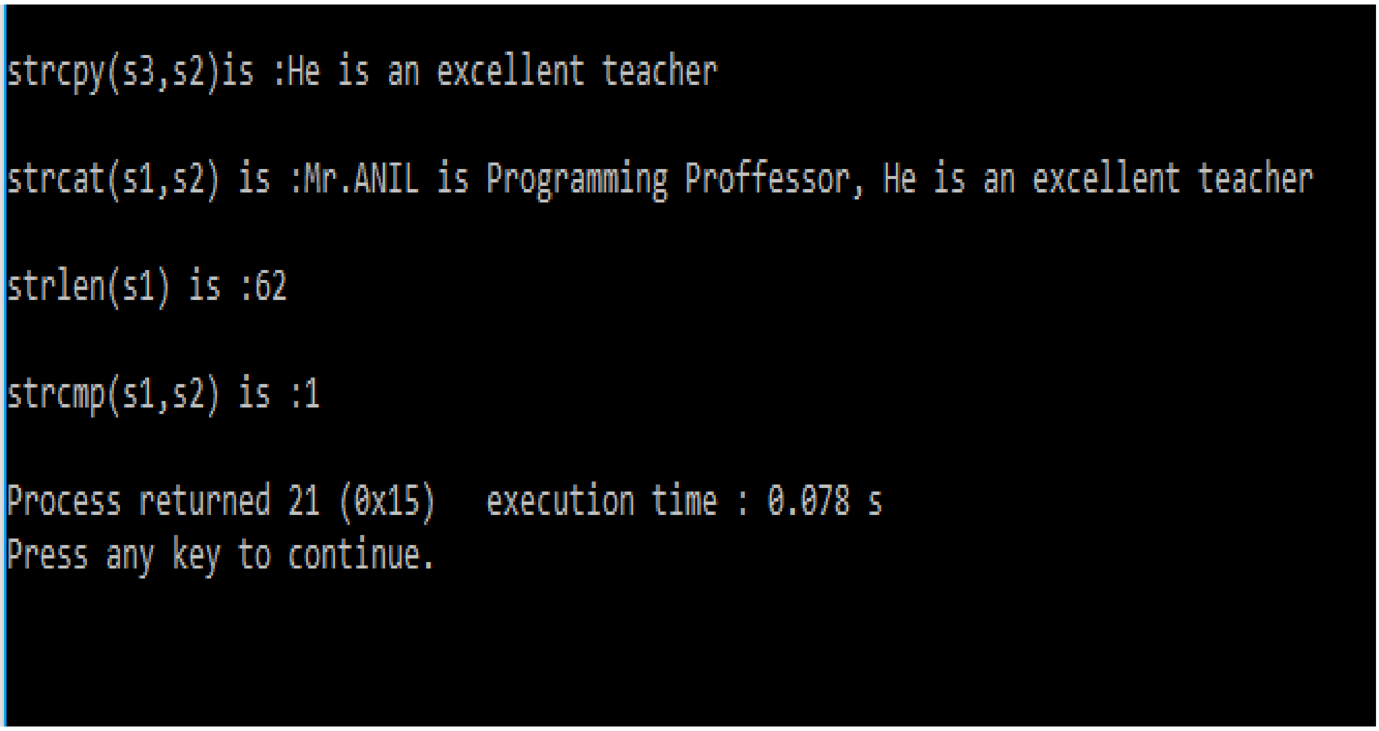
STEP 7: use strlen(s1) to find the length od string on array s1.

STEP 8: use strcmp(s1,s2) operation to compare the strings in s1 and s2 lexicographically.

PROGRAM:

```
#include<stdio.h>

#include<string.h>

 // Implementing the algorithm using library of strings



void main()

{

   char s1[100]= "Mr.ANIL is Programming Proffessor, ";

   char s2[100]= "He is an excellent teacher ";

   char s3[100];

   printf("\nstrcpy(s3,s2)is :%s\n",strcpy(s3,s2));

   printf("\nstrcat(s1,s2) is :%s\n",strcat(s1,s2));

   printf("\nstrlen(s1) is :%d\n",strlen(s1));

   printf("\nstrcmp(s1,s2) is :%d\n",strcmp(s1,s2)); // return 0 if strings are
equal

   // return 1 if s1>s2 return -1 if s1<s2

}
```

OUTPUT:



```
strcpy(s3,s2)is :He is an excellent teacher

strcat(s1,s2) is :Mr.ANIL is Programming Proffessor, He is an excellent teacher

strlen(s1) is :62

strcmp(s1,s2) is :1

Process returned 21 (0x15)   execution time : 0.078 s
Press any key to continue.
```

11.2)

AIM: Implementation of string manipulation operations without library function:

a. copy

b. concatenate

c. length

d. compare

DESCRIPTION:

The program takes input of two strings and performs the described operations without using the string libraray.

ALGORITHM:

STEP 1: START.

STEP 2: read s1 and s1 using gets method.

STEP 3: pass them to the function cps() by passing an empty string s3 and string s2 obtained from the user.

STEP 4: Display the string in s3 after copying it.

STEP 5: pass the strings s1 & s2 to the method ccs() to concatenate the 2 strings.

STEP6: display string s1.(changes due to concatenation).

STEP 7: pass the strings to method ls() to find the lengths of the two strings.

STEP 8: pass the two strings to compare them lexicographically.

STEP 9: display the result.

STEP 10: END.

Algo for the method - cps() – a) copying :

STEP 1: declare a variable i of int type.

STEP 2: using a for loop, ietrationg from i=0 to conditional statement s2[i] not equal to terminating zero. At each step increment I by 1.

STEP 3: in the body of the for loop, assign the values of s2 to the s1 using indexing method.

STEP 4: outside the for loop, i.e after completion of all the copying… terminate the copied string by adding terminating string at the index of i in that instance.

STEP 5: END.

Algo for the method - ccs() – b) concatenating :

STEP 1: declare variables i,j of int type, j = length of s1.

STEP 2: using a for loop iterate from I =0 to s2[i] not equal to terminating zero ('\0"), increment i by 1 at each step.

STEP 3: in the body of the for loop assign s2[j] with the value of s2[i] and increment j by 1.

STEP 4: after the completion of the loop terminate the string s1 by add terminating zero after it's last element i.e at at index of current instance of i.

STEP 5: END.

Algo for finding the length of a string - ls() – c)length :

STEP 1: declare variables I,length of int type.

STEP 2: initialise length = 0.

STEP 3: using a for loop iterate from i = 0 ,s[i] != '\0' as terminating condition , increment i by 1 at each step.

STEP 4: in the body of for loop , increment length by 1 at each iteration.

STEP 5: return length.

STEP6: END.

Algo for comparing the two strings - coms() – d) compare :

STEP 1: after receiving the two string as parameters, using a for loop we will compare string s1 against string s2.

STEP 2: for loop statement cosists of i declared and initialised from 0 and i<ls(s1) as a terminating condition. Incrementing I by 1 at each iteration.

STEP 3: the body of the loop consists of statements that work for comparing the strings and evaluating them if anywhere the strings are found to be lexicographically unequal.

STEP 4: if s1[i] is equal to s2[i], continue to the next iteration.

STEP 5: else

If s1[i] is less than s2[i] then print -1 indicating string s1 is less than the string s2.

And exit the method.

Else , print 1 indicating that string s1 is greater than s2 and return the function.

STEP6: end for.

STEP 7: now we add staetements to evaluate some possibilities of strings while comparing.

