



Channabasaveshwara Institute of Technology

(Affiliated to VTU, Belgaum & Approved by AICTE, New Delhi)

NH 206 (B.H. Road), Gubbi, Tumkur – 572 216. Karnataka



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

LAB MANUAL

PRINCIPLES OF PROGRAMMING USING C LABORATORY / BPOPS203

(Effective from the academic year 2022 -2023)

II SEMESTER

Prepared by,

Prof Suhas K C / Prof Kotresh Naik D

Assistant Professor

Department of Computer Science & Engineering

C I T, Gubbi - 572216



Channabasaveshwara Institute of Technology

(Affiliated to VTU, Belgaum & Approved by AICTE, New Delhi)

NH 206 (B.H. Road), Gubbi, Tumkur – 572 216. Karnataka



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

PRINCIPLES OF PROGRAMMING USING C LABORATORY

[As Per Choice Based Credit System (CBCS) System]

(Effective from the academic year 2022 -2023)

SEMESTER I/II

Subject Code	BPOPS103/203	CIE Marks	15
Number of Lecture Hours/Week	2	Lab Test	05
Total Number of Lab Hours	24	Exam Hours	3 Hrs

Credits – 1

Course Objectives :

CLO 1. Elucidate the basic architecture and functionalities of a Computer

CLO 2. Apply programming constructs of C language to solve the real-world problems

CLO 3. Explore user-defined data structures like arrays, structures and pointers in implementing solutions to problems

CLO 4. Design and Develop Solutions to problems using structured programming constructs such as functions and procedures

Descriptions (if any):

- The laboratory should be preceded or followed by a tutorial to explain the approach or algorithm being implemented for the problems given.
- Note: Two hours tutorial is suggested for each laboratory session.
- Questions related with experiment 1, need to be asked during viva-voce for all experiments.
- Every experiment should have algorithm and flowchart be written before writing the program.
- Code should be traced using minimum two test cases which should be recorded.
- It is preferred to implement using TURBO C, Linux and GCC.

Laboratory Programs:

	Familiarization with programming environment, concept of naming the program files, storing, compilation, execution and debugging. Taking any simple C- code.
1	Simulation of a Simple Calculator.
2	Compute the roots of a quadratic equation by accepting the coefficients. Print appropriate messages.
3	An electricity board charges the following rates for the use of electricity: for the first 200 units 80 paise per unit: for the next 100 units 90 paise per unit: beyond 300 units Rs 1 per unit. All users are charged a minimum of Rs. 100 as meter charge. If the total amount is more than Rs 400, then an additional surcharge of 15% of total amount is charged. Write a program to read the name of the user, number of units consumed and print out the charges.

4	<p>Write a C Program to display the following by reading the number of rows as input,</p> <pre> 1 121 12321 12 3 4 3 2 1 ----- nth row </pre>
5	Implement Binary Search on Integers.
6	Implement Matrix multiplication and validate the rules of multiplication.
7	Compute $\sin(x)/\cos(x)$ using Taylor series approximation. Compare your result with the built-in library function. Print both the results with appropriate inferences.
8	Sort the given set of N numbers using Bubble sort.
9	Write functions to implement string operations such as compare, concatenate, string length. Use the parameter passing techniques.
10	Implement structures to read, write and compute average- marks of the students, list the students scoring above and below the average marks for a class of N students.
11	Develop a program using pointers to compute the sum, mean and standard deviation of all elements stored in an array of n real numbers.
12	Write a C program to copy a text file to another, read both the input file name and target file name.

Course Outcomes: At the end of the course the student should be able to:

CO1. Elucidate the basic architecture and functionalities of a computer and also recognize the hardware parts.

CO 2. Apply programming constructs of C language to solve the real world problem.

CO 3. Explore user-defined data structures like arrays in implementing solutions to problems like searching and sorting.

CO 4. Explore user-defined data structures like structures, unions and pointers in implementing solutions.

CO5. Design and Develop Solutions to problems using modular programming constructs using functions .

INTRODUCTION

Description about Functional block diagram of Computer:

A computer is an electronic device, which mainly performs the four functions as **reading, processing, displaying** and **storing** on data. These functions of a computer system can be carried out by using the three main units namely input unit, system unit and output unit. The block diagram of a computer system is as follows:

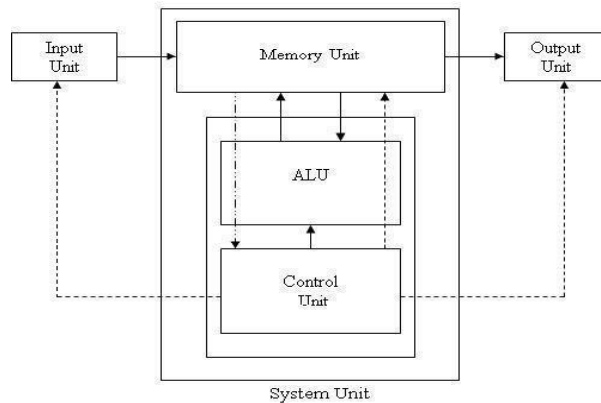


Fig 1: Block Diagram of a Computer

Notations:

- Data and Results flow
- - - Control instructions to other units from control unit
- - - Instructions from memory unit to control unit

System or Central Processing Unit (CPU): is commonly known as “processor” that executes the instructions of a computer program. It has Control Unit (CU) and Arithmetic & Logical Unit (ALU). These two units perform the basic arithmetic, logical, and input/output operations.

- a) Input unit:** is used to enter data and information into a computer. The devices like keyboard, mouse and scanner are commonly used input devices.
 - A keyboard is used to enter alphanumeric characters and symbols.
 - The mouse is used to pick or select a command from the monitor screen.
 - A scanner is used to scan an image or read a barcode and so on.
- b) Arithmetic and Logic Unit (ALU):** ALU is a digital circuit that perform arithmetic (Add, Sub, Multiplication, Division) and logical (AND, OR, NOT) operations. It helps in fast computation of scientific calculations on floating-point number.
- c) Control unit (CU):** CU is the circuitry that controls the flow of information through the processor and coordinates the activities of the other units within the processor.

Functions of Control unit

- Accessing data & instructions from memory unit
- Interpreting instructions, Controlling I/O Units

d) **Memory Unit (MU):** is the unit where all the input data and results are stored either temporarily or permanently. The CPU memory is also called as memory register. The memory of a computer has two types:

a. **Main Memory / Primary Memory units**

- i. Random Access Memory (RAM)
- ii. Read Only Memory (ROM)

b. **Secondary Memory / Auxiliary Memory**

e) **Output Unit:** It is used to display or print results from a computer. Monitor, printer and plotters are commonly used output devices.

f) **Bus:** A bus is a collection of wires that carries data/Instructions. It connects physical components such as cables, printed circuits, CPU, Memory, Peripherals etc., for sharing of Information and communication with one another. The purpose of buses is to reduce the number of "pathways" needed for communication between the components, by carrying out all communications over a single data channel.

Types of Buses:

1. **System Buses:** The system buses are used to transfer the data and instructions between Main memory (Random Access Memory) and CPU. These are classified into following three types.

Data Bus	Address Bus	Control Bus
It is used to transfer the data between Processor, Memory and I/O devices	It is used to transfer the addresses of data and Instructions stored in memory.	It is used to transfer the control signals between CPU, Memory and I/O devices.
Bidirectional in nature	Unidirectional in nature	Unidirectional or Bidirectional in nature

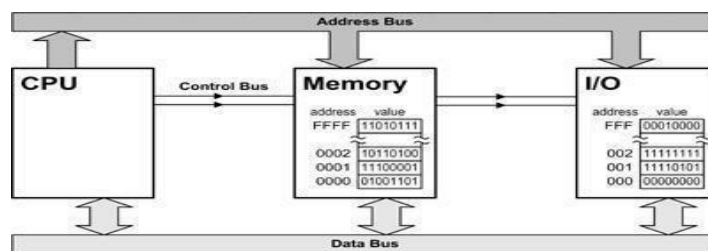


Fig 2: Types of Buses

2. **I/O Buses:** The buses which are used to connect all I/O devices with CPU and Memory are called I/O buses. These are classified into following three types.

PCI Bus	ISA Bus	USB Bus
PCI stands for <i>Peripheral</i>	ISA stands for <i>Industry</i>	USB stands for <i>Universal</i>

<i>Component Interconnect</i>	<i>Standard Architecture</i>	<i>Serial Bus</i>
The motherboard will be having 3 or 4 PCI connectors, so that we can insert various chips.	This is simple and slowest bus used in IBM PCs	It helps to connect various I/O devices like keyboard, mouse, pen drives, printer, etc.
Fastest and presently more powerful bus	Oldest, simplest and slowest Bus	Newest and widely used bus

Main Board or Mother Board: Mother Board is a set of Integrated Chips (ICs) which are designed to work together. It controls the flow of data/instructions within our computer. It is the main board on which other hardware components are connected to enable the computer system to work as an integrated unit. It consists of sockets, slots, power connectors and bus.

Chip sets: Chip set is the set of integrated chips that are designed to work together. These set of chips controls the flow of information on computer. The chips may be controllers for memory, cache, hard drive, key board and peripherals.

Operating System and its types: An Operating System (OS) is system software that **controls** and **supervises** the **hardware components** of a computer system and it provides the services to computer users. Also called as **Resource Manager** that manages the resources such as CPU, Memory, I/O devices, Job/Task/Process etc., a computer cannot run without it. The major functions of OS includes:

CPU Management,

Memory Management,

File Management,

Device Management,

Process/Task/Job Management and Security Management.

The primary goal of an OS is to make the computer system *convenient and efficient to use*. An OS ensures that the system resources (such as CPU, memory, I/O devices, etc) are utilized efficiently. For example, there may be many programs residing in the main memory. Therefore, the system needs to determine which programs are active and which need to wait for some I/O operation.

Some of the examples of Operating Systems:

Windows –XP, Unix, Windows 7, Windows 8, Macintosh OS, Fedora, and Android, etc.

Types of Operating Systems: The operating systems are classified into 7 types based on their capability and usage.

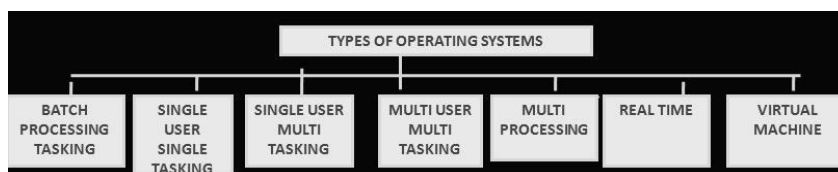
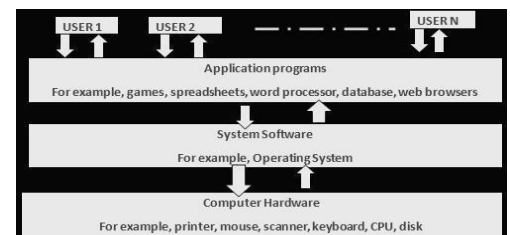


Fig 3: Types of OS

1. **Batch Processing Tasking OS:** The data is collected into a group called batch and provides only one batch (one after another) of jobs as input to the computer system at a time. The jobs in a batch are processed on first come first serve basis. In this type, the process takes place at specified time intervals i.e. weekly or monthly without user interaction. E.g. Punch cards were using to store the data in batch processing and in payroll preparation in a business batch processing was helpful.

2. **Single user and single tasking OS:** The OS that allows only one program to execute at a time is called single user single tasking operating system. Using this operating system user can do only one task at a time. **E.g.** DOS (Disk Operating System).
3. **Single user and multi-tasking OS:** The OS that allows a single user to perform more than one task at a time is called single user multi-tasking operating system. While working with the Ms-Word user can perform other work like print a document, listen music. **E.g.** Windows-XP, Windows Vista, Windows – 7, etc.
4. **Multi user and multitasking OS:** The O.S. that allows two or more users to use a main computer system to do more than one task is called multiuser and multitasking operating system. **E.g.** UNIX is a multiuser and multitasking operating system.
5. **Multiprocessing OS:** The OS that allows multiple programs to be executed by multiple CPUs (Processors) is called multiprocessing operating system. Super and main frame computers have more than one CPU and multiprocessing operating system.
6. **Real Time Operating System (RTOS):** The OS that is used for real time applications and to carry out certain calculations within the specified time constraint. This OS is used in applications such as mobile phones, supporting systems in hospitals, nuclear power plants, oil refining, chemical processing, environmental applications and air -traffic control systems, disaster management etc.,
7. **Virtual machine OS:** Allows several users of a computer system to operate as if each has the only terminal attached to the computer.

Random Access Memory (RAM): RAM is basically main memory of the computer. RAM is a semiconductor memory made up of small memory chips that form a memory module. These modules are installed in the RAM slots on the motherboard of computer. Every time you open a program, it gets loaded from the hard drive into the RAM. This is because reading data from the RAM is much faster than reading data from the hard drive.

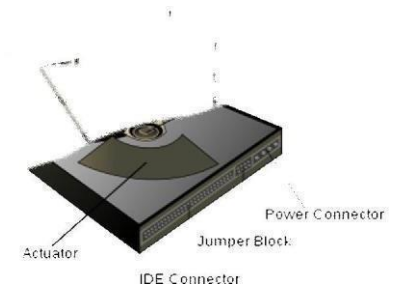


Synchronous Dynamic Random Access Memory (SDRAM): It is an improvement to standard DRAM because it retrieves data alternately between two sets of memory. This eliminates the delay caused when one bank of memory addresses is shut down while another is prepared for reading. It is called "Synchronous" DRAM because the memory is synchronized with the clock speed that the computer's CPU bus speed is optimized for. The faster the bus speed, the faster the SDRAM can be. SDRAM speed is measured in Megahertz.

FLASH memory: Flash memory is a type of Electrically Erasable Programmable Read-Only Memory (EEPROM). The name comes from how the memory is designed -- a section of memory cells can be erased in a single action or in a "flash." Flash memory cards used for digital cameras, cellular phones, networking hardware, and PC cards.

Hard disks: Hard disk is prime unit of storage of the computer. Huge amount of data can be stored and accessed in few milliseconds. The hard disk consists of more number of disks arranged in the cylindrical order, one above another on a spindle.

The read/write heads are attached to single access mechanism so that they cannot move independently. All read/write heads are moved together to position that heads on the required track. The hard disks available today ranges from 200 GB to 2TB and so on. The present day hard disk ranges from 3600 rpm to more than 10000 rpm and so on.



Advantages: High storage capacity, high data accessing rate and permanent storage medium.

Disadvantages: It is not portable.

Optical media: An optical storage media is kind of storage, which is coated with thin metal on which bits are stored. The data can be stored in to optical storage media or read form the optical storage media.

The devices which perform read or write operation on optical storage media are called optical storage media. The laser technology is used to read the data or write the data on optical storage devices.

Examples: CD-ROM, DVD etc.

Compact Disc Read-Only-Memory (CD-ROM): It is a type of optical disc that uses laser technology to read and write data on the disc. The information stored on CDRom becomes permanent and cannot be altered. This means that the stored information can only be read for processing.

A CD-ROM uses the round shaped optical disk to store data, applications, games and audio files. It can store up to 700 MB of data. It has become integral part of every organization due to its features like reliability, reasonable, storage capacity and easy to use of carry.



CD-Drive will be with motor to rotate the disks to perform read and write operations. A CD-drive will consists of the components like Disc drive, disk drive motor, laser pick up assembly tracking drive and tracking motor and so on.

Compact Disk Recordable (CD-R): The CD-R allows you to create your own CD. CD-R drives have the ability to create CDs but they can write data on the disk only once. CD-R technology also called as Write Once-Read much (WORM) technology. Laser technology is used to write the data on the compact disk. CD-R drives come in IDE, SCSI and USB models.

