KAMMAVARI SANGHAM (R) - 1952

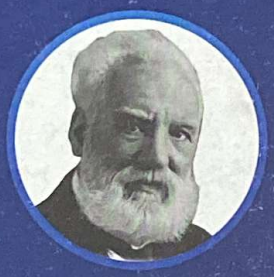
K S GROUP OF INSTITUTIONS

K. S. INSTITUTE OF TECHNOLOGY

Accredited by NAAC

(Approved by AICTE & Affiliated to VTU)

#14, Raghuvanahalli, Kanakapura Main Road, Bengaluru - 560109



GRAHAM BELL

A P J ABDUL KALAM

MARIE CURIE

SRINIVASA RAMANUJAN

DENNIS RITCHIE

J R D TATA

THOMAS ALVA EDISION

"Tell me and I'll forget; Show me and I may remember; Involve me and I'll understand"

PRACTICAL RECORD

NAME : Rupa Rajasri Puthineedi

SEM / BRANCH : 3rd / AIML

SUBJECT & CODE : CPL / 18CPL27

USN :

1	K	S	2	0	A	I	0	3	4

KAMMAVARI SANGHAM (R) - 1952

K S GROUP OF INSTITUTIONS

K. S. INSTITUTE OF TECHNOLOGY

Accredited by NAAC
(Approved by AICTE & Affiliated to VTU)
#14, Raghuvanahalli, Kanakapura Main Road, Bengaluru - 560109

Laboratory Certificate

This is to certify that Mr./Ms. Rupa Rajasri Puthineedi has satisfactorily completed the course of experiments in C Programming laboratory, Code ..... prescribed by Visvesvaraya Technological University, Belagam for the SECOND Semester B.E AI and ML Branch in this college during the academic year 20.20.. - 20.24..

Name of the Candidate : Rupa Rajasri Puthineedi

USN : 1KS20A1034 Subject (with code) ~~18CPS23~~ 18CPL27

Internal assessment marks awarded :

30		8		38	
30	+	10	=	40	

Signature of Staff Incharge

Date : 14/09/21

14/9/21

Signature of Head of Department

Head of the Department
Dept. of Computer Science & Engg
K.S. Institute of Technology
Bengaluru -560 109

INSTRUCTIONS FOR MAINTAINING THE PRACTICAL RECORD

1. Record should be written neatly in black ball point pen on the right side page only, left side pages being reserved for diagrams and graphs in pencil.

2. Record should contain

   - Number and name of Experiment
   - The date
   - Principle
   - Procedure
   - Observation calculation (to be entered on the left hand side pages in neat tabular forms wherever applicable)
   - Results

CONTENTS AND EVALUATION

Sl.No.	Date of Conducting Experiment	Date of Submission	Page No.	Title of the Experiment	Assessment Marks				Signature Staff with Date
					Experiment 10 Marks	Record 10 Marks	Viva (10)	Total	
1.	1/09/21	8/09/21	1-20	Introduction to Hardware	10	10	10	30	14/09/21
2.	1/09/21	8/09/21	21-26	Laboratory Session - II	10	10	10	30	14/09/21
3.	1/09/21	8/09/21	27-28	Simple Commercial Calculator	10	10	10	30	14/09/21
4.	1/09/21	8/09/21	29-31	Quadratic Equation	10	10	10	30	14/09/21
5.	6/09/21	8/09/21	32-33	Check for Palindrome or Not	10	10	10	30	14/09/21
6.	6/09/21	8/09/21	34-36	Electricity Bill	10	10	10	30	14/09/21
7.	6/09/21	8/09/21	37-39	1D Array Manipulation	10	10	10	30	14/09/21
8.	6/09/21	8/09/21	40-42	Check whether the given number is Prime	10	10	10	30	14/09/21
9.	6/09/21	12/09/21	43-46	2D Array Manipulation	10	10	10	30	14/09/21
10.	6/09/21	12/09/21	47-48	Compute sin(x) using Taylor series approximation	10	10	10	30	14/09/21
11.	9/09/21	12/09/21	49-53	Implement string operations to concatenate string length	10	10	10	30	14/09/21
12.	9/09/21	12/09/21	54-56	Bubble Sort	10	10	10	30	14/09/21
13.	9/09/21	12/09/21	57-58	Find square root of N and execute	10	10	10	30	14/09/21

CONTENTS AND EVALUATION

Sl.No.	Date of Conducting Experiment	Date of Submission	Page No.	Title of the Experiment	Assessment Marks				Signature