STEP 8: after all iterations sometimes a possibility exist for the sring s1 is equal to s2 upto the index of it's length . But not they are actually equal f the string s2 is larger in size than s1 right after the consecutive index of s1 termination and vice versa.

STEP 9: we add a if statement with the condition if length of s1 is less than that of s2 we print -1 indicating s1 is smaller than s2.

Else if after all evaluations in the for loop the strings are found to be exactly same. Then we return 0 indicating the two string are same or equal.

STEP 10: else print 1 indiating string s1 is greater than s2.

STEP 11. END.

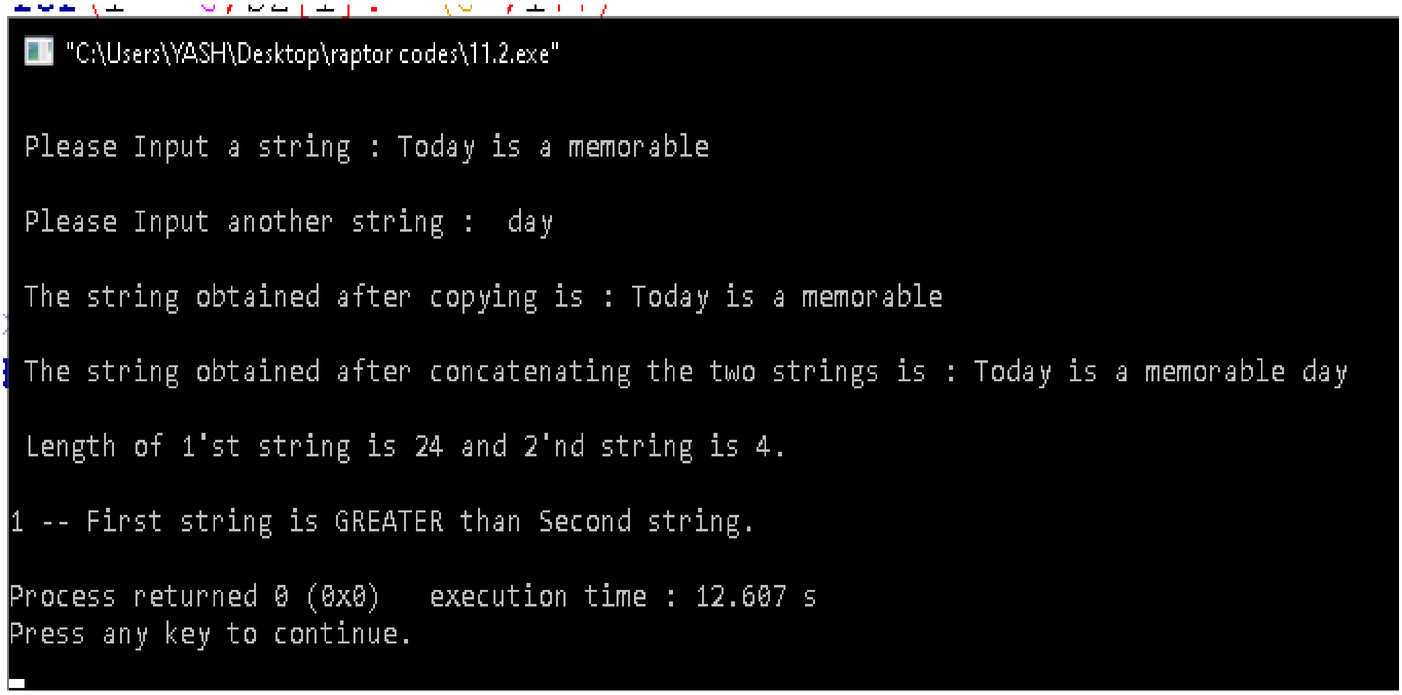
PROGRAM:

```
#include<stdio.h>

// code for finding length of a string
```

```
int ls(char s[])
{
   int i,length = 0;
   for(i = 0;s[i]!='\0';i++)
   {
      length++;
   }
   return length;
}
// code for copying one string into another string (a)
void cps(char s1[],char s2[])
{
   int i;
   for(i = 0;s2[i]!='\0';i++)
   {
      s1[i]=s2[i];
   }
   s1[i]='\0';
}
// code for concatenating 2 strings
```

```
void ccs(char s1[],char s2[])
{
    int i,j = ls(s1);
    for(i = 0;s2[i]!='\0';i++)
    {
        s1[j]=s2[i];
        j++;
    }
    s1[j]='\0';

}
// code for comparing two strings
void coms(char s1[],char s2[])
{
    for(int i = 0;i<ls(s1);i++)
    {
        if(s1[i]=s2[i])
        {
            continue;
        }
        else
        {
            if(s1[i]<s2[i])
            {
                printf("\n-1 -- First string is SMALLER than Second string. \n");
                return;
            }
            else
            {
                printf("\n1 -- First string is GREATER than Second string. \n");
                return;
            }
        }
    }
    if(ls(s1)<ls(s2))
    {
        printf("\n-1 -- First string is SMALLER than Second string. \n");
    }
    else if(ls(s1)==ls(s2))
    {
        printf("\n0 -- The Two strings are EQUAL. \n");

    }

    else

    {

      printf("\n1 -- First string is GREATER than Second string. \n");

    }

}

void main()

{

    char s1[100],s2[100],s3[100];

    printf("\n Please Input a string : ");

    gets(s1);

    printf("\n Please Input another string : ");

    gets(s2);

    //a copying

    cps(s3,s2);

    printf("\n The string obtained after copying is : %s\n",s1);



    ccs(s1,s2);
    printf("\n The string obtained after concatenating the two strings is : %s\n",s1);

    //c length

    printf("\n Length of 1'st string is %d and 2'nd string is %d.\n",ls(s1),ls(s2));

    //d comparing

    coms(s1,s2);

}
```

OUTPUT:



"C:\Users\YASH\Desktop\raptor codes\11.2.exe"

```
 Please Input a string : Today is a memorable

 Please Input another string :  day

 The string obtained after copying is : Today is a memorable

 The string obtained after concatenating the two strings is : Today is a memorable day

 Length of 1'st string is 24 and 2'nd string is 4.

1 -- First string is GREATER than Second string.

Process returned 0 (0x0)   execution time : 12.607 s
Press any key to continue.
```

EXPERIMENT NO 12: STRUCTURES

12.1)

AIM: Write a C Program to Store Information of a book Using Structure.

DESCRIPTION:

The program takes input from the user about the information and prints the data stored into the structure.

ALGORITHM:

STEP 1: START.

STEP 2: create a structure using struct keyword and bookinfo as it's rag name.

STEP 3: declare name, author, subjects arrays of char type and an id variable of int type.

STEP 4: in the main() method, declare a instance for the structure.

STEP 5: read the values into the data members of the structure.

STEP6: to access the members of a structure we use member access operator (" . ").

STEP 7: print the data read to the screen of the user.

STEP 8: END.