Compact Disc Rewritable (CD-RW): CD-RW is an erasable optical disk which is used to write data multiple times on a disk, CD-RW disks are good for data backup, data archiving or data distribution on CDs. The disk normally holds 700MB of data. Technology to write data multiple times on a CD was known as the Phase change Dual (PD) technology. The reflective properties of a CD-RW are different than regular CD-ROM disks.

Disk or Digital Versatile Disc (DVD-ROM): A DVD is a small optical disk having high density medium and capable of storing a full-length movie on a single disk. The high density is achieved by using both sides of the disk, special data-compression technology, and extremely small tracks to store the data.

Advantages: Storage capacity is more compared to CDs.

Flash Drives (Pen drives): USB flash drives are removable, rewritable, and physically much smaller drives weighing even less than 30 g. A flash drive consists of a small printed circuit board carrying the circuit elements and a USB connector, insulated electrically and protected inside a plastic, metal, or rubberized case which can be carried in a pocket or on a key chain.

Advantages

- Data stored on flash drives is impervious to scratches and dust
- Mechanically very robust
- Easily portable
- Have higher data capacity than any other removable media. Compared hard drives, flash drives use little power
- Flash drives are small and light-weight devices
- Flash drives can be used without installing device drivers.



Disadvantages

- Can sustain only a limited number of write and erase cycles before the drive fails. Most flash drives do not have a write-protect mechanism
- Flash drives are very small devices that can easily be misplaced, left behind, or otherwise lost. The cost per unit of storage in a flash drive is higher than that of hard disks

Keyboard: A keyboard is the primary input device used in all computers. Keyboard has a group of switches resembling the keys on an ordinary typewriter machine. Normally keyboard has around 101 keys. The keyboard includes key that allows us to type letters, numbers and various special symbols such as *, /, [, % etc.

Mouse: The mouse is the key input device to be used in a Graphical User Interface (GUI). The users can use mouse to handle the cursor pointer easily on the screen to perform various functions like opening a program or file.

With mouse, the users no longer need to memorize commands, which was earlier a necessity when working with text-based command line environment such as MS-DOS.

Advantages:

Easy to use; Cheap; Can be used to quickly place the cursor anywhere on the screen Helps to quickly and easily draw figures
Point and click capabilities makes it unnecessary to remember certain commands

Disadvantages:

Needs extra desk space to be placed and moved easily
The ball in the mechanical mouse needs to be cleaned very often for smooth movements

Printers: The printer is an output device, which is used to get hard copy of the text displayed on the screen. The printer is an external optional device that is connected to the computer system using cables. The printer driver software is required to make the printer working. The performance of a printer is measured in terms of Dots Per Inch (DPI) and Pages Per Minute (PPM) produced by the printer.

Plotters: A plotter is similar to printer that produces hard-copy output with high-quality color graphics. Plotters are generally more expensive than printers, ranging from about \$1000 to \$75000.

Problem Solving Techniques:

The process of working through details of a problem to reach a solution. There are three approaches to problem solving:

1. **Algorithm**
2. **Flowchart**
3. **Pseudo Code**

Algorithm: The algorithm is a *step-by-step procedure* to be followed in solving a problem. It provides a scheme to solve a particular problem in *finite number of unambiguous steps*. It helps in implementing the solution of a problem using any of the *programming languages*.

In order to qualify as an algorithm, a sequence of instructions must possess the following characteristics:

Definiteness: Instructions must be *precise and unambiguous* i.e. each and every instruction should be clear and should have only one meaning.

Finiteness: *Not* even a single instruction must be *repeated infinitely*. i.e., each instruction should be performed in finite time.

Termination: After the algorithm gets executed, the user should get the desired *result*

Key features of an algorithm:

Any algorithm has a finite number of steps and some steps may involve decision making, repetition. Broadly speaking, an algorithm exhibits three key features that can be given as:

Sequence: Sequence means that each step of the algorithm is executed in the specified order.

Decision: Decision statements are used when the outcome of the process depends on some condition.

Repetition: Repetition which involves executing one or more steps for a number of times can be implemented using constructs like the while, do-while and for loops. These loops executed one or more steps until some condition is true.

Example: To compute the Area of Rectangle

ALGM: AREA_of_RECTANGLE [This algorithm takes length and breadth, the sides of the rectangle as input and computes the area of rectangle using the formula *area=length * breadth*. Finally it prints the area of rectangle]

STEPS:

Step 1: Start

Step 2: [Input the sides of Rectangle]

Read **length, breadth**

Step 3:[Compute the area of rectangle]

Area **length*breadth**



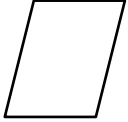
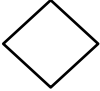
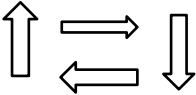



Step 4:[Display the Area]

Print **Area**

Step 5: [Finished]

Stop

Flowcharts: A flowchart is a graphical or symbolic representation of an algorithm. They are basically used to design and develop complex programs to help the users to visualize the logic of the program so that they can gain a better understanding of the program and find flaws, bottlenecks, and other less obvious features within it. Basically, a flowchart depicts the “*flow*” of a program. The following table shows the symbols used in flowchart along with its descriptions.

Symbol	Name	Description
	oval	Represents the terminal point
	Rectangle	Represents the process steps defined in algorithm
	Parallelogram	Indicate the reading Operation used for input/output or data or information from/to any device
	Diamond	Indicates the decisions (questions) and consequently branch points or the paths to be followed based on the result of the question
	Arrows	Shows the flowchart direction and connects the various flow chart symbols.
	Small circle	Shows the continuation from one point in the process flow to another.
	Process	Subroutine function
	Hexagon	Represents Looping structures

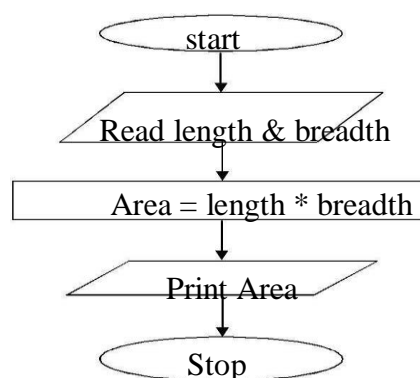
Advantages of Flowcharts:

- A flowchart is a diagrammatic representation that illustrates the sequence of steps that must be performed to solve a problem. They are usually drawn in the early stages of formulating computer solutions to facilitate communication between programmers and business people.
- Flowcharts help programmers to understand the logic of complicated and lengthy problems. They help to analyse the problem in a more effective manner
- Flowchart can be used to debug programs that have error(s).

Limitations of using Flowcharts:

Drawing flowcharts is a laborious and a time consuming activity. Flowchart of a complex program becomes, complex and clumsy. At times, a little bit of alteration in the solution may require complete redrawing of the flowchart. Essentials of what is done may get lost in the technical details of how it is done. There are no well-defined standards that limits the details that must be incorporated in a flowchart

E.g.: To compute the Area of Rectangle



ANSI STANDARD FLOWCHART SYMBOLS

Production Activity Symbols	Activity or Operation	Decision	Flow of Control	Report
Documentation Symbols	Start/End	Annotation		
Storage Activity Symbols	Storage	Delay		
Transportation Activity Symbols	Transmission	Physical Movement		
Inspection Activity Symbols	Inspector			

Pseudo code: It is a form of structured English that describes algorithms. It facilitates the designers to focus on the logic of the algorithm without getting bogged down by the details of language syntax.

Pseudocode is a compact and informal high-level description of an algorithm that uses the structural conventions of a programming language. It is meant for human reading rather than machine reading, so it omits the details that are not essential for humans. Such details include keywords, variable declarations, system-specific code and subroutines. There are no standards defined for writing a *pseudocode* because it is not an executable program. Flowcharts can be considered as a graphical alternative to *pseudocode*, but are more spacious on paper.

E.g.: To compute the area of Rectangle

Begin

Input length, breadth

Area=length*breadth

Print Area

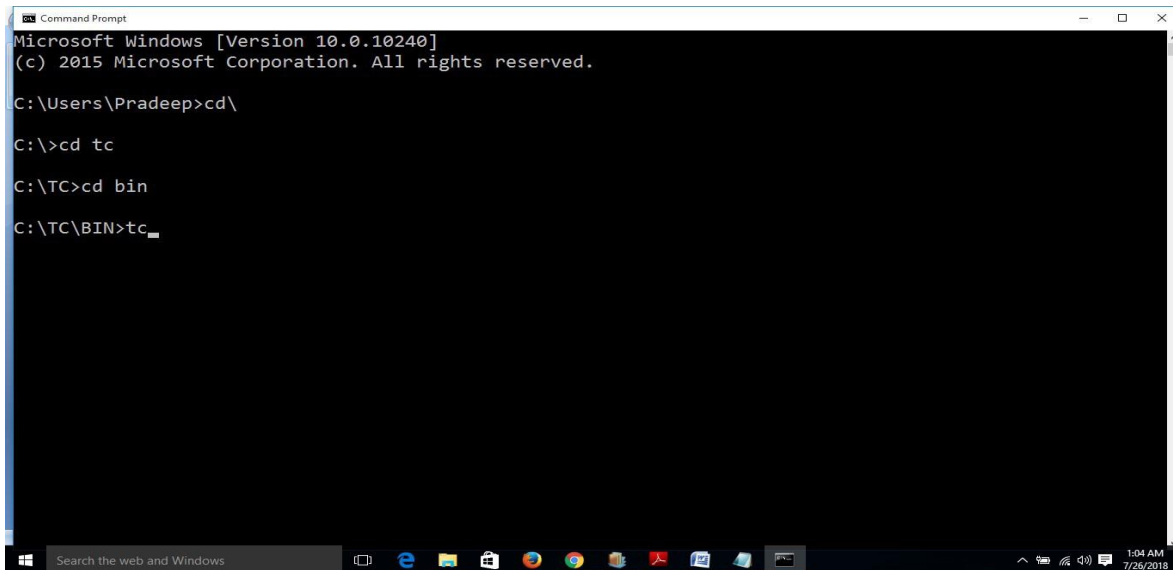
End

Laboratory 1

Familiarization with computer hardware and programming environment, concept of naming the program files, storing, compilation, execution and debugging. Taking any simple C- code.

Working with TurboC

Step 1: Locate the TC.exe file and open it. You will find it at location C:\TC\BIN\.



```
Microsoft Windows [Version 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

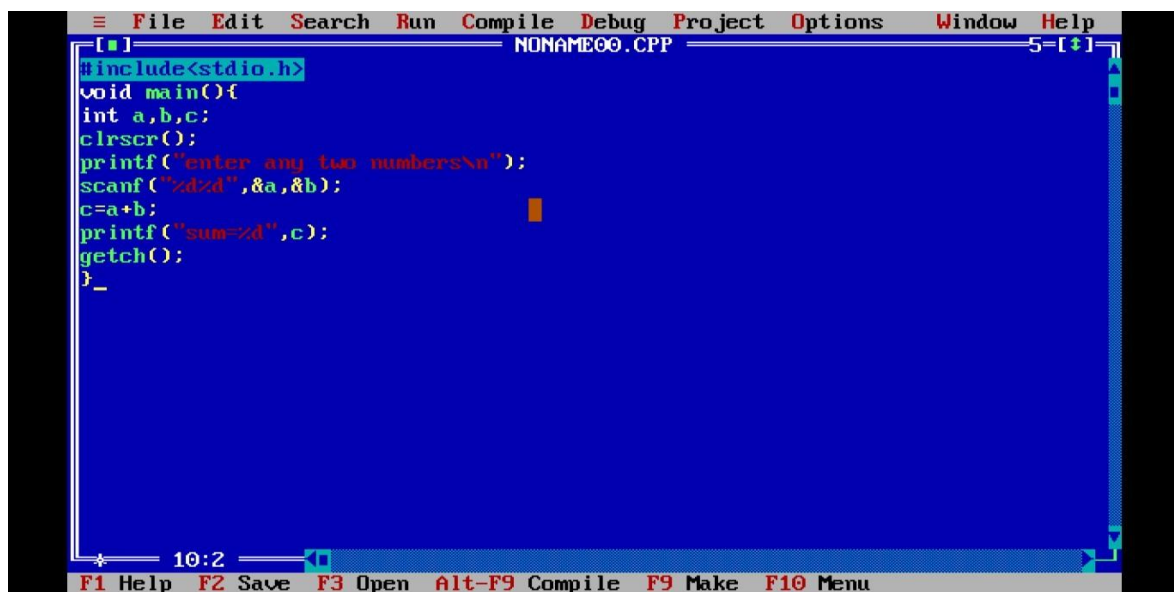
C:\Users\Pradeep>cd \

C:\>cd tc

C:\TC>cd bin

C:\TC\BIN>tc_
```

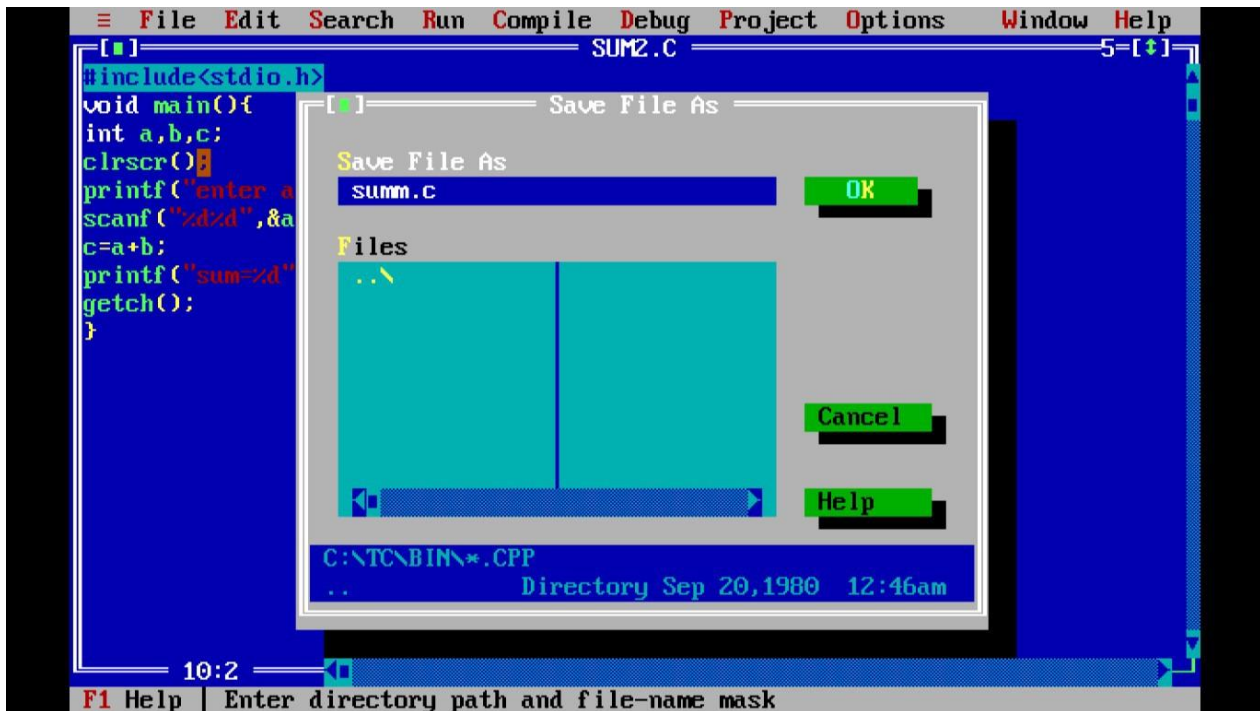
Step 2: File > New (as shown in the below picture) and then write your C program