PROGRAM:

```
#include<stdio.h>


struct bookInfo
```

```
{
    char name[100];
    char author[100];
    char subject[100];
    int id;


};
void main()
{
    struct bookInfo book;
    //Information about the book is taken from the user
    printf("\nPlease enter the name of the book :");
    scanf("%[^\n]%*c",book.name);
    printf("\nPlease enter the name of the author :");
    scanf("%[^\n]%*c",book.author);
    printf("\nPlease enter the name of the subject :");
    scanf("%[^\n]%*c",book.subject);
    printf("\nPlease enter the book identification number :");


    scanf("%d",&book.id);
```

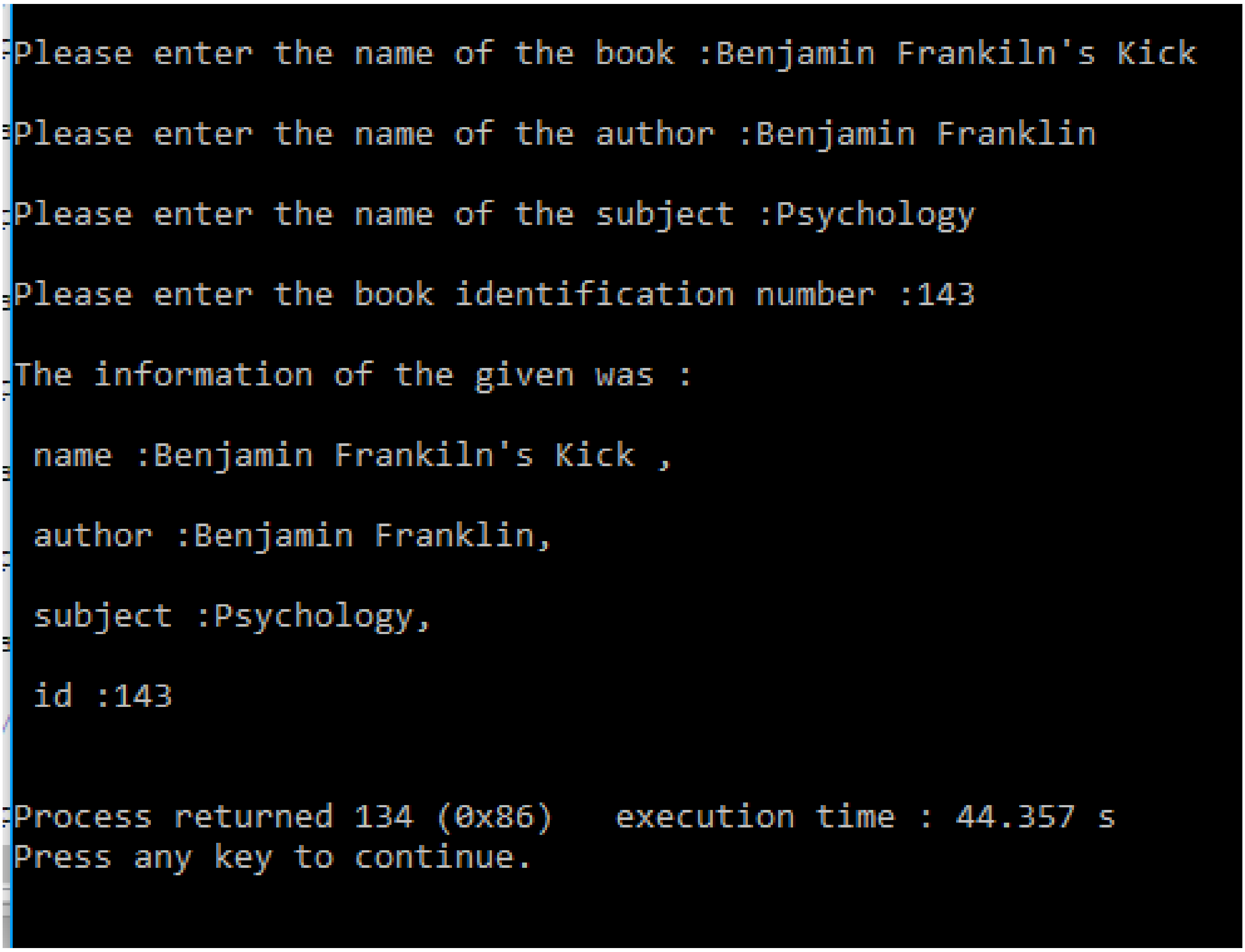
//information is now stored in the instance / variable book of structure type

printf("\nThe information of the given was :\n\n name :%s,\n\n author :%s,\n\n subject :%s,\n\n id :%d\n\n",book.name,book.author,book.subject,book.id);

//The whole code can also be written using gets,puts,fgets functions predefined in stdio.h

}

OUTPUT:



```
Please enter the name of the book :Benjamin Frankiln's Kick

Please enter the name of the author :Benjamin Franklin

Please enter the name of the subject :Psychology

Please enter the book identification number :143

The information of the given was :

 name :Benjamin Frankiln's Kick ,

 author :Benjamin Franklin,

 subject :Psychology,

 id :143


Process returned 134 (0x86)   execution time : 44.357 s
Press any key to continue.
```

12.2)

AIM: Write a C Program to Add Two Complex Numbers by Passing Structure to a Function.

DESCRIPTION:

Program takes input from the user and stores into variables created with required structure . A method is defined to find the sum of the two numbers and display to the user.

ALGORITHM:

```
STEP 1:  START.

STEP 2: declare a structure ComplexNo as tag name with it's data members real and imag of int type.

STEP 3: In main() method create instances cno1, cno2, and result of ComplexNo struct type.

STEP 4: read values into cno1 and cno2.

STEP 5: pass cno1 and cno2 to the function addcom() having cno x and cno y as it's parameters.

STEP6: declare an instance add of cno type.

STEP 7: assign sum of values in real data member of the two cno's passed to the real variable of add variable.

STEP 8: assign sum of values in imag data member of the two cno's passed to the imag variable of add variable.

STEP 9: return add and print it in the main method.
```

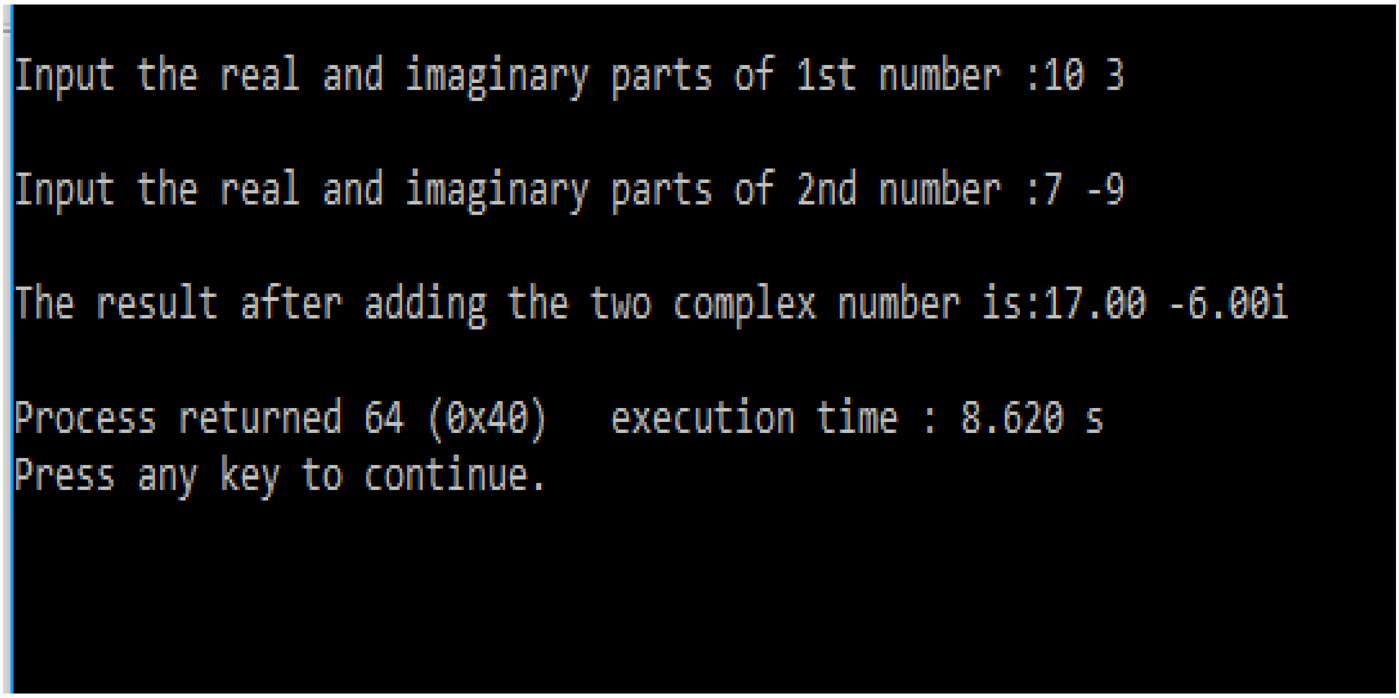
STEP 10: END.

PROGRAM:

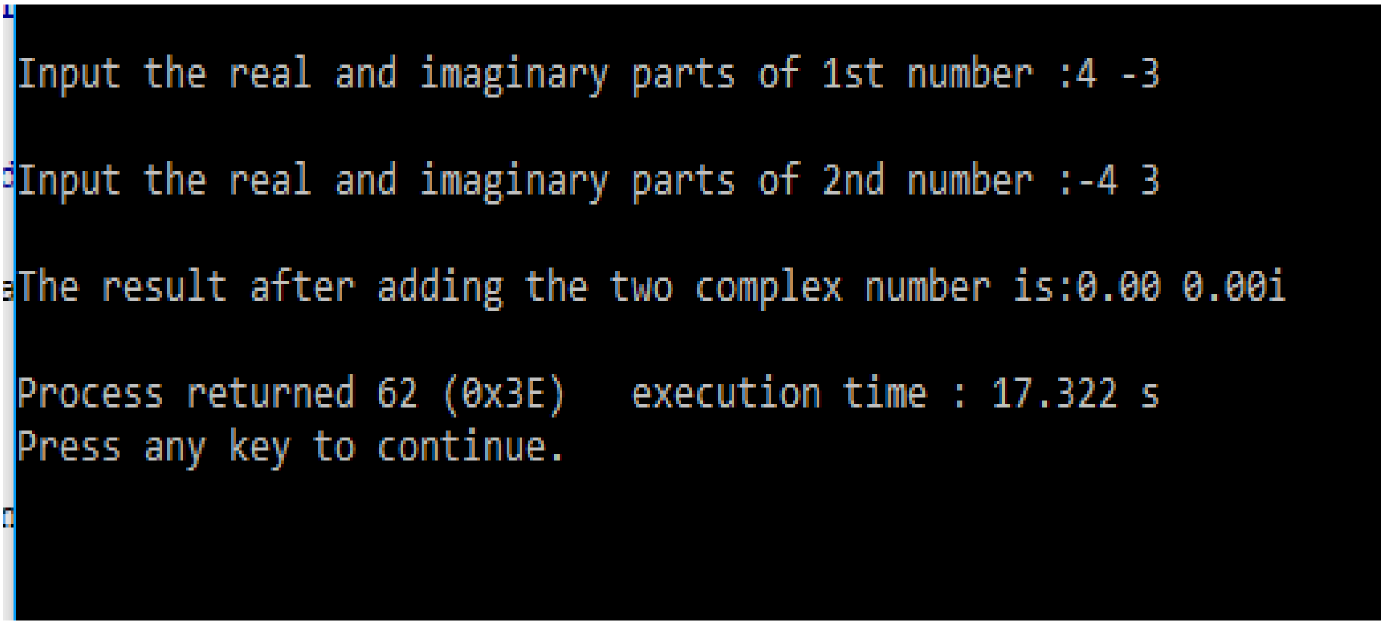
```
#include<stdio.h>
struct ComplexNo
{
    float real;
    float imag;
};
typedef struct ComplexNo cno;
cno addcom(cno x,cno y);
void main()
{
   cno cno1,cno2,result;
   printf("\nInput the real and imaginary parts of 1st number :");
   scanf("%f %f",&cno1.real,&cno1.imag);
   printf("\nInput the real and imaginary parts of 2nd number :");
   scanf("%f %f",&cno2.real,&cno2.imag);
  result = addcom(cno1,cno2);
```

```
    printf("\nThe result after adding the two complex number is:%.2f
%.2fi\n",result.real,result.imag);

}

cno addcom(cno x,cno y)

{

    cno add;

    add.real = x.real+y.real;

    add.imag = x.imag+y.imag;

    return add;

}
```

OUTPUT:



```
Input the real and imaginary parts of 1st number :10 3

Input the real and imaginary parts of 2nd number :7 -9

The result after adding the two complex number is:17.00 -6.00i

Process returned 64 (0x40)   execution time : 8.620 s
Press any key to continue.
```



```
Input the real and imaginary parts of 1st number :4 -3

Input the real and imaginary parts of 2nd number :-4 3

The result after adding the two complex number is:0.00 0.00i

Process returned 62 (0x3E)   execution time : 17.322 s
Press any key to continue.
```

EXPERIMENT NO 13: FILES

13.1)

AIM: Write a C program to open a file and to print the contents of the file on screen.

DESCRIPTION:

The program opens a file if it is previously exist in the current directory and shows the data as it is.

If the passed file doesn’t exist the program creates a new file with the same name and shows nothing on the terminal to the user.

The file created here is in the form of strucutures, so algorithm and program varies according to the purpose.

ALGORITHM:

STEP 1: START.

STEP 2: Declare a file pointer *fp of data type file and integer variable i.

STEP 3: assign the file pointer the address of the file by using fopen statement passing “file_name” and read mode “r” as arguments.

STEP 4: Using a while loop scan the values into the structures’s instance created as stud using fscanf statement.

STEP 5: in the same loop print the each value scanned from the file on the screen of the user using printf.