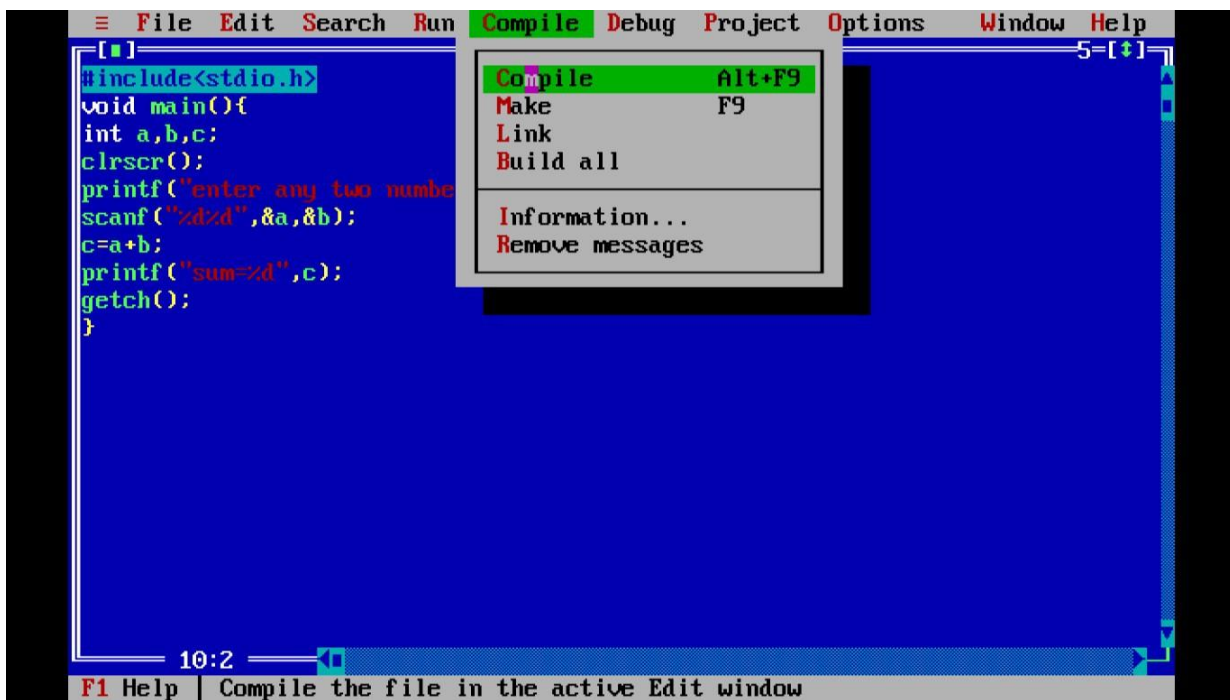
```
File Edit Search Run Compile Debug Project Options Window Help
NONAME00.CPP 5-[↑]
#include<stdio.h>
void main()
int a,b,c;
clrscr();
printf("enter any two numbers\n");
scanf("%d%d",&a,&b);
c=a+b;
printf("sum=%d",c);
getch();
}_

10:2
F1 Help F2 Save F3 Open Alt-F9 Compile F9 Make F10 Menu
```

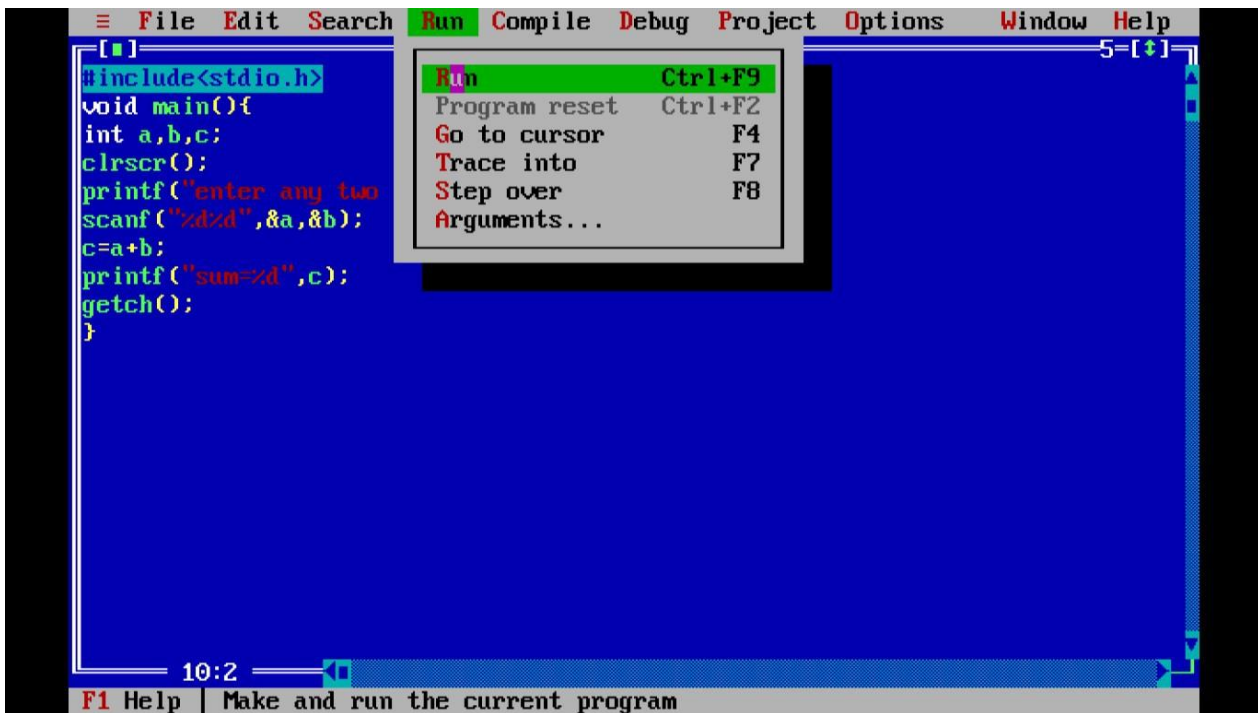
Step 3: Save the program using F2 (OR file > Save), remember the extension should be “.c”. In the below screenshot I have given the name as summ.c.



Step 4: Compile the program using Alt + F9 OR Compile > Compile (as shown in the below screenshot).



Step 5: Press Ctrl + F9 to Run (or select Run > Run in menu bar) the C program.



The screenshot shows a development environment with a menu bar (File, Edit, Search, Run, Compile, Debug, Project, Options, Window, Help) and a code editor. The code in the editor is:

```
#include<stdio.h>
void main(){
int a,b,c;
clrscr();
printf("enter any two numbers\n");
scanf("%d%d",&a,&b);
c=a+b;
printf("sum=%d",c);
getch();
}
```

The 'Run' menu is open, showing options: Run (Ctrl+F9), Program reset (Ctrl+F2), Go to cursor (F4), Trace into (F7), Step over (F8), and Arguments... The status bar at the bottom indicates '10:2' and 'F1 Help | Make and run the current program'.

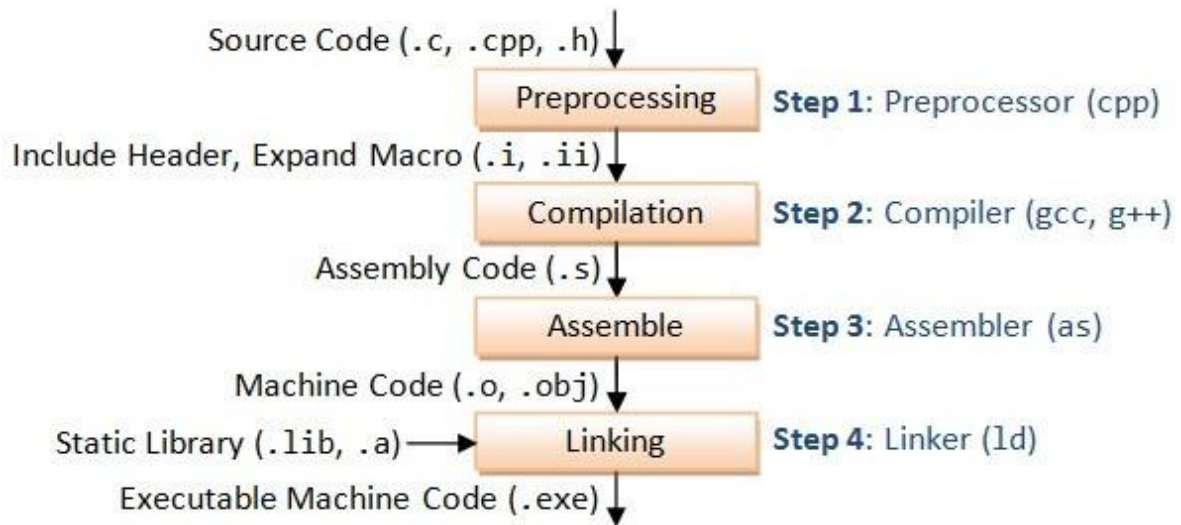
Step 6: Alt+F5 to view the output of the program at the output screen.

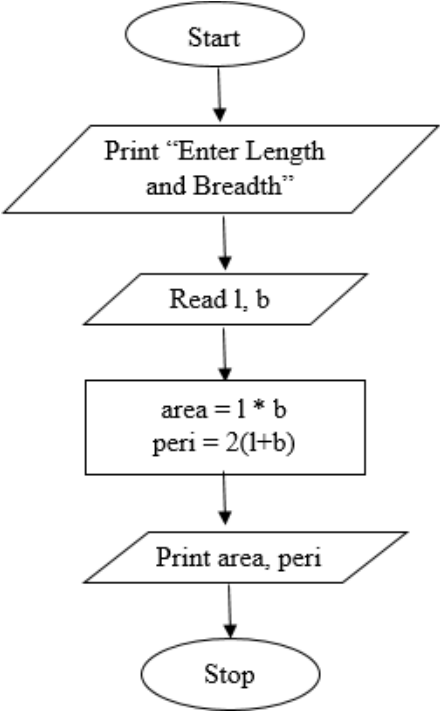


The screenshot shows the output of the program on a black background:

```
enter any two numbers
23
40
sum=63_
```

GCC Compilation Process



Algorithm	Flowchart
<p>Step 1. [Initialize] Start</p> <p>Step 2. [Input the length and breadth] Read l, b</p> <p>Step 3. [Calculate area] $area = l * b$</p> <p>Step 4. [Calculate perimeter] $peri = 2(l+b)$</p> <p>Step 5. [Display area and perimeter] Print area, peri</p> <p>Step 6. [Finished] Stop.</p>	 <pre> graph TD Start([Start]) --> Print1[/Print "Enter Length and Breadth"/] Print1 --> Read[/Read l, b/] Read --> Process[area = l * b peri = 2(l+b)] Process --> Print2[/Print area, peri/] Print2 --> Stop([Stop]) </pre>
Program	Output
<pre> #include<stdio.h> { int l, b, area, peri; printf("Enter length and breadth\n"); scanf("%d%d", &l, &b); area = l * b; peri = 2 * (l + b); printf("The area is %d\n", area); printf("The perimeter is %d\n", peri); } </pre>	<pre> Enter length and breadth 2 4 The area is 8 The perimeter is 12 </pre>

Laboratory Program 1

SIMULATION OF A SIMPLE CALCULATOR

AIM: To write a C Program To Simulate A Simple Calculator

ALGORITHM

Step 1: Start

Step 2: Read the value of operator op.

Step 3: Read the values of num1 and num2.

Step 4: Verify op value by using switch statement

if op='+' then goto step 5 (or)

if op='-' then goto step 6 (or)

if op='*' then goto step 7 (or)

if op='/' then goto step 8 (or)