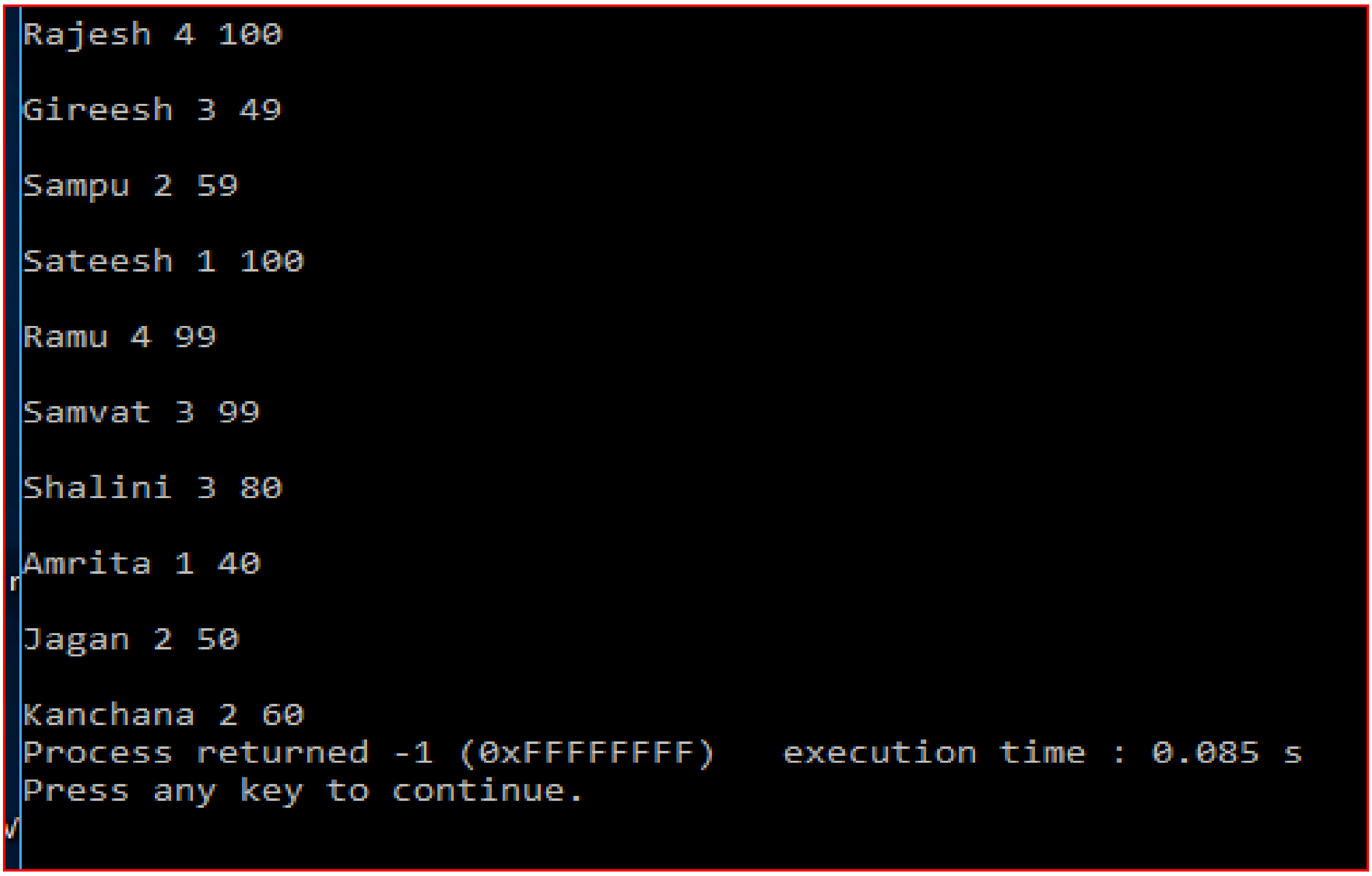
STEP6: The terminating condition for the loop is the EOF- END of the the FILE.

STEP 7: END.

PROGRAM:

```
#include<stdio.h>
struct student
{
    char name[100];
    int year,marks;
};
void main()
{
    FILE *fp;
    struct student stud;
    fp = fopen("DATA.TXT","r");
    while(fscanf(fp,"\n\n%s %d %d",&stud.name,&stud.year,&stud.marks)!=
EOF)
    {
        printf("\n\n%s %d %d",stud.name,stud.year,stud.marks);
    }
   fclose(fp);
}
```

OUTPUT:



```
Rajesh 4 100

Gireesh 3 49

Sampu 2 59

Sateesh 1 100

Ramu 4 99

Samvat 3 99

Shalini 3 80

Amrita 1 40

Jagan 2 50

Kanchana 2 60
Process returned -1 (0xFFFFFFFF)   execution time : 0.085 s
Press any key to continue.
```

13.2)

AIM: Write a C program to copy content of one file to another file.

DESCRIPTION:

Program works by using file functions like fscanf and fprintf to scand and print the data from one file to a new file.

ALGORITHM:

STEP 1: START.

STEP 2: Create two file pointers fp1 and fp2 pointing to two files one which already exists in read only mode and another new file or already existing file in write only mode.

STEP 3: Using a while loop iterate over the file until the end of the file(EOF) is reached.

STEP 4: scan the data according to the format it was created using fscanf.

STEP 5: in the same loop using fprintf write the data in the same format as scanned.

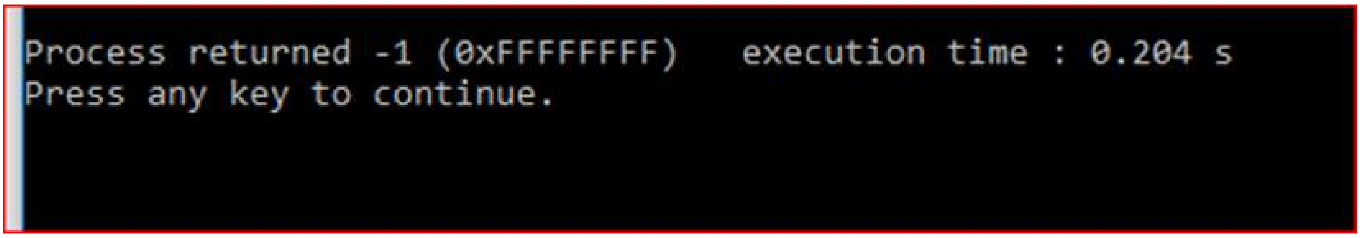
STEP6: Open the newly created file to ensure data transferred is as desired.

STEP 7: END.

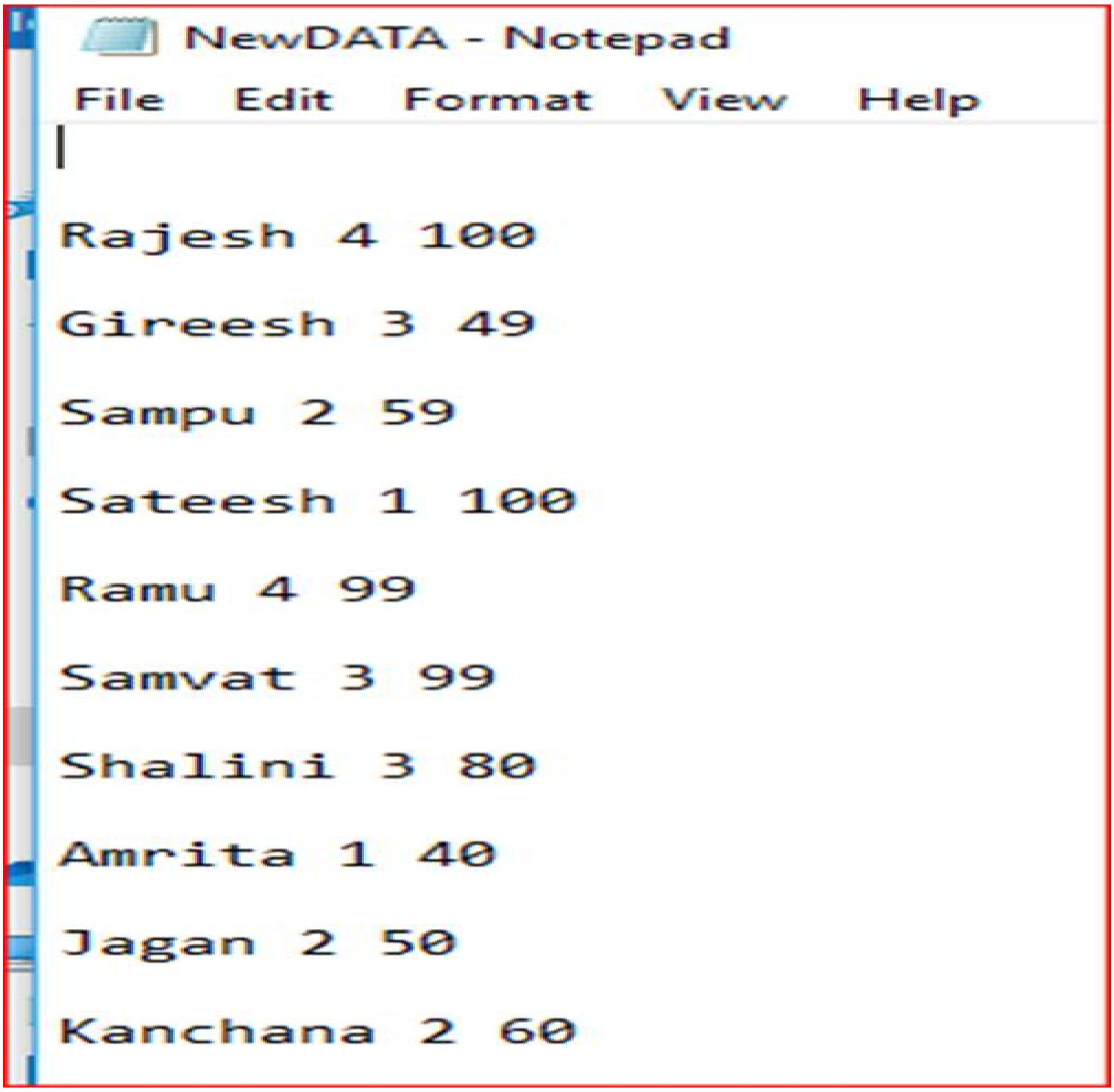
PROGRAM:

```
#include<stdio.h>

struct student

{
    char name[100];

    int year,marks;

};

void main()

{

    FILE *fp1,*fp2;

    struct student stud;

    fp1 = fopen("DATA.TXT","r");

    fp2 = fopen("NewDATA.TXT","w");

     while(fscanf(fp1,"\n\n%s %d
%d",&stud.name,&stud.year,&stud.marks)!= EOF)

    {

        fprintf(fp2,"\n\n%s %d %d",stud.name,stud.year,stud.marks);

    }

   fclose(fp1);

    fclose(fp2);

}
```

OUTPUT:



```
Process returned -1 (0xFFFFFFFF)   execution time : 0.204 s
Press any key to continue.
```



```
Rajesh 4 100
Gireesh 3 49
Sampu 2 59
Sateesh 1 100
Ramu 4 99
Samvat 3 99
Shalini 3 80
Amrita 1 40
Jagan 2 50
Kanchana 2 60
```

13.3)

AIM: Write a C program to merge two files and store content in another file.

DESCRIPTION: The program developed scans data from one file and append it to the another file (merging) and then copies the same data into another file.

ALGORITHM:

STEP 1: START.

STEP 2: using three file pointers fp1, fp2, fp3 assign the corresponding files and the required modes using fopen statement.

STEP 3: using while loop and fscanf and printf scanf data from (in read only mode) NewDATA.TXT file to DATA.TXT file.

STEP 4: using fclose function close the file which was merged (here DATA.TXT).

STEP 5: using the same fp1 open the file again in read only mode.

STEP6: again using while loop scan the data from DATA.TXT file and write into MergeDATA.TXT file which was opened in write only mode reviously.

STEP 7: Check for the data formatation is right, by opening the files.

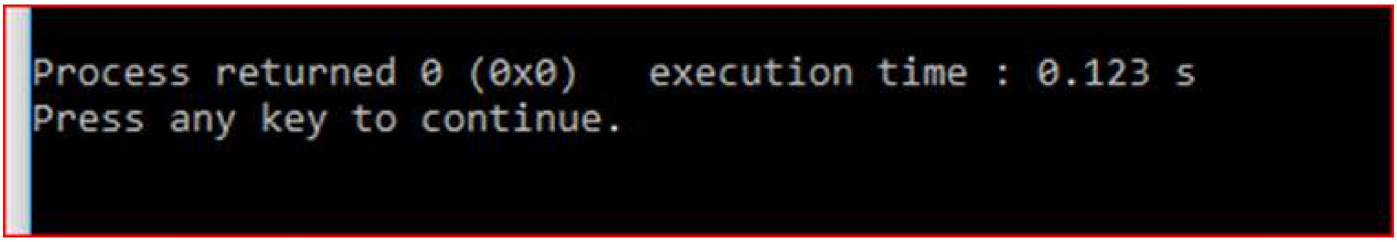
STEP 8: END.

PROGRAM:

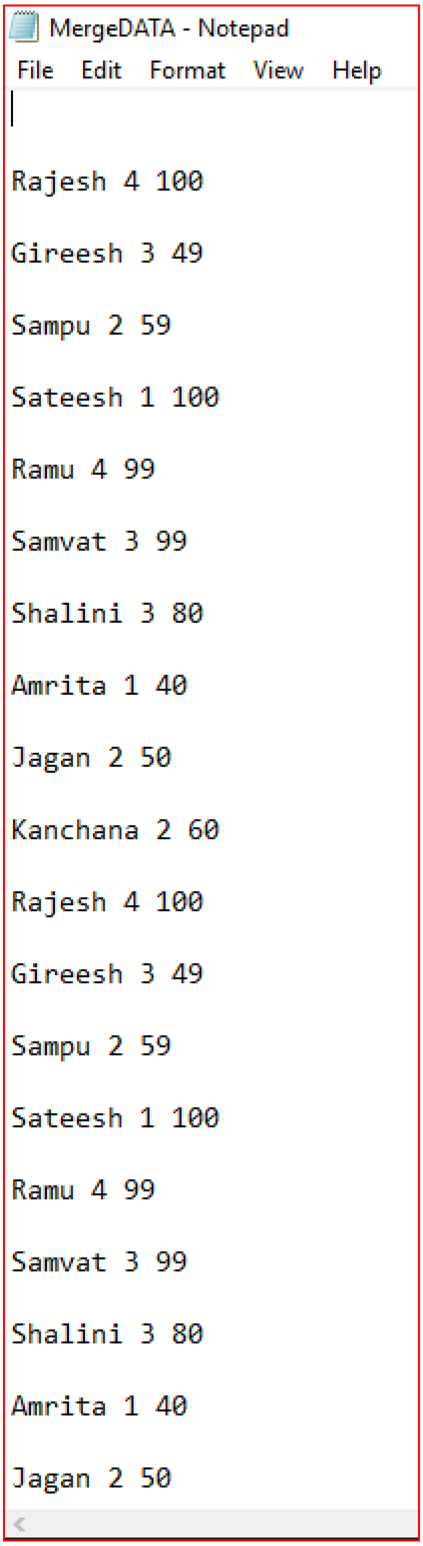
```
#include<stdio.h>
```

```
struct student
{
    char name[100];
    int year,marks;
};
void main()
{
    FILE *fp1,*fp2,*fp3;
    struct student stud;
    fp1 = fopen("DATA.TXT","a");
    fp2 = fopen("NewDATA.TXT","r");
    fp3 = fopen("MergeDATA.TXT","w");
     while(fscanf(fp2,"\n\n%s %d %d",&stud.name,&stud.year,&stud.marks)!= EOF)
    {
        fprintf(fp1,"\n\n%s %d %d",stud.name,stud.year,stud.marks);
    }
    fclose(fp1);
    fp1 = fopen("DATA.TXT","r");
    while(fscanf(fp1,"\n\n%s %d
%d",&stud.name,&stud.year,&stud.marks)!= EOF)

  {

     fprintf(fp3,"\n\n%s %d %d",stud.name,stud.year,stud.marks);

  }

  fclose(fp2);

  fclose(fp3);  }
```