otherwise goto step 9

Step 5: if op='+' then compute result=num1+num2

print result and goto step 10

Step 6: if op='-' then compute result=num1-num2

print result and goto step 10

Step 7: if op='*' then compute result=num1*num2

print result and goto step 10

Step 8: if op='/' then compute divresult=num1/num2

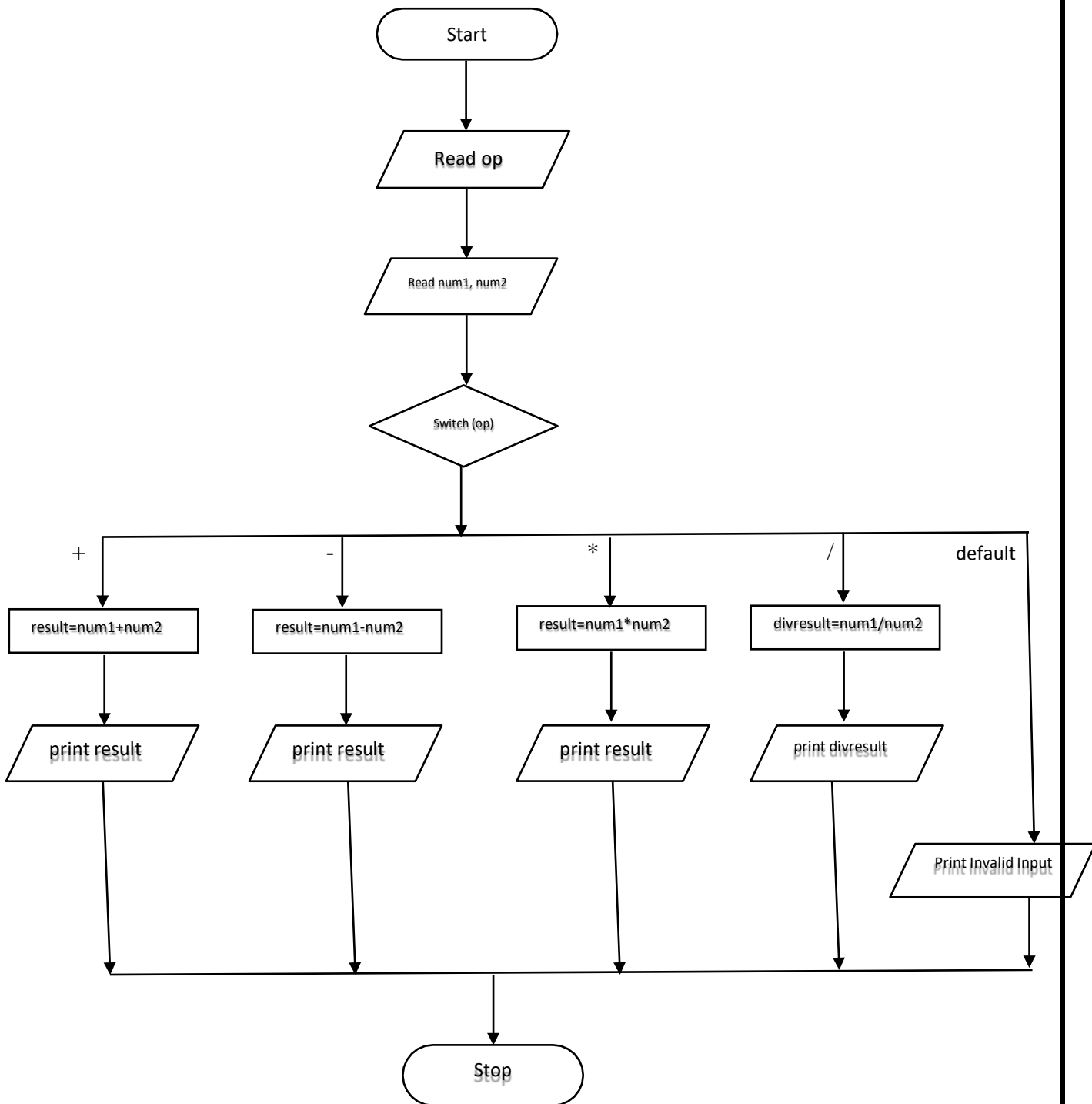
print divresult and goto step 10

Step 9: if op value is other than above options then

print "Invalid input, Try again" goto step 10

Step 10: Stop

FLOWCHART



PROGRAM

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

void main()
{
int num1,num2,result;
float divresult;
char op;
clrscr();
printf("Enter the operation to be done \n");
scanf("%c",&op);
printf("Enter the value of num1 and num2 \n");
scanf("%d%d",&num1,&num2);
switch(op)
{
case '+': result = num1 + num2;
printf("Addition=%d\n",result);
break;
case '-': result = num1 - num2;
printf("Subtraction=%d\n",result);
break;
case '*': result = num1 * num2;
printf("Multiplication=%d\n",result);
break;
case '/': divresult = num1 / (float)num2;
printf("Division=%.2f\n",divresult);
break;
default: printf("Invalid Input ,Try Again\n");
}
getch();
}
```

OUTPUT

First Run:

Enter the operation to be done

+
Enter the value of num1 and num2
10
20
Addition=30

Second Run:
Enter the operation to be done
-
Enter the value of num1 and num2
10
20
Subtraction=-10

Third Run:
Enter the operation to be done
*
Enter the value of num1 and num2
20
30
Multiplication=600

Fourth Run:
Enter the operation to be done
/
Enter the value of num1 and num2
40
50
Division=0.80

Fifth Run:
Enter the operation to be done
@
Enter the value of num1 and num2
40
50
Invalid Input ,Try Again

RESULT: Thus, the program to implement the simple calculator has been executed successfully and the output was verified.

VIVA QUESTIONS:-

- a. What is switch statement?
- b. What is a case in a switch statement?
- c. Is Default necessary in switch case?
- d. How many cases can you have in a switch statement?

Laboratory Program 2

COMPUTE THE ROOTS OF A QUADRATIC EQUATION BY ACCEPTING THE COEFFICIENTS. PRINT APPROPRIATE MESSAGES.

AIM: To write a C Program to Compute the roots of a quadratic equation by accepting the coefficients

ALGORITHM

Step 1: Start

Step 2: Read the value of non-zero coefficient a.

Step 3: If a is zero goto Step 10. Otherwise read the values of coefficients b and c.

Step 4: Compute the value of discriminant (disc) using the formula $(b*b) - (4*a*c)$.

Step 5: Check if disc is equal to 0. If true, then go to Step 6. Otherwise, goto Step 7

Step 6: Compute the equal roots.

$$\text{root1} = \text{root2} = (-b)/(2*a)$$

Print the values of roots, root1 and root2. Go to Step 10.

Step 7: Check if disc is greater than zero or not. If true, then go to Step 8. Otherwise, goto Step 9.

Step 8: Compute the real and distinct roots

$$\text{root1} = (-b + \sqrt{\text{disc}})/(2*a)$$

$$\text{root2} = (-b - \sqrt{\text{disc}})/(2*a)$$

Print the values of roots, root1 and root2. Go to Step 10.

Step 9: Compute the complex and imaginary roots.

$$\text{Compute the real part, } \text{realp} = (-b)/(2*a)$$

$$\text{Compute the imaginary part, } \text{imagp} = \sqrt{-\text{disc}}/(2*a)$$

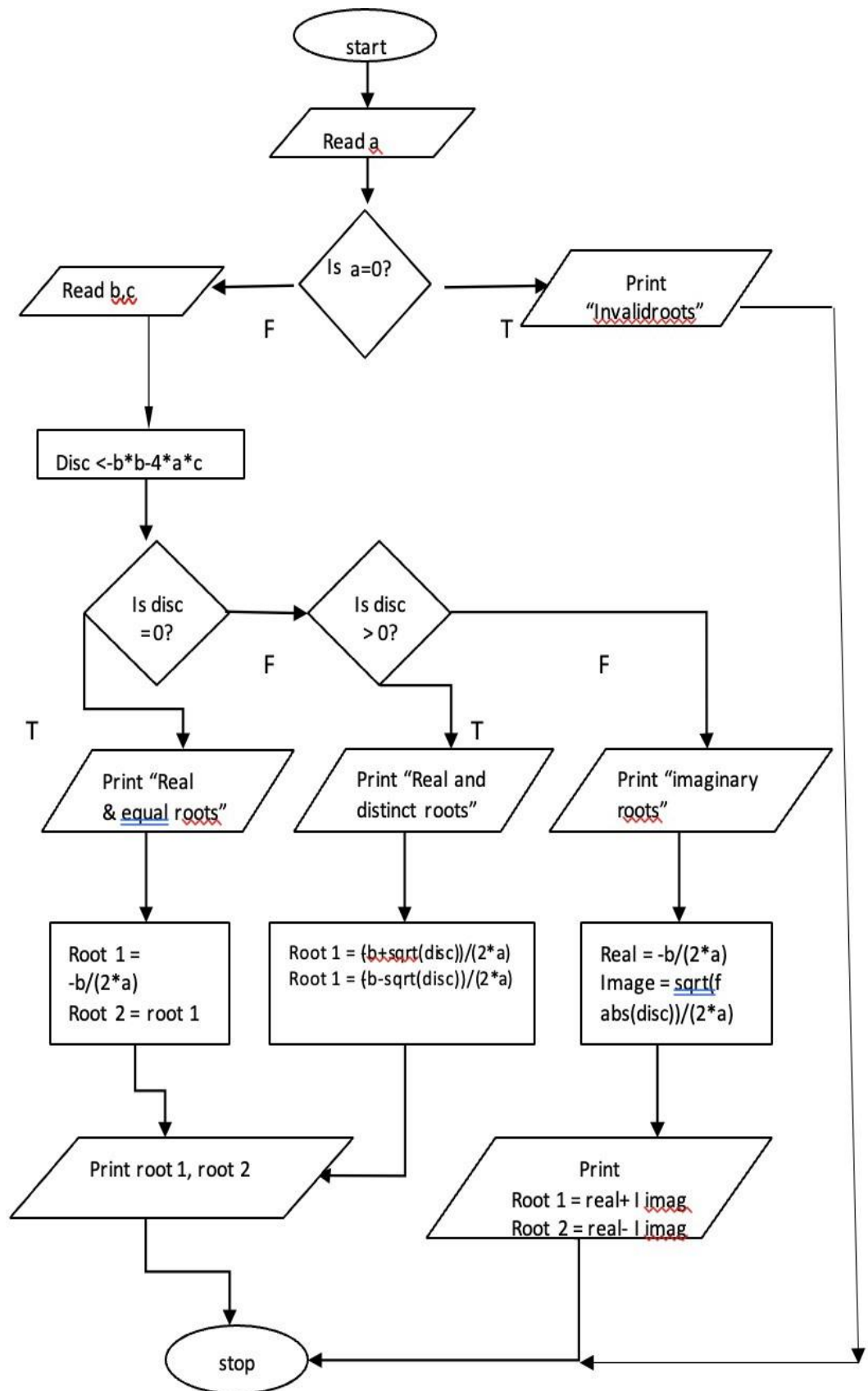
Print roots as

$$\text{root1} = \text{realp} + \text{imagp}$$

$$\text{root2} = \text{realp} - \text{imagp}$$

Step 10: Stop.

FLOWCHART



PROGRAM

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
#include <stdlib.h>

void main()
{
    float a,b,c,root1,root2,realp,imagp,disc;
    clrscr();

    printf("\n Enter the value of coefficient a: ");
    scanf("%f",&a);

    if(a == 0)
    {
        printf("\n Invalid input...Retry again");
        exit(0);
    }
    printf(" Enter the value of coefficients b and c:\n ");
    scanf("%f%f",&b,&c);
    disc = b*b-4*a*c; // compute discriminant

    if(disc == 0)
    {
        printf("The roots are real and equal\n");
        root1 = root2 = -b / (2.0*a);
        printf(" Root1 = Root2 = %.2f\n", root1);
    }
    else
    {
        if(disc > 0)
        {
            printf("The roots are real and distinct\n");
            root1 = (-b + sqrt(disc))/(2.0*a);
            root2 = (-b - sqrt(disc))/(2.0*a);
            printf("Root1 = %.2f\n", root1);
            printf("Root2 = %.2f\n", root2);
        }
        else
        {
            printf("The roots are complex\n");
            realp = -b/(2.0*a);
            disc=-disc;
            imagp = sqrt(disc)/(2.0*a);
```

```
        printf("Root1 = %.2f + i%.2f\n",realp,imagp);
        printf("Root2 = %.2f - i %.2f\n",realp, imagp);
    }
}
getch();
}
```

OUTPUT

First Run:

Enter the value of coefficient a: 1

Enter the value of coefficients b and c:

4 4

The roots are real and equal

Root1 = Root2 = -2.00

Second Run:

Enter the value of coefficient a: 1

Enter the value of coefficients b and c:

-7 10

The roots are real and distinct

Root1 = 5.00

Root2 = 2.00

Third Run:

Enter the value of coefficient a: 2

Enter the value of coefficients b and c:

-3 6

The roots are complex

Root1 = 0.75 + i 1.56

Root2 = 0.75 - i 1.56

Fourth Run:

Enter the value of coefficient a: 0

Invalid input...Retry again

RESULT: Thus, the program to compute the roots of quadratic equation has been executed successfully and the output was verified.

VIVA QUESTIONS:-

- What is quadratic Equation?
- What is math.h ? why it is used in C program?
- What are decision making statements in C language?
- Write the syntax of “ if ”statement?
- Difference between if and switch statements?

Laboratory Program 3

AN ELECTRICITY BOARD CHARGES THE FOLLOWING RATES FOR THE USE OF ELECTRICITY: FOR THE FIRST 200 UNITS 80 PAISE PER UNIT: FOR THE NEXT 100 UNITS 90 PAISE PER UNIT: BEYOND 300 UNITS RUPEES 1 PER UNIT. ALL USERS ARE CHARGED A MINIMUM OF RUPEES 100 AS METER CHARGE. IF THE TOTAL AMOUNT IS MORE THAN RS 400, THEN AN ADDITIONAL SURCHARGE OF 15% OF TOTAL AMOUNT IS CHARGED. WRITE A PROGRAM TO READ THE NAME OF THE USER, NUMBER OF UNITS CONSUMED AND PRINT OUT THE CHARGES.

AIM

To write a C program to read the Name of the User, Number of Units Consumed and Print out the charges.

ALGORITHM

Step 1: Start

Step 2: Read the name of customer and the unit consumed by the customer.

Step 3: Check if the unit consumed is greater than 1 and less than 200, if true goto step 4 else goto step 5.

Step 4: Compute: $\text{amt} = 100 + (0.8 * \text{units})$.

Step 5: if unit is greater than 200 and less than 300, if true goto step 6 else goto step 7

Step 6: Compute $\text{amt} = 100 + (200 * 0.8) + ((\text{units} - 200) * 0.9)$

Step 7: Compute $\text{amt} = 100 + (200 * 0.8) + (100 * 0.9) + ((\text{units} - 300) * 1)$, then goto step 8

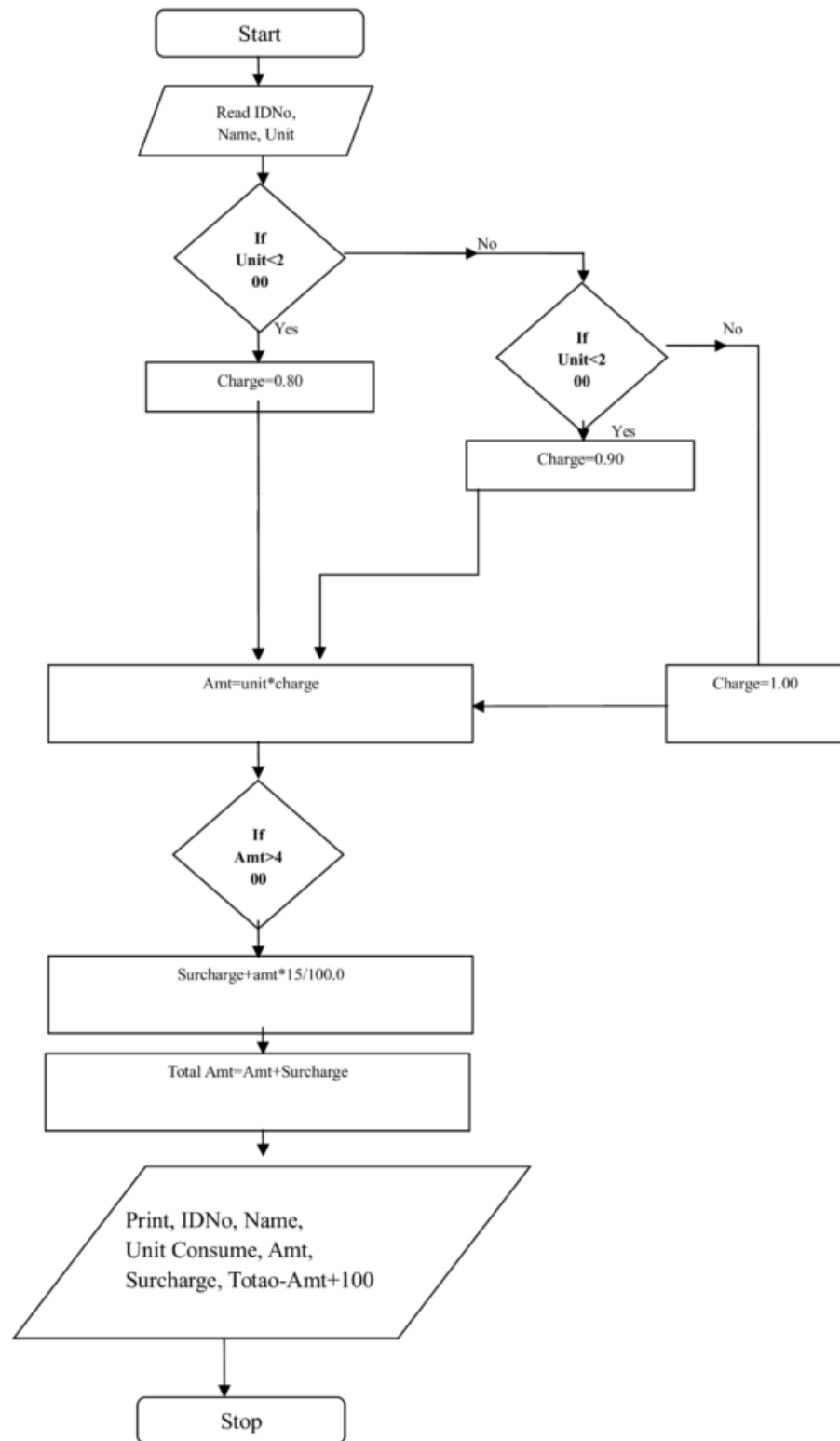
Step 8: Check if the amt is greater than or equal to 400, if true goto step 9 otherwise goto step 10.

Step 9: Compute $\text{amt} = \text{amt} * 1.15$, goto step 10

Step 10: Print the amount charged and goto step 11.

Step 11: Stop

FLOW CHART



PROGRAM

```
#include <stdio.h>
#include <conio.h>
void main()
{
char name[10];

float unit, amt;

clrscr();
printf("Enter your name and unit Consumed:");
scanf("%s %f",name,&unit);
if(unit<=200)
amt=unit*0.80+100;
else if((unit>200)&&(unit<=300))
amt=200*0.80+((unit-200)*0.90)+100;
else
amt=200*0.80+100*0.90+((unit-300)*1)+100;
if(amt>400)

amt=1.15*amt;
printf("Name: %s\n Unit=%f \n charge=%f ",name,unit,amt);
getch();
}
```

OUTPUT

First Run:

Enter your name and unit Consumed: Siri 52Name: Siri

Unit=52 charge=141.600000Second Run:

Enter your name and unit Consumed: Rajesh 460Name: Rajesh

Unit=460 charge=586.500000

RESULT: -

Thus, the program to read the Name of the User, Number of Units Consumed and Print out the charges has been executed successfully and the output was verified.

VIVA QUESTIONS: -

- a. Difference between float and double data types.
- b. Write syntax of for loop?
- c. What is the use of break statement?
- d. Difference between continue and break statement?