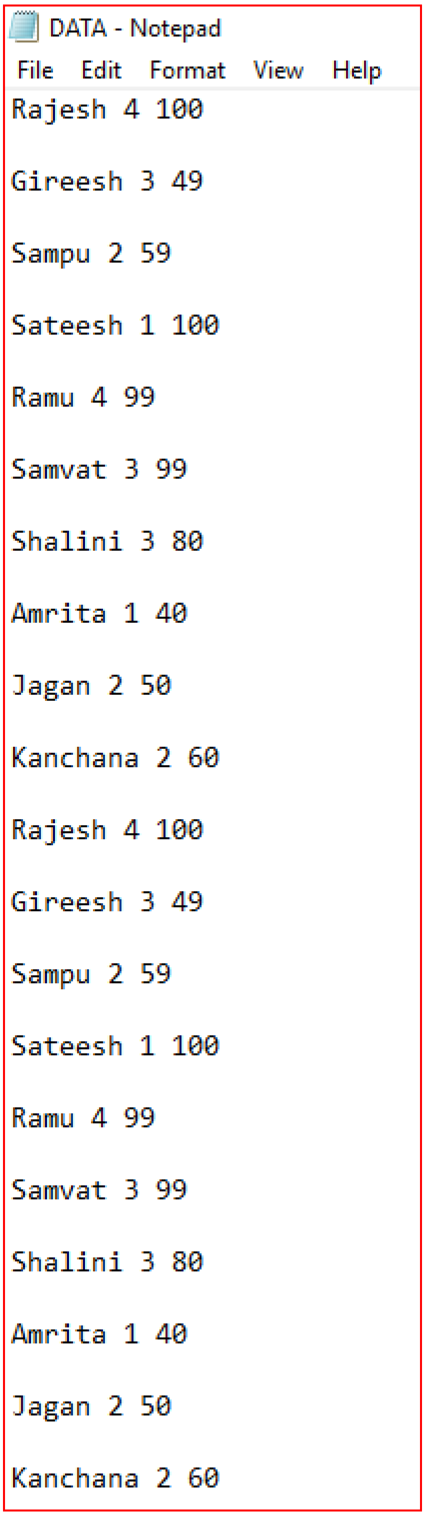
OUTPUT:



```
Process returned 0 (0x0)   execution time : 0.123 s
Press any key to continue.
```



```


Rajesh 4 100

Gireesh 3 49

Sampu 2 59

Sateesh 1 100

Ramu 4 99

Samvat 3 99

Shalini 3 80

Amrita 1 40

Jagan 2 50

Kanchana 2 60

Rajesh 4 100

Gireesh 3 49

Sampu 2 59

Sateesh 1 100

Ramu 4 99

Samvat 3 99

Shalini 3 80

Amrita 1 40

Jagan 2 50
```



```
Rajesh 4 100

Gireesh 3 49

Sampu 2 59

Sateesh 1 100

Ramu 4 99

Samvat 3 99

Shalini 3 80

Amrita 1 40

Jagan 2 50

Kanchana 2 60

Rajesh 4 100

Gireesh 3 49

Sampu 2 59

Sateesh 1 100

Ramu 4 99

Samvat 3 99

Shalini 3 80

Amrita 1 40

Jagan 2 50

Kanchana 2 60
```

EXPERIMENT NO 14: APPLICATION

14)

AIM: Creating structures to capture the student's details save them in file in proper record format. search and prints the student details requested by the user.

DESCRIPTION:

Program takes input from user about the details of a student and stores in a text file format in a proper record format and finds the data of a student by taking input of the roll number of the student and displays the details to the user.

ALGORITHM:

STEP 1: START.

STEP 2: create a structure having two character arrays skiils and name variables and roll_no variable of int type.

STEP 3: using typedef replace the struct student_info by student.

STEP 4: in the main method declare a file pointer fp of data type FILE and varbiables n,I,key of int type.

STEP 5: Delcare an array stud of size 66 and stud instance both of student type.

STEP6: read n (size of the student's class).

STEP 7: open the file with name string "ITstudDATA.txt" in write ("w") mode and pointer fp points to the opened file.

STEP 8: using a for loop read input for each data member from the user ans store in the file created using fprintf method in a proper format.

STEP 9: close the file using fclose with fp as it's argument.

STEP 10: assign i = 1.

STEP 11: while i==1 as terminating condition for the while loop . inside it's body implement the following statements and steps.

STEP 12: open the file previously closed in read ("r") mode and fp points to the that.

STEP 13: read key.

STEP 14: using a while loop, using fscanf method scan the data in the file and use the fp pointing to the end of the file (EOF) as terminating condition.

STEP 15: in the while loop if key is matched with roll of of student at a particular iteration, print the data of the student and assign I = 0 and break the while loop.

STEP 16: outside the while loop, if i==1 , it indicates that student is not found in the file of requested roll no and file is closed.

STEP 17: read input for I to continue search for another student ind=fo or to exit the program.

STEP 18: end while.

STEP 19: close the file if I is any value other than 1.

STEP 20: END.