Laboratory Program 4

WRITE A C PROGRAM TO DISPLAY THE FOLLOWING BY READING THE NUMBER OF ROWS AS INPUT,

```
    1
   1 2 1
  1 2 3 2 1
 1 2 3 4 3 2 1
```

nth row

AIM

To write a C program to display the pattern by reading the number of rows as input.

ALGORITHM

Step 1: Start

Step 2: Read the number of rows.

Step 3: Let $i=0$, if $i \leq n$, go to step 14

Step 4: Initialize $j=1$ if $j \leq n-i$, Go to step 8

Step 5: Print blank spaces

Step 6: Increment j (ie. $j=j+1$) Go to step 5

Step 7: Initialize $j=1$ if $j \leq i$ Go to step 11

Step 8: Display number in ascending order up to middle.

Step 9: Increment j (ie. $j=j+1$) Go to step 8

Step 10: initialize $j=i-1$ if $j \geq 1$ Go to step 14

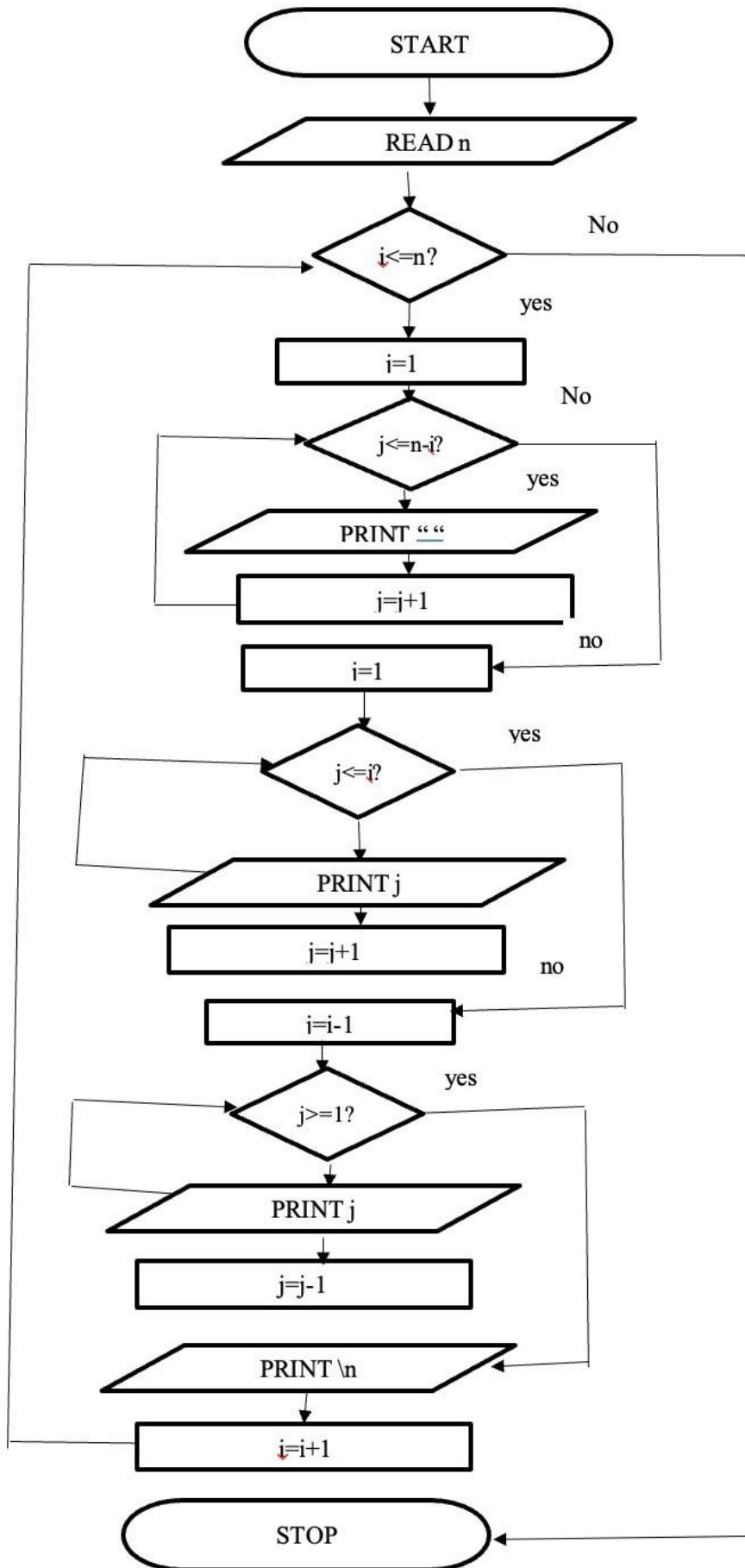
Step 11: Display number in reverse order after middle

Step 12: Increment j (ie. $j=j+1$) Go to step 13

Step 13: Increment i (ie. $i=i+1$) Go to step 3

Step 14: Stop.

FLOWCHART



PROGRAM

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int i,j,n;
    clrscr( );
    printf("Input number of rows : ");
    scanf("%d",&n);
    for(i=0; i<=n;i++)
    {
        for(j=1;j<=n-i;j++)
        {
            printf(" ");
        }
        for(j=1;j<=i;j++)
        {
            printf("%d",j);
        }
        for(j=i-1;j>=1;j--)
        {
            printf("%d",j);
        }
        printf("\n");
    }
    getch();
}
```

OUTPUT

First Run:

Input number of rows: 4

```
    1
   1 2 1
  1 2 3 2 1
 1 2 3 4 3 2 1
```

Second Run:

Input number of rows : 6

```
        1
       1 2 1
      1 2 3 2 1
     1 2 3 4 3 2 1
    1 2 3 4 5 4 3 2 1
   1 2 3 4 5 6 5 4 3 2 1
```

RESULT: -

Thus, the program to display the pattern by reading the number of rows as input has been executed successfully and the output was verified.

VIVA QUESTIONS:-

- a. Write syntax of for loop?
- b. What is the use of nested for loop?
- c. Define Pattern.

Laboratory Program 5

IMPLEMENT BINARY SEARCH ON INTEGERS.

AIM: To write a C program to Implement Binary Search on Integers.

ALGORITHM

Step 1: Start

Step 2: Read size of the array n

Step 3: Read the list of elements in sorted order a[i].

Step 4: Read the key element in key

Step 5: initialize low=0 and high= n-1

Step 6: Check if low is less than or equal to high. If condition is true, goto step 7, otherwise goto Step 11

Step 7: find the middle element.i.e. $mid=(low+high)/2$;

Step 8: if key element is equal to mid element, then initialize loc value, $loc = mid+1$; otherwise goto step 9

Step 9: Check if key element is less than mid element is true, then initialize $high=mid-1$ then goto step 6, if it is false goto step10.

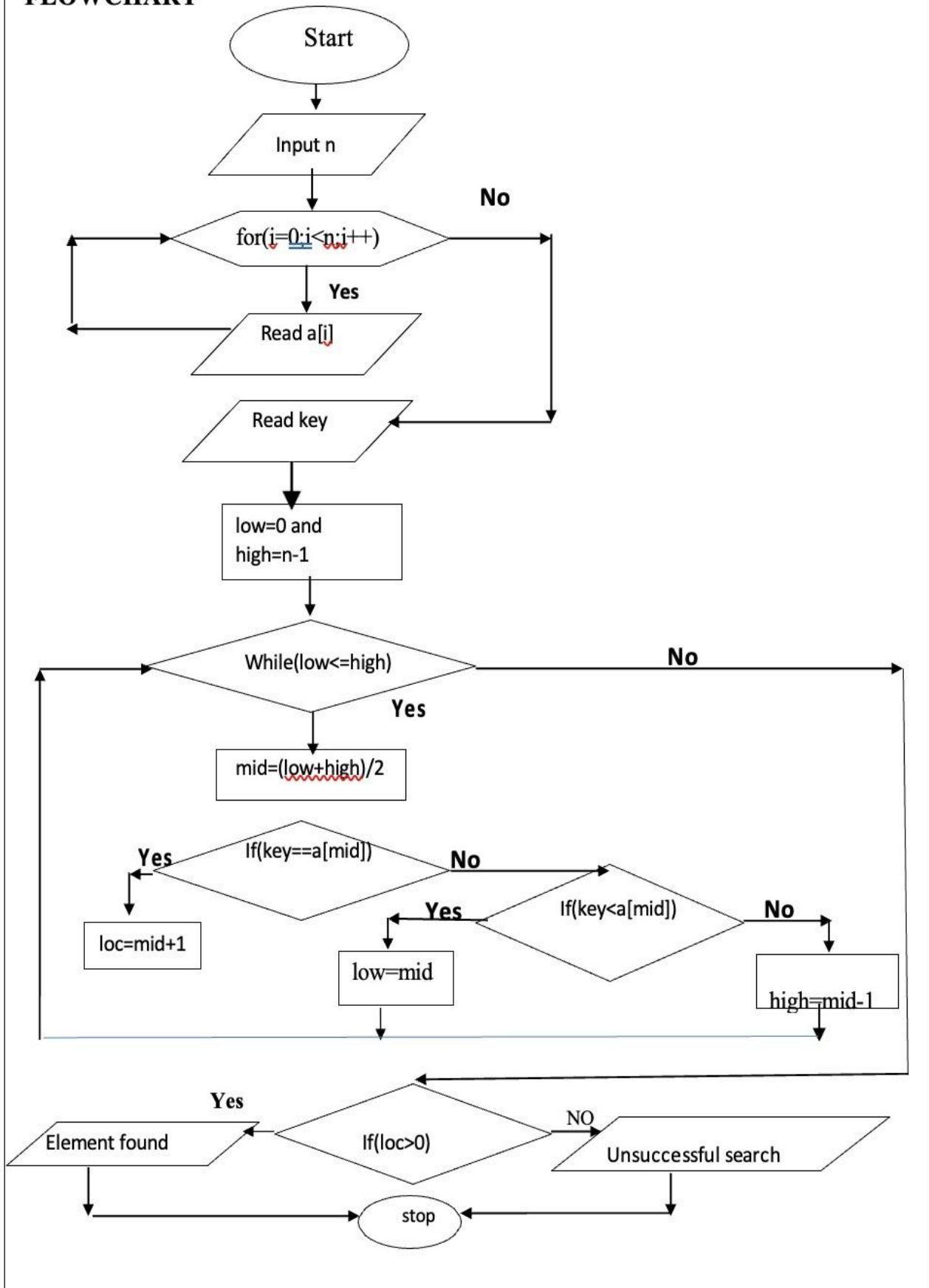
Step 10: Check if key element is greater than mid element is true, then initialize $low=mid+1$ then goto step 6.

Step 11: Check if loc value is greater than zero then print the search is successful then goto step 13, otherwise goto step 12.

Step 12: print search is unsuccessful, then goto step 13.

Step 13: Stop

FLOWCHART



PROGRAM

```
#include <stdio.h>
#include <conio.h>
void main()
{
int n, a[100], i, key, high, low, mid, loc=-1;
clrscr( );
printf("Enter the size of the array\n");
scanf("%d",&n);
printf("Enter the elements of array in sorted order\n");
for(i=0;i<n;i++)
scanf("%d",&a[i]);
printf("Enter the key element to be searched\n");
scanf("%d",&key);
low=0;
high=n-1;
while(low<=high)
{
mid=(low+high)/2;
if(key==a[mid])
{
loc = mid+1;
break;
}
else
{
if(key<a[mid])
high=mid-1;
else
low=mid+1;
}
}
if(loc>0)
printf("\n The element %d is found at %d ",key,loc);
```

```
else
printf("\nThe search is unsuccessful");
getch();
}
```

OUTPUT

First Run:

```
Enter the size of the array 5
Enter the elements of array in sorted order 10
20
30
40
50
Enter the element to be searched 40
The element 40 is found at 4
```

Second Run:

```
Enter the size of the array
4
Enter the elements of array in sorted order
4
6
8
9
Enter the key element to be searched
2
```

The search is unsuccessful

RESULT: -Thus, the program to implement the binary search on integers has been executed successfully and the output was verified.

VIVA QUESTIONS: -

- a. What is an array/definition of array.
- b. What are the types of array?
- c. What is a multidimensional array?
- d. How to declare and initialize one dimensional array?
- e. What are the advantages of an array?
- f. What is the difference between array & string?
- g. Write the syntax of declaring an array.

Laboratory Program 6

IMPLEMENT MATRIX MULTIPLICATION AND VALIDATE THE RULES OF MULTIPLICATION.

AIM: To develop a program to introduce 2D Array manipulation and implement Matrix multiplication and ensure the rules of multiplication are checked.

ALGORITHM:

Step 1: GET THE SIZE OF MATRIX a

Input the size of matrix a and read the values of m and n.

Step 2: GET THE VALUES OF MATRIX a

For i=0, i < m, i++

For j=0, j < n, j++

Read a[i][j]

End For

End For

Step 3: GET THE SIZE OF MATRIX b

Input the size of matrix b and read the values of p and q.

Step 4: GET THE VALUES OF MATRIX b

For i=0, i < p, i++

For j=0, j < q, j++

Read b[i][j]

End For

End For

STEP 5:

If(n!=p)

Print("multiplication of matrices is not possible")

Goto Step 8 otherwise goto step 6

End if

STEP 6:

For i=0, i < m, i++

For j=0, j < q, j++

$c[i][j]=0$

For $k=0, k < n, k++$

$c[i][j]=c[i][j]+a[i][k]*b[k][j]$

End for

End for

Step 7: DISPLAY RESULT

For $i=0, i < m, i++$

For $j=0, j < q, j++$

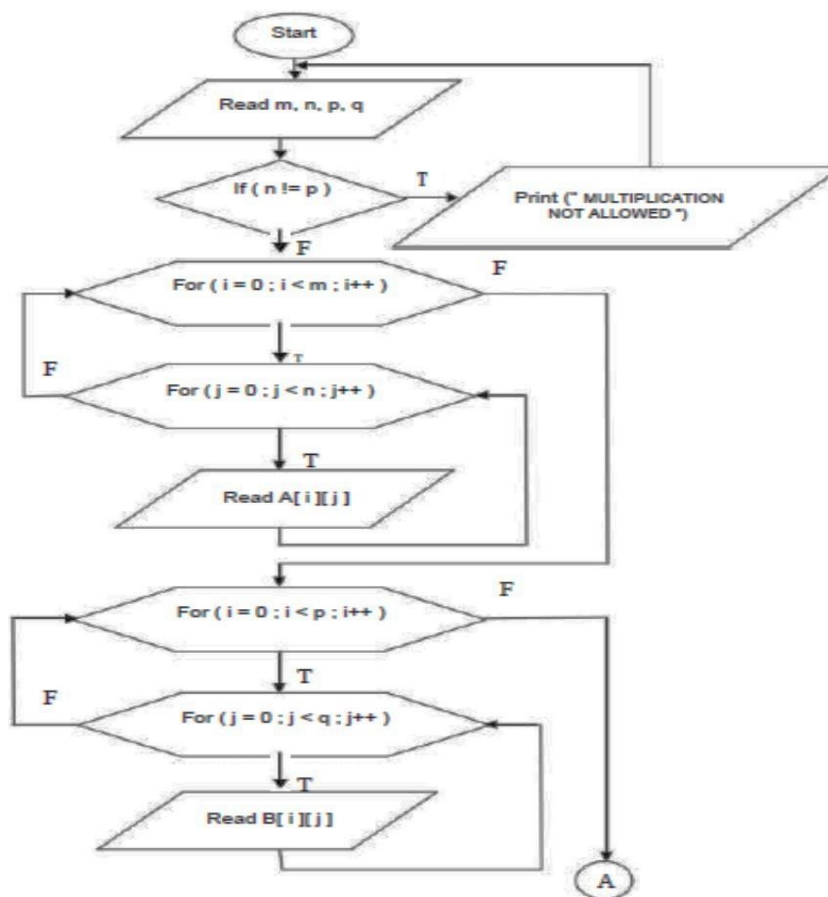
Print $c[i][j]$

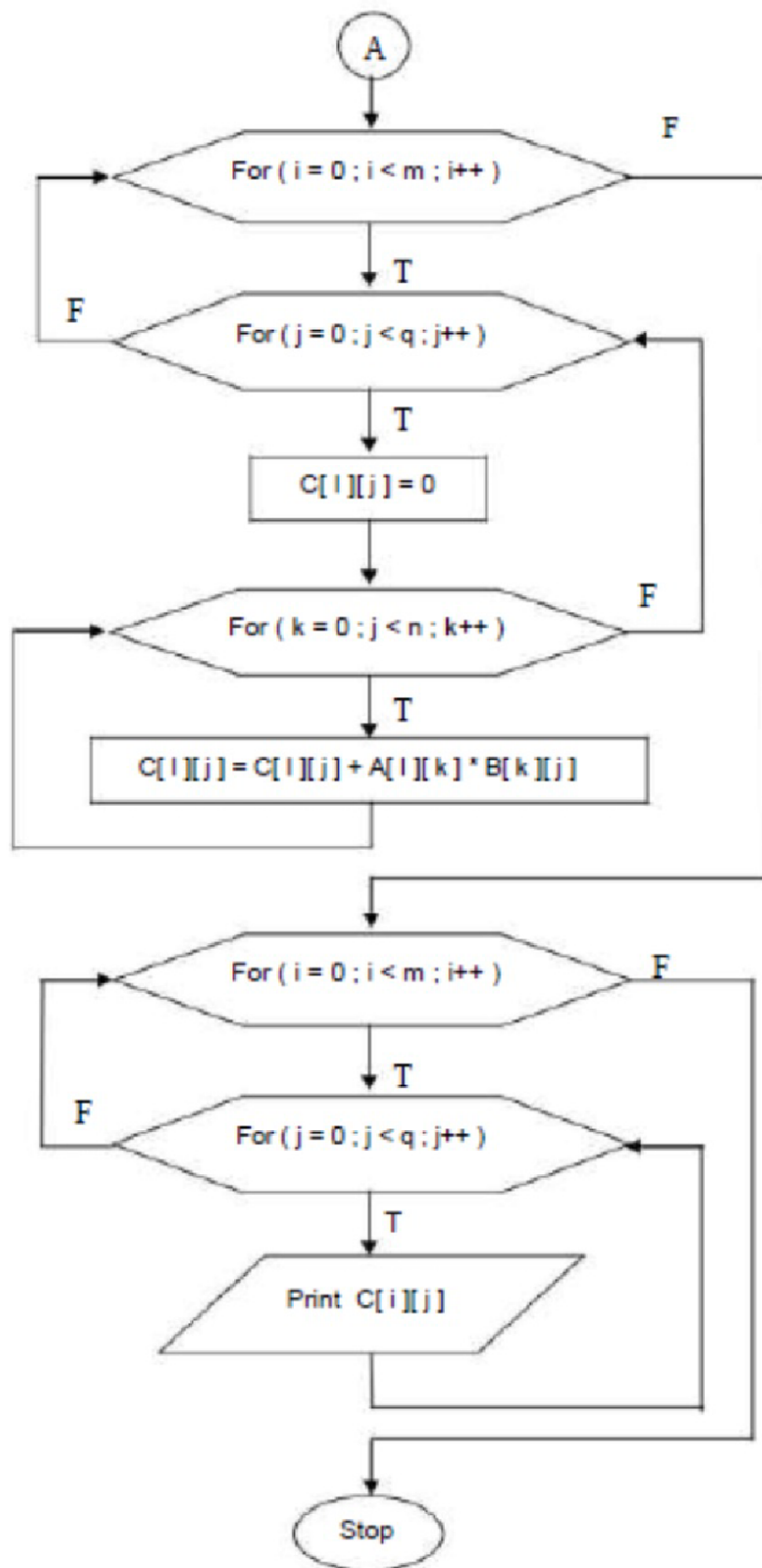
End for

End for

Step 8: STOP

FLOWCHART





PROGRAM

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int a[5][5],b[5][5],c[5][5],m,n,p,q,i,j,k;
    clrscr();
    printf("Enter the size of first matrix\n");
    scanf("%d %d",&m,&n);
    printf("Enter the size of second matrix\n");
    scanf("%d %d",&p,&q);
    if(n!=p)
        printf("Matrix multiplication is not possible");
    else
    {
        printf("Enter the elements of first matrix\n");
        for(i=0;i<m;i++)
            for(j=0;j<n;j++)
                scanf("%d",&a[i][j]);
        printf("Enter the elements of the second matrix\n");
        for(i=0;i<p;i++)
            for(j=0;j<q;j++)
                scanf("%d",&b[i][j]);
        for(i=0;i<m;i++)
            for(j=0;j<q;j++)
            {
                c[i][j]=0;
                for(k=0;k<n;k++)
                    c[i][j]=c[i][j]+a[i][k]*b[k][j];
            }
        printf("\n A- matrix is\n");
    }
}
```

```
for(i=0;i<m;i++)
{
for(j=0;j<n;j++)
printf("%d\t",a[i][j]);
printf("\n");
}
printf("\n B- matrix is\n");
for(i=0;i<p;i++)
{
for(j=0;j<q;j++)
printf("%d\t",b[i][j]);
printf("\n");
}
printf("The product of two matrices is\n");
for(i=0;i<m;i++)
{
for(j=0;j<q;j++)
printf("%d\t",c[i][j]);
printf("\n");
}
}
getch();
}
```

OUTPUT:

```
Enter the size of first matrix
2 3
Enter the size of second matrix
3 2
Enter the elements of first matrix
1 2 3 4 5 6
Enter the elements of the second matrix
1 2 3 4 5 6
A- matrix is
1    2    3
```

4 5 6

B - matrix is

1 2

3 4

5 6

The product of two matrices is

22 28

49 64

RESULT: -Thus, the program to implement matrix multiplication has been executed successfully and the output was verified.

VIVA QUESTIONS: -

- a. How to initialize two dimensional arrays?
- b. How to pass a two dimensional array as function parameter?
- c. How the memory is allocated for two dimensional array
- d. Write the program to add and subtract two matrices.
- e. Program to find the transpose of a matrix.
- f. Program to find determinants of a matrix.
- g. Program to find the diagonal elements of a matrix.

Laboratory Program 7

COMPUTE SIN(X)/COS(X) USING TAYLOR SERIES APPROXIMATION. COMPARE YOUR RESULT WITH THE BUILT-IN LIBRARY FUNCTION. PRINT BOTH THE RESULTS WITH APPROPRIATE INFERENCES.

AIM: To develop a program to compute $\sin(x)/\cos(x)$ using taylor series approximation and compare the result with built-in function.

ALGORITHM:

Step-1: Start

Step-2: [Read the value of x in degree]- Read x

Step-3: [Initialization and Radians Conversion]

Temp = x

$x = x * (3.142 / 180.0)$

Term = x

sinx = term

n=1

Step-4: [Computer sin value]

Repeat while (term > FLT_EPSILON)

Fact = $2 * n * (2 * n + 1)$

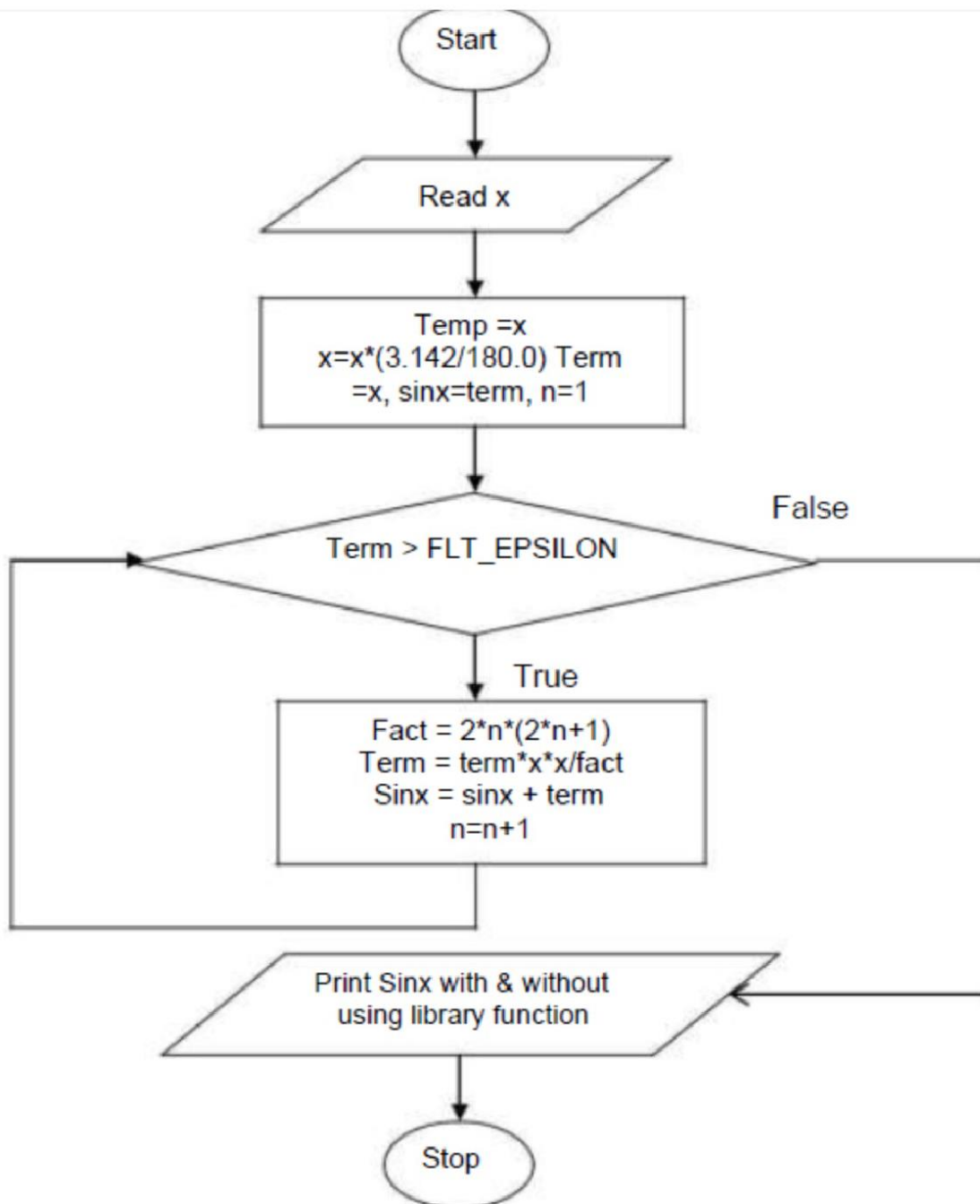
Term = term * x * x / fact Sinx

sinx + term = n + 1n

Step-5: [Output]- Print $\sin(x)$ without using library function of Print $\sin(x)$ and with using library function

Step-6: Stop

FLOWCHART



PROGRAM

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<math.h>
int fact(int m)
{
    int i,f=1;
    for(i=1;i<=m;i++)
    {
        f=f*i;
    }
    return f;
}
void main()
{
    int x,n,i;
    float rad, res, sum=0;
    clrscr();
    printf("Enter degree\n");
    scanf("%d",&x);
    printf("Enter number of terms\n");
    scanf("%d",&n);
    rad=x*3.14/180;
    for(i=1;i<=n;i+=2)
    {
        if ((i-1)%4==0)
            sum=sum+pow(rad,i)/fact(i);
        else
            sum=sum-pow(rad,i)/fact(i);
    }
}
```



```
}  
printf("Calculate sin(%d) = %f", x,sum);  
printf("\nLibrary sin(%d) = %f", x,sin(rad));  
getch();  
}
```

OUTPUT

First Run:

Enter degree

30

Enter number of terms

5

Calculate sin(30) = 0.499772

Library sin(30) = 0.499770

Second Run:

Enter degree

60

Enter number of terms

2

Calculate sin(60) = 0.866029

Library sin(60) = 0.865760

RESULT: -Thus, the program to calculate sin(x) has been executed successfully and the output was verified.

VIVA QUESTIONS: -

- a. What is pre-processor directive?
- b. What is difference between const and #define.
- c. What is use of fabs().
- d. What is variable initialization and why is it important?
- e. What is the difference between the = symbol and == symbol?
- f. Can the curly brackets { } be used to enclose a single line of code?

Laboratory Program 8

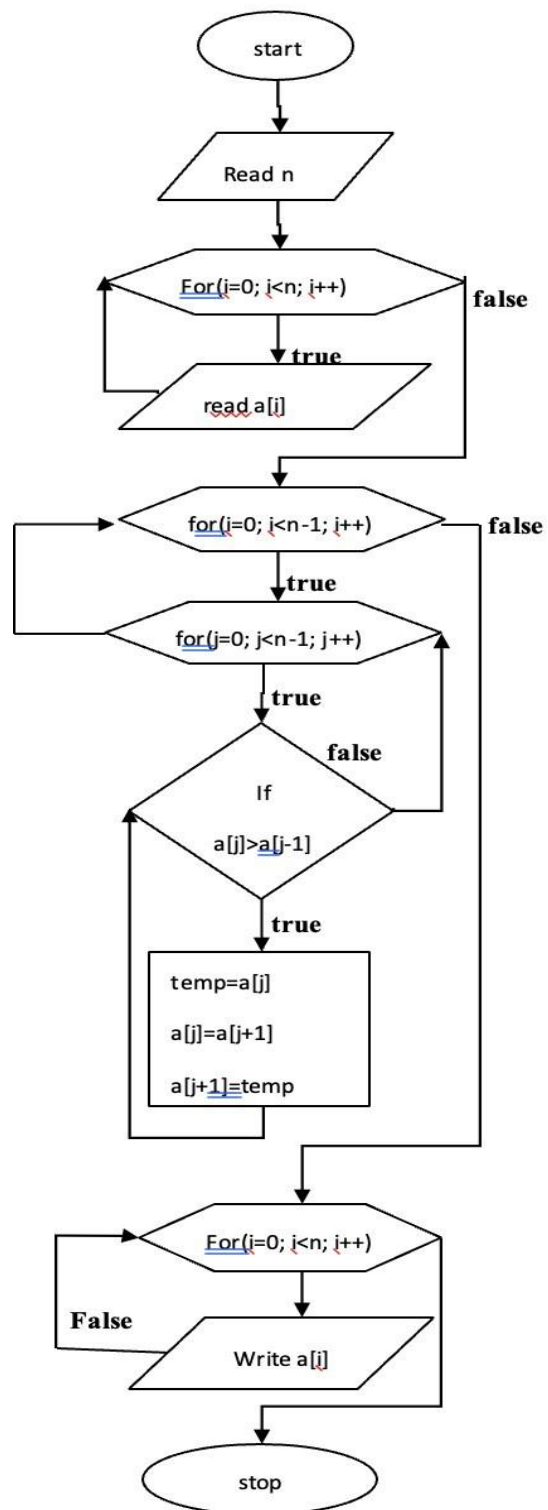
SORT THE GIVEN SET OF N NUMBERS USING BUBBLE SORT.

AIM: To develop a program to sort the given set of N numbers using Bubble sort.

ALGORITHM

- Step 1. [Initialize] Start
2. [Input number of elements] read n
3. [Input unsorted elements in array] read elements in array $a[]$
4. print elements of array $a[]$
5. [Iterate array $a[]$ in two loops.
Outer loop gives number of passes.
Inner loop does swap task. In each pass, compare each pair of adjacent items.
If former element is greater than latter one, swap them.
[Iterate array $a[]$ with
for each value i in array $a[i]$ to n **do**
for each value j in array $a[j]$ to $n-1$ **do**
 [Compare each pair of adjacent elements]
 if ($a[j] > a[j+1]$)**then**
 [Swap these elements using $temp$ variable]
 $temp \leftarrow a[j]$
 $a[j] \leftarrow a[j+1]$
 $a[j+1] \leftarrow temp$
 endif
end for
end for
6. Print array with sorted elements
7. [Finished] End.

FLOWCHART



PROGRAM

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int n,i,j,a[10],temp;
    clrscr();
    printf("Enter the no. of elements : \n");
    scanf("%d",&n);
    printf("Enter the array elements \n");
    for(i = 0 ; i < n ; i++)
        scanf("%d",&a[i]);
    printf("The original elements are \n");
    for(i = 0 ; i < n ; i++)
        printf("%d ",a[i]);
    for(i = 0 ; i < n-1 ; i++)
    {
        for(j = 0 ; j < (n-i)-1; j++)
        {
            if(a[j] > a[j+1])
            {
                temp = a[j];
                a[j] = a[j+1];
                a[j+1] = temp;
            }
        }
    }
    printf("\n The Sorted elements are \n");
    for(i = 0 ; i < n ; i++)
        printf("%d ",a[i]);
    getch();
}
```

OUTPUT

First Run:

Enter the no. of elements : 5

Enter the array elements

30 10 50 20 40

The original elements are

30 10 50 20 40

The Sorted elements are

10 20 30 40 50

Second Run:

Enter the no. of elements : 6

Enter the array elements

6 5 4 3 2 1

The original elements are

6 5 4 3 2 1

The Sorted elements are

1 2 3 4 5 6

RESULT: -Thus, the program to sort the elements using bubble sort has been executed successfully and the output was verified.