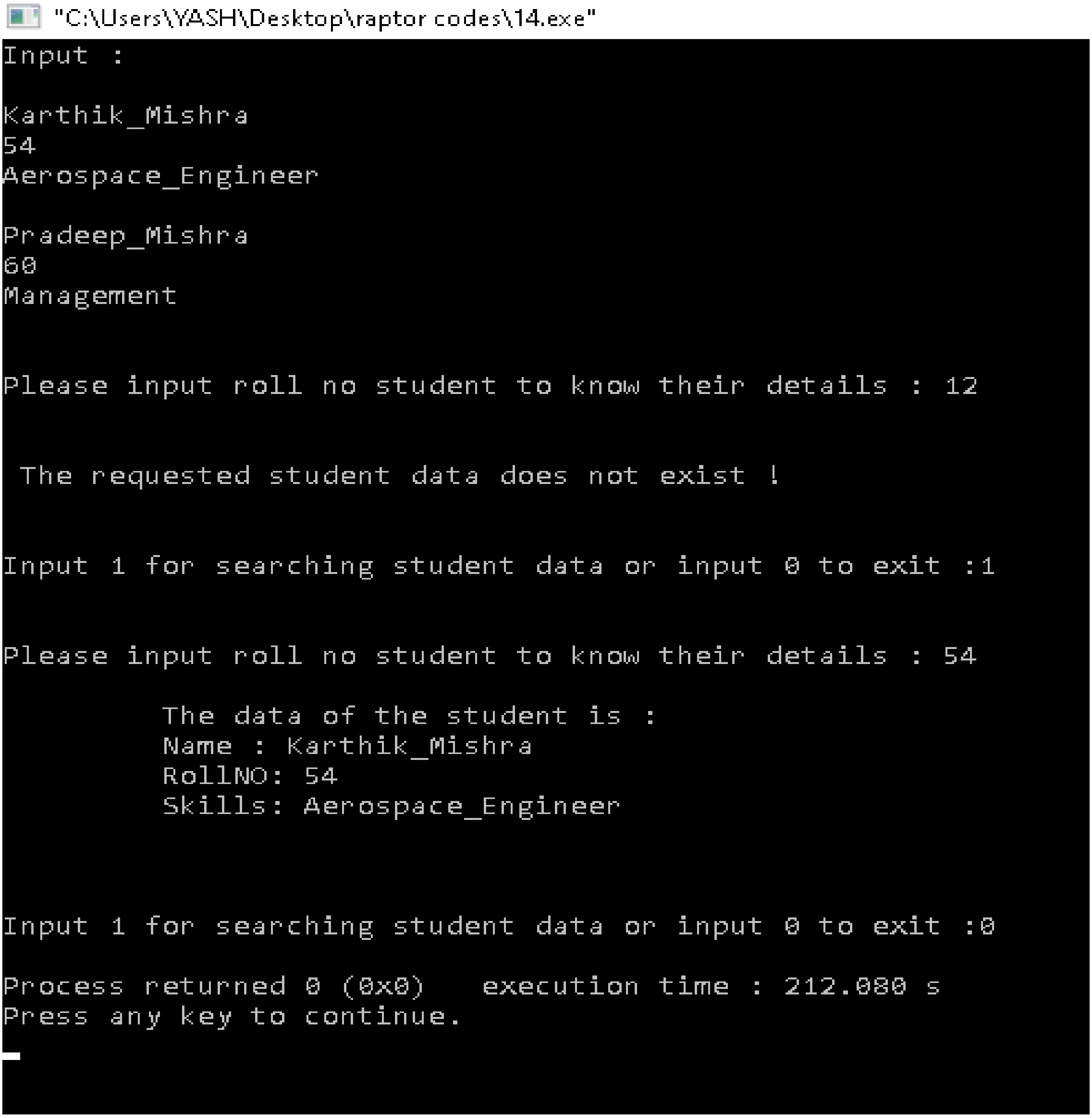
PROGRAM:

```
#include<stdio.h>
```

```
struct student_info
{
    char name[100];
    int Roll_no;
    char skills[100];
};
typedef struct student_info student;
int main()
{
   FILE *fp;
   int n,i,key;
   student stud[66];
   student std;
   printf("\n\nInput the number of students data to be added :");
   scanf("%d",&n);
   printf("\n\nInput the student details one by one as first name followed by roll_number and skills : \n");
   printf("\n\nInput values of names and skills, replace space by underscore \"_\":\n\nInput :\n ");
   fp = fopen("ITstudDATA.txt","w");
```

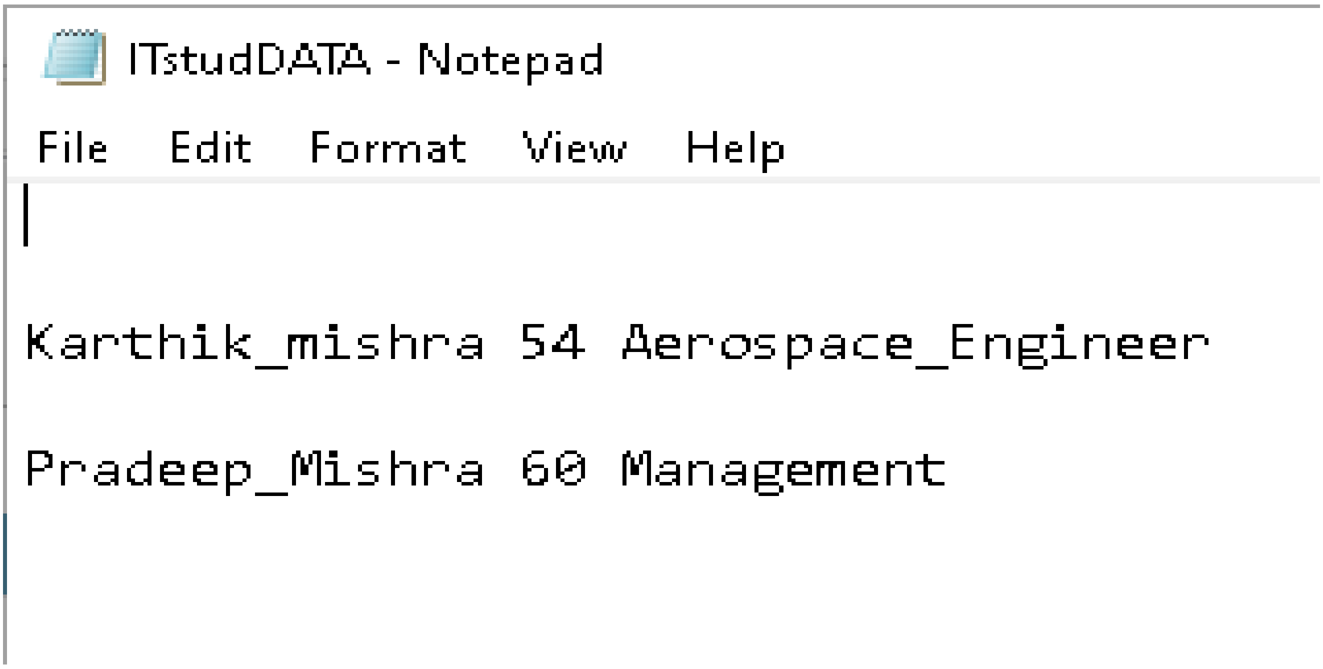
```
  for(i=0;i<n;i++)
  {
     printf("\n");
     scanf("%s",stud[i].name);
     scanf("%d",&stud[i].Roll_no);
     scanf("%s",stud[i].skills);
     fprintf(fp,"\n\n%s %d %s",stud[i].name,stud[i].Roll_no,stud[i].skills);
  }
  fclose(fp);
  i=1;
  while(i==1)
  {
      fp = fopen("ITstudDATA.txt","r");
      printf("\n\nPlease input roll no student to know their details : ");
      scanf("%d",&key);
      while(fscanf(fp,"\n\n%s %d %s",std.name,&std.Roll_no,std.skills)!=
EOF)
      {
         if(key==std.Roll_no)
      {
          printf("\n\t The data of the student is : ");

          printf("\n\t Name : %s \n\t RollNO: %d \n\t Skills:
%s\n\n",std.name,std.Roll_no,std.skills);

          i=0;

          break;

      }

      }

      if(i==1)

      {

           printf("\n\n The requested student data does not exist !\n");

           fclose(fp);

      }

      printf("\n\nInput 1 for searching student data or input 0 to exit :");

      scanf("%d",&i);//using same variable as no longer in use

  }

  fclose(fp);

  return 0;

}
```

OUTPUT:



"C:\Users\YASH\Desktop\raptor codes\14.exe"

```
Input :

Karthik_Mishra
54
Aerospace_Engineer

Pradeep_Mishra
60
Management


Please input roll no student to know their details : 12


 The requested student data does not exist !


Input 1 for searching student data or input 0 to exit :1


Please input roll no student to know their details : 54

         The data of the student is :
         Name : Karthik_Mishra
         RollNO: 54
         Skills: Aerospace_Engineer



Input 1 for searching student data or input 0 to exit :0

Process returned 0 (0x0)   execution time : 212.080 s
Press any key to continue.
```



```
Karthik_mishra 54 Aerospace_Engineer

Pradeep_Mishra 60 Management
```

THE END