VIVA QUESTIONS: -

- a. Why the name bubble sort?
- b. What are the different types of sorting techniques?
- c. Explain the logic of bubble sort with an example.
- d. What is nested for loop?

Laboratory Program 9

WRITE FUNCTIONS TO IMPLEMENT STRING OPERATIONS SUCH AS COMPARE, CONCATENATE, STRING LENGTH. USE THE PARAMETER PASSING TECHNIQUES.

AIM: To write functions to implement string operations such as compare, concatenate, string length.

ALGORITHM

Step 1: Start

Step 2: [Input two source strings]

 read source1,source2

Step 3: [Calculate length1 of source1 by calling the user defined function, strlen();

 Repeat the same for length2 of source2]

 length1=strlen(source1)

 length2=strlen(source2)

Step 4: [Output length1,length2]

 Print length1,length2

Step 5: [Compare the two strings by calling the user defined function, strcmp()]

 k=strcmp(source1,source2)

Step 6: [check k, to find the whether the strings are same or not]

 if(k==0)

 print Both strings are same

 else

 print strings are different

 end if

Step 7: [Concatenate two strings by calling the user defined function, strcat() and the concatenated string is stored in source1]

 strcat(source1,source2)

 print source1

Step 8: Stop

User Defined Function - strlen()

Step 1: Start

Step 2: [set i=0]

```
i=0
```

Step 3: [receive the source string as str, read character by character, count one by one until we reach NULL character]

```
while(str[i]!='\0')
```

```
i++ end while
```

Step 4: [return i to the calling function]

```
return i
```

User Defined Function - strcmpare()

Step 1: Start

Step 2: [set i=0]

```
i=0
```

Step 3: [Receive both the source strings as str1 and str2, read character by character until they match, if the matched character is a NULL then go out of while loop, when any unmatched character then go out of loop]

```
while(str1[i] == str2[i])
```

```
if(str1[i] == '\0')
```

```
break
```

```
end if
```

```
i++
```

```
end while
```

Step 4: [calculate k]

```
k=str1[i]-str2[j]
```

Step 5: [return i to the calling function]

```
return k
```

User Defined Function - strconcat()

Step 1: Start

Step 2: [set i=0]

```
i=0
```

Step 3: [Receive both the source strings as str1 and str2, calculate length of str1 using strlen() as l]

```
l=strlength(str1)
```

Step 4: [read str2 character by character and store it from the end of str1]

```
while(str2[i]!='\0')
```

```
str1[l+i]=str2[i]
```

```
i++
```

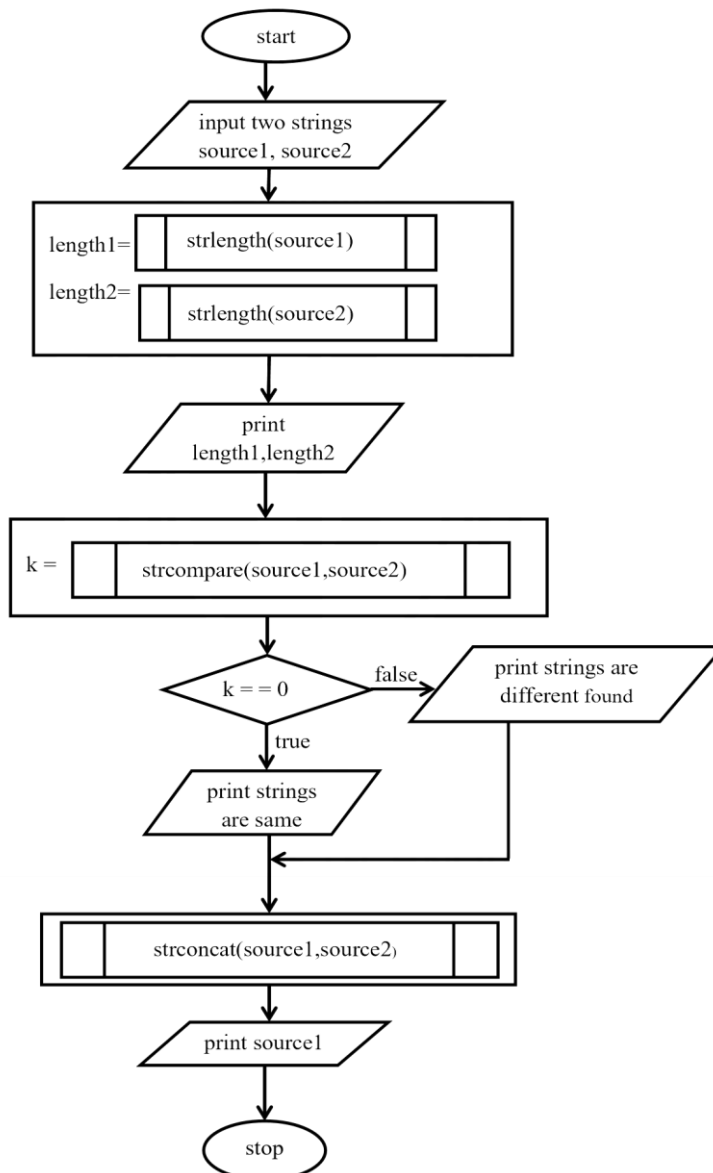
```
end while
```

Step 5: [return to the calling function]

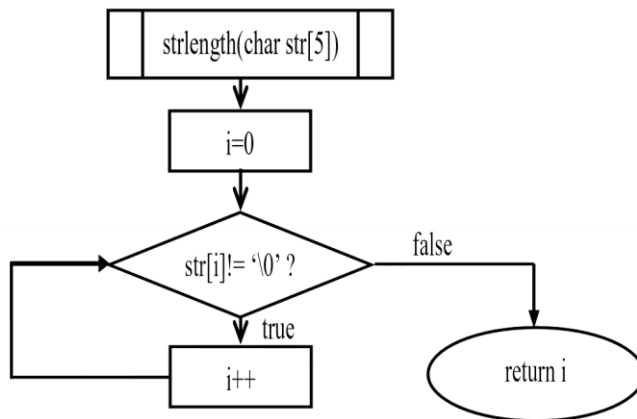
```
return;
```

```
return
```

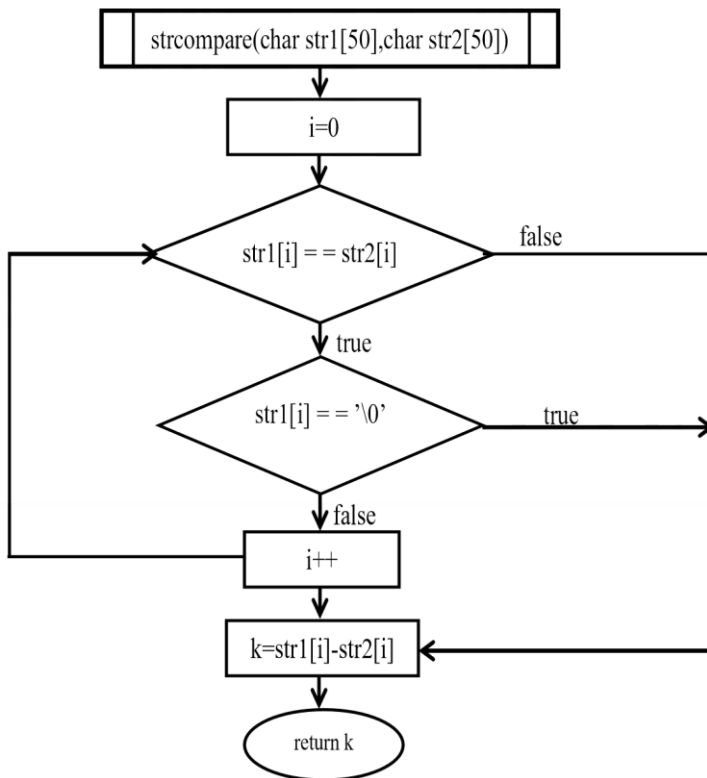
Flowchart:

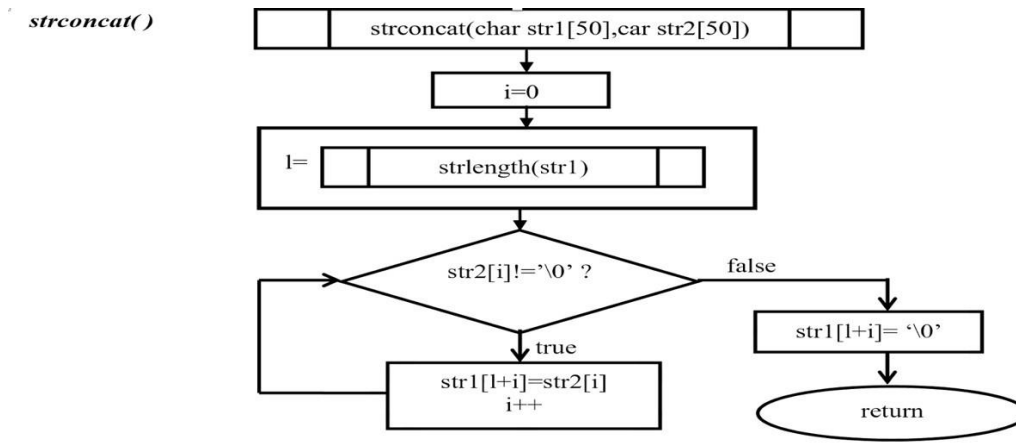


strlen()



strcmp()





PROGRAM

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

```
#include<conio.h>
```

```
int strlen(char str[50]);
```

```
void strconcat(char str1[50],char str2[50]);
```

```
int strcmp(char str1[50],char str2[50]);
```

```
int strlen(char str[50])
```

```
{
```

```
int i=0;
```

```
while(str[i]!='\0')
```

```
i++;
```

```
return i;
```

```
}
```

```
void strconcat(char str1[50],char str2[50])
```

```
{
```

```
int i=0,l;
```

```
l=strlen(str1);
```

```
while(str2[i]!='\0')
```

```
{
```

```
str1[l+i]=str2[i];
```

```
i++;
}
str1[l+i]='\0';
}
int strcmpare(char str1[50],char str2[50])
{
int i=0, k;
while(str1[i]==str2[i])
{
if(str1[i]=='\0')
break;
i++;
}
k=str1[i]-str2[i];
return k;
}
void main()
{
char source1[50],source2[50],dest[50];
int length1,length2,k;
clrscr();
printf("\n Enter the source string 1:");
gets(source1);
printf("\n Enter the source string 2:");
gets(source2);
length1=strlength(source1);
length2=strlength(source2);
printf("\n string length of string 1 is %d",length1);
printf("\n string length of string 2 is %d",length2);
k=strcmpare(source1,source2);
if(k==0)
printf("\n Both string are same");
```

```
else
printf("\n Both string are different");
strconcat(source1,source2);
printf("\n concatenated string is ");
puts(source1);
getch();
}
```

OUTPUT

First Run:

```
Enter the source string1: good
Enter the source string2: night
String length of string1 is: 4
String length of string2 is: 5
strings are different
concatenated string is: goodnight
```

Second Run:

```
Enter the source string1: good
Enter the source string2: good
String length of string1 is: 4
String length of string2 is: 4
Both strings are same
concatenated string is: goodgood
```

RESULT: - Thus, the program to implement string operations such as compare, concatenate, string length has been executed successfully and the output was verified.

VIVA QUESTIONS: -

- a. What is string?
- b. How to declare string?
- c. What are the string manipulation function?
- d. What is gets() and puts() function in string?

Laboratory Program 10

IMPLEMENT STRUCTURES TO READ, WRITE AND COMPUTE AVERAGE- MARKS OF THE STUDENTS, LIST THE STUDENTS SCORING ABOVE AND BELOW THE AVERAGE MARKS FOR A CLASS OF N STUDENTS.

AIM: To write a C Program to implement structures to read, write and compute average-marks of the students, list the students scoring above and below the average marks for a class of N students.

ALGORITHM

Step-1: Start

Step-2: Read number of students

Step-3: For every student, read the student id, name , marks for all the subjects

Step-4: Calculate the average marks and store it in the avg field

Step-5: Print the results

Step-6: Initialise loop

Step-7: Read the average of every student

Step-8: Check for if $avg > 35.00$

Step-9: If yes than print the result else goto next iteration

Step-10: Initialise the loop

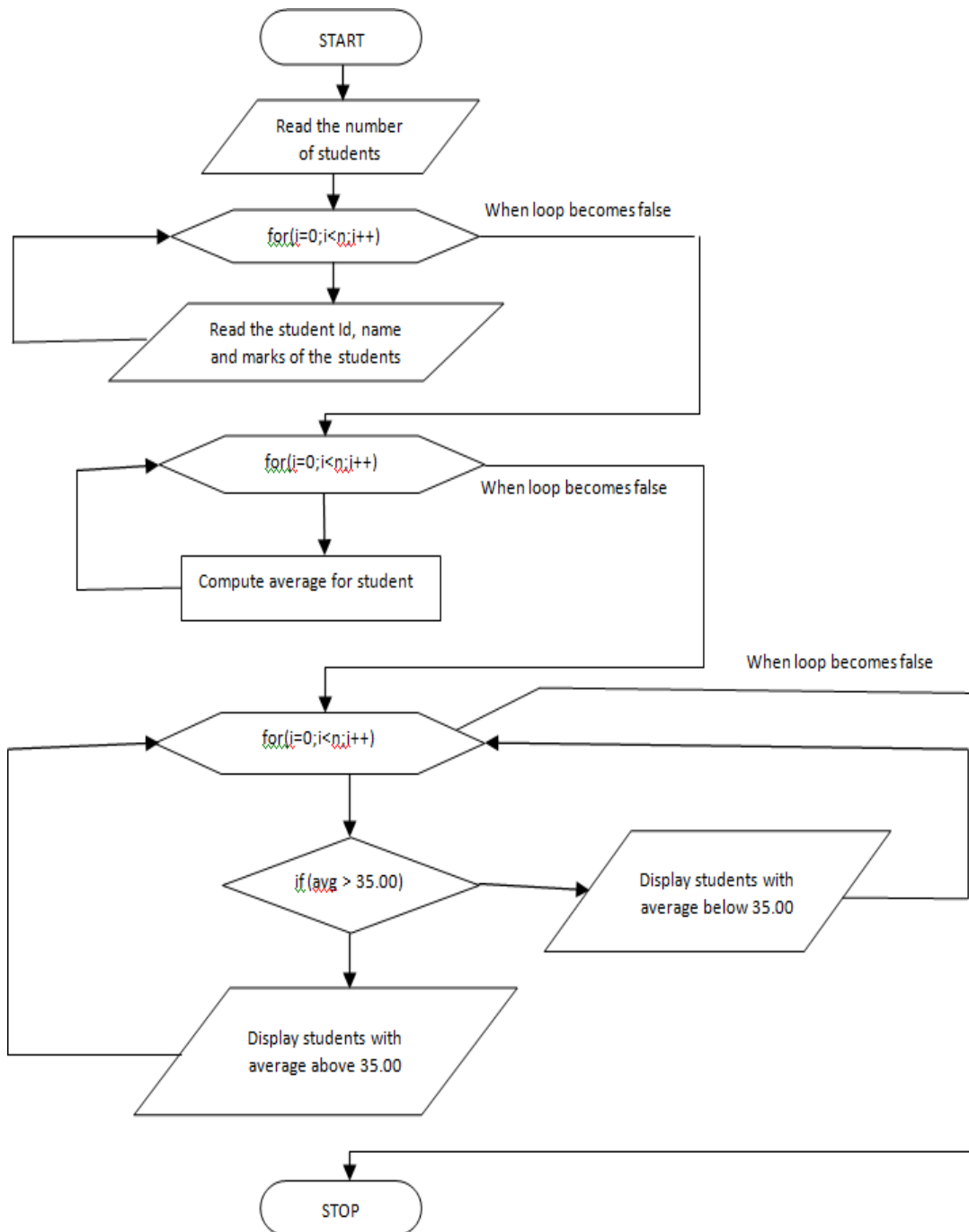
Step-11: Read average of every student

Step-12: Check if $avg < 35.00$

Step-13: If yes than print result else goto next iteration

Step-14: Stop

FLOWCHART



PROGRAM

```
#include<stdio.h>
#include<conio.h>
struct student
{
char usn[10];
char name[10];
int m1,m2,m3;
float avg, total;
};
void main()
{
struct student s[20];
int n,i;
float tavg,sum=0.0;
clrscr();
printf("Enter the number of students");
scanf("%d",&n);
for(i=0;i<n;i++)
{
printf("Enter the detail of %d students\n",i+1);
printf("\n Enter USN=");
scanf("%s",s[i].usn);
printf("\n Enter Name=");
scanf("%s",s[i].name);
printf("\nEnter the three subjects marks\n");
scanf("%d%d%d",&s[i].m1,&s[i].m2,&s[i].m3);
s[i].total=s[i].m1+s[i].m2+s[i].m3;
s[i].avg=s[i].total/3;
}
for(i=0;i<n;i++)
{
```

```
if(s[i].avg>=35)
printf("\n %s has scored above the average marks",s[i].name);
else
printf("\n %s has scored below the average marks",s[i].name);
}
getch();
}
```

OUTPUT

Enter the number of students2

Enter the detail of student 1

Enter USN=1

Enter Name=Arun

Enter the three-subject score

23 45 67

Enter the detail of student 2

Enter USN=2

Enter Name=Tharun

Enter the three-subject score

5 3 2

Arun has scored above the average marks

Tharun has scored below the average marks

RESULT: -Thus, the program to implement structures to read, write and compute average- marks of the students, list the students scoring above and below the average marks for a class of N students has been executed successfully and the output was verified.

VIVA QUESTIONS: -

- a. What is structure?
- b. How to declare a structure?
- c. What is structure member?
- d. What is difference between array and structure?
- e. What is nested structure?
- f. What is typedef?

Laboratory Program 11

DEVELOP A PROGRAM USING POINTERS TO COMPUTE THE SUM, MEAN AND STANDARD DEVIATION OF ALL ELEMENTS STORED IN AN ARRAY OF N REAL NUMBERS.

AIM: To Develop a C program using pointers to compute the sum, mean and standard deviation of all elements stored in an array of n real numbers.

ALGORITHM

Step-1: Start

Step-2: Read n

Step-3: For every value of n read the x

Step-4: Initialize sum=0 and i=0

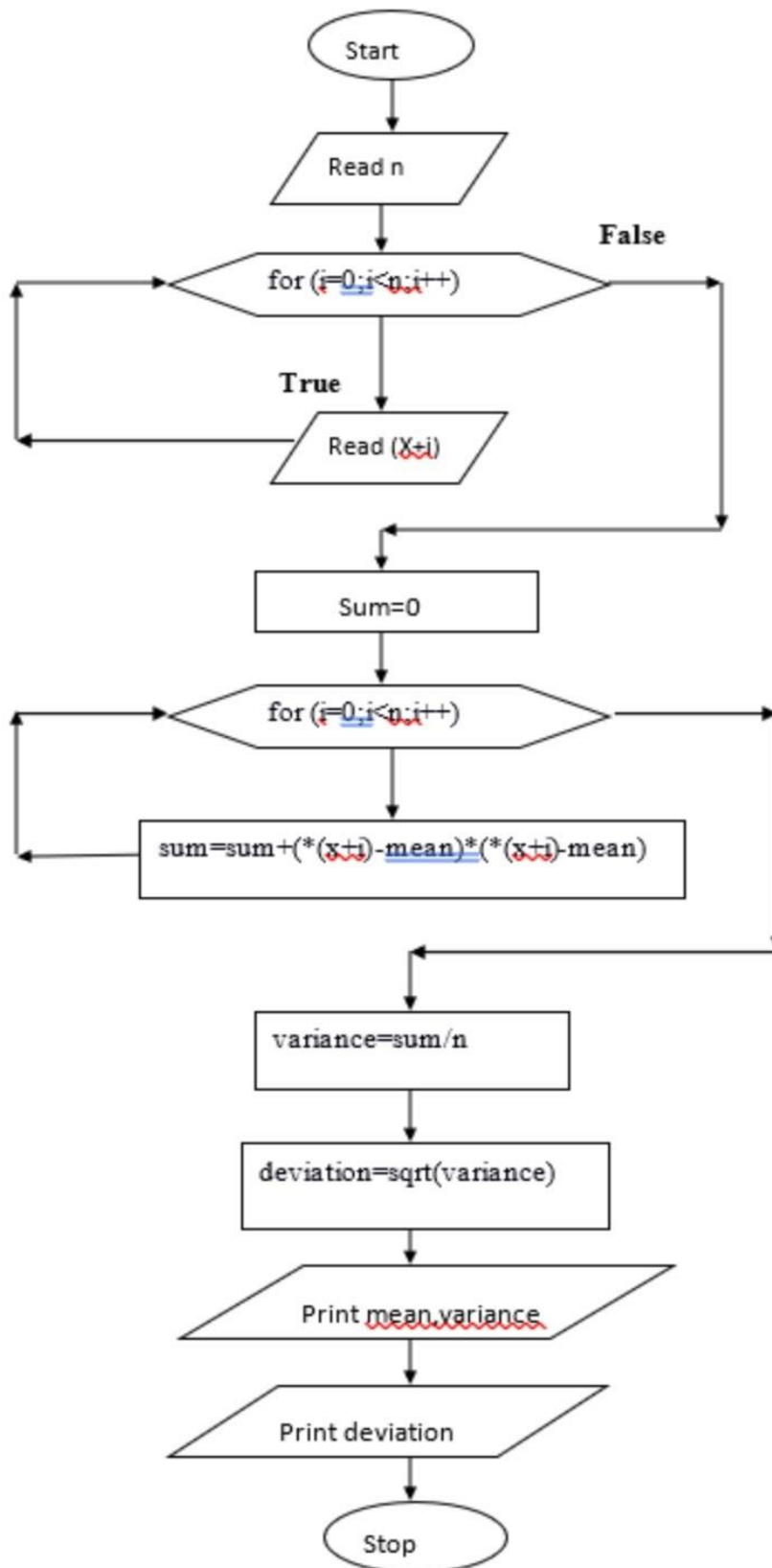
Step-5: For every value of n and i, compute sum using $sum = sum + (x+i) - mean$ * ($x+i) - mean$)

Step-6: Using the sum value compute variance = sum / n and deviation = $\sqrt{variance}$)

Step-7: Display mean, variance, deviation

Step-8: Stop

FLOWCHART



PROGRAM

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
int main()
{
int n , i;
float x[20],sum,mean;
float variance , deviation;
clrscr();
printf("Enter the value of n \n");
scanf("%d",&n);
printf("enter %d real values \n",n);
for (i=0;i<n;i++)
{
scanf("%f",(x+i));
}
sum=0;
for(i=0;i<n;i++)
{
sum= sum+*(x+i);
}
printf("sum=%f\n",sum);
mean=sum/n;
sum=0;
for(i=0;i<n;i++)
{
sum=sum+(*(x+i)-mean)*(*(x+i)-mean);
}
variance=sum/n;
deviation=sqrt(variance);
printf("mean(Average)=%f\n",mean);
```

```
printf("variance=%f\n", variance);  
printf("standard deviation=%f\n", deviation);  
getch();  
}
```

Output:

Enter the value of n

5

Enter the 5 real values

3

7

23

1

4

Sum = 38.0000

Mean (Average) = 7.6000

Variance = 63.039997

Standard deviation = 7.9397

RESULT: -Thus, the program to implement pointers to compute the sum, mean and standard deviation of all elements stored in an array of n real numbers has been executed successfully and the output was verified.

VIVA QUESTIONS:-

- a. Define pointer
- b. Define array of pointer
- c. Difference between $(x+i)$ and $*(x+i)$
- d. Define array

Laboratory Program 12

WRITE A C PROGRAM TO COPY A TEXT FILE TO ANOTHER, READ BOTH THE INPUT FILE NAME AND TARGET FILE NAME.

AIM: To Write a C program to copy a text file to another, read both the input file name and target file name.

ALGORITHM

Step 1 : Start

Step 2: Read the source file name fname1

Step 3: Open the file fname1 in read mode

Step 4: if fptr1 is equal to NULL

 print " File does not found or error in opening.!!"

 goto Step 12

Step 5: Read the new file name fname2

Step 6: Open the file fname2 in write mode

Step 7: if fptr2 is equal to NULL

 print " File does not found or error in opening.!!"

 goto Step 12

Step 8: Repeat while(1)

 ch=fgetc(fptr1);

 if ch is equal to EOF

 break;

 else

 fputc(ch, fptr2);

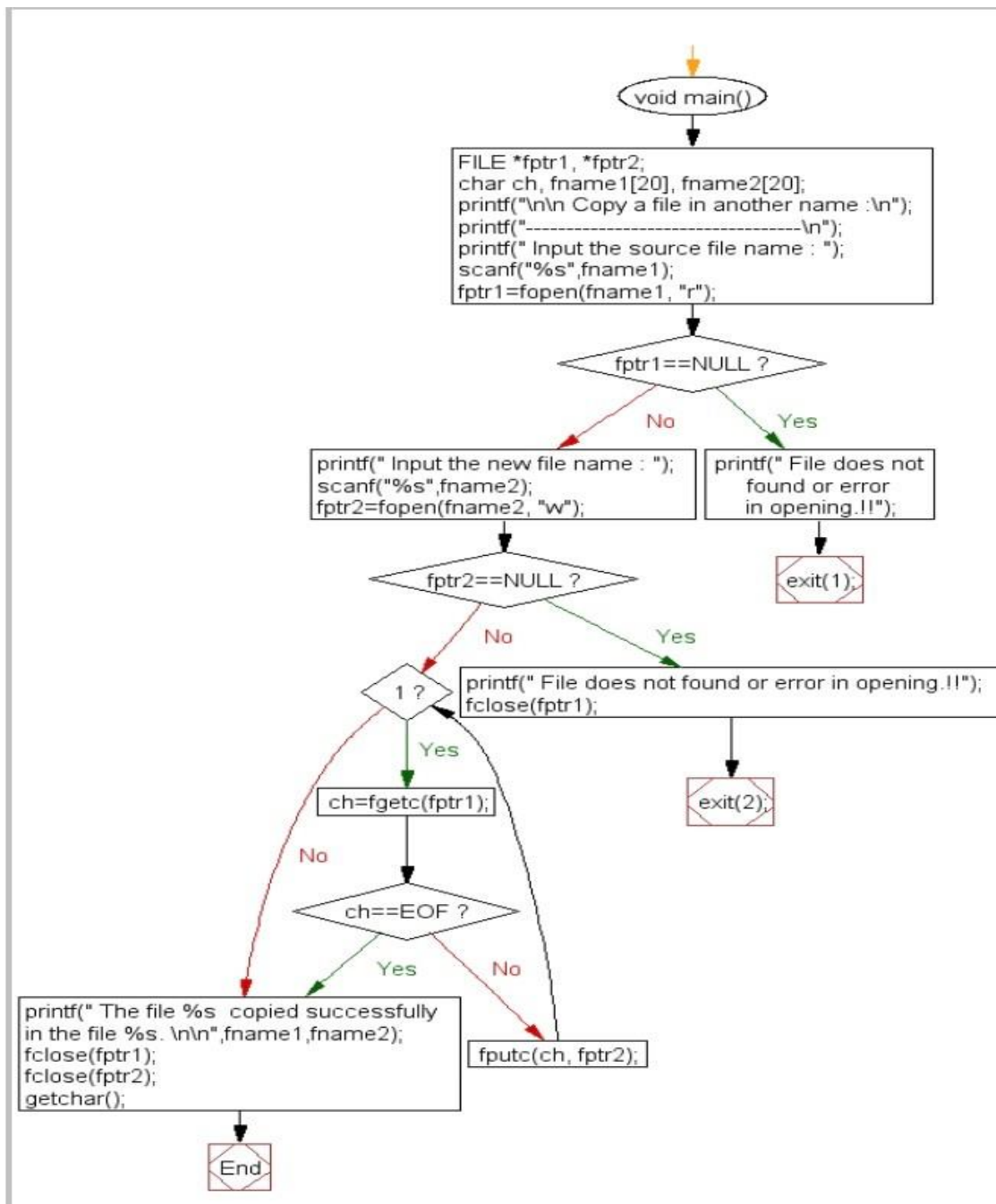
Step 9: print "The file fname1 copied successfully in the file fname2"

Step 10: close file pointer fptr1

Step 11: close file pointer fptr2

Step 12: Stop

FLOWCHART



PROGRAM

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
void main()
{
    FILE *fptr1, *fptr2;
    char ch, fname1[20], fname2[20];
    clrscr();

    printf("\n\n Copy a file in another name :\n");
    printf(".....\n");

    printf(" Input the source file name : ");
    scanf("%s",fname1);

    fptr1=fopen(fname1, "r");
    if(fptr1==NULL)
    {
        printf(" File does not found or error in opening.!!");
        exit(1);
    }
    printf(" Input the new file name : ");
    scanf("%s",fname2);
    fptr2=fopen(fname2, "w");
    if(fptr2==NULL)
    {
        printf(" File does not found or error in opening.!!");
        fclose(fptr1);
        exit(2);
    }
    while(1)
    {
        ch=fgetc(fptr1);
        if(ch==EOF)
        {
            break;
        }
        else
```

```
        {
            fputc(ch, fptr2);
        }
    }
    printf(" The file %s copied successfully in the file %s. \n\n",fname1,fname2);
    fclose(fptr1);
    fclose(fptr2);
    getchar();
}
```

OUTPUT:

Copy a file in another name :

Input the source file name : test.txt

Input the new file name : test1.txt

The file test.txt copied successfully in the file test1.txt.

RESULT: - Thus, the program to copy a text file to another, read both the input file name and target file name has been executed successfully and the output was verified.

VIVA QUESTIONS:-

- a. Define file.
- b. What is fopen() and fclose() in file?
- c. What is fgetc() and fputc() in file?