13.	9/09/21	12/09/21	57-58	Find square root of N and execute	10	10	10	30	14/09/21

CONTENTS AND EVALUATION

Sl.No.	Date of Conducting Experiment	Date of Submission	Page No.	Title of the Experiment	Assessment Marks				Signature Staff with Date
					Experiment 10 Marks	Record 10 Marks	VI-VA 10M	Total	
14.	9/09/21	12/09/21	59-61	Compute avg, marks and scoring above / below avg	10	10	10	30	14/09/21
15.	9/09/21	12/09/21	62-64	Using pointers, computed sum, mean, sd and variance	10	10	10	30	14/09/21
16.	9/09/21	12/09/21	65-67	Binary to Decimal Conversion	10	10	10	30	14/09/21
									Observation (10/10)
									Record (10/10)
									Viva (10/10)
									Total (30/30)

DATE 1/09/21 | EXPT. TITLE : | PAGE NO. 1

EXP. NO. 1

Introduction to Hardware

Functional block diagram of Computer -

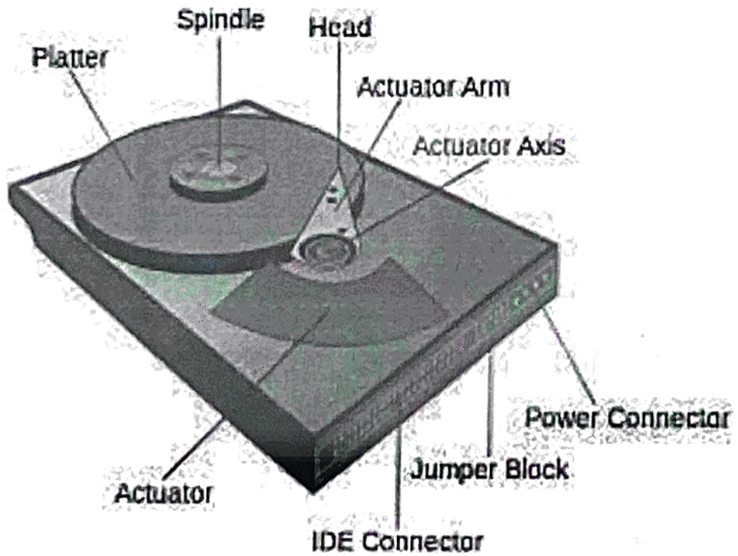
Description -

A computer is an electronic device which is capable of receiving information (data) in a particular form and of performing a sequence of operations in accordance with a predetermined but variable set of procedural instructions (program) to produce a result in the form of information or signals.

A computer can proces data, pictures, sound and graphics. They can solve highly complicated problems quickly and accurately.

A computer as shown in the below figure basically performs five major operations or functions irrespective of their size and make. These are -

* It accepts data or instructions by way of input.
* It stores data
* It can proces data as required by the user.
* It gives results in the form of output.
* It controls all operations inside the computer.



Secondary Storage (Hard Disc)

Primary Storage -

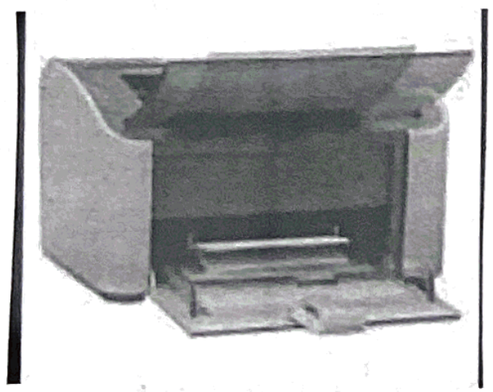
It is also known as main memory, it is the storage area that is directly accesible by the CPU at very high speeds. Data has to be fed into the system before the actual procesing starts. It is belause procesing speed of central Procesing Unit (CPU) is so fast that that the data has to be provided to CPU with the same speed. This storage unit or the primary storage of the computer system is designed to do the above functionality. It provides space for storing datd and instructions.

Secondary Storage -

It is also known as secondary memory or auxilary memory. It basically overcomes all the drawbacks of the primary storage area. It is cheaper, non-violatile and used to permanently store data and programs that are not being cursently executed by the CPU.

3) Processing -

The procen of performing operations on the data as per the instructions specified by the user (program) is called Procesing. Data and instructions are taken



from the primary memory and transferred to the arithmetic and logical Unit (ALU), which performs all sorts of calculations. The immediate results of procesing may be stored in the main memory, as they might be required again. When the processing completes, the final result in then transfetred to the main memory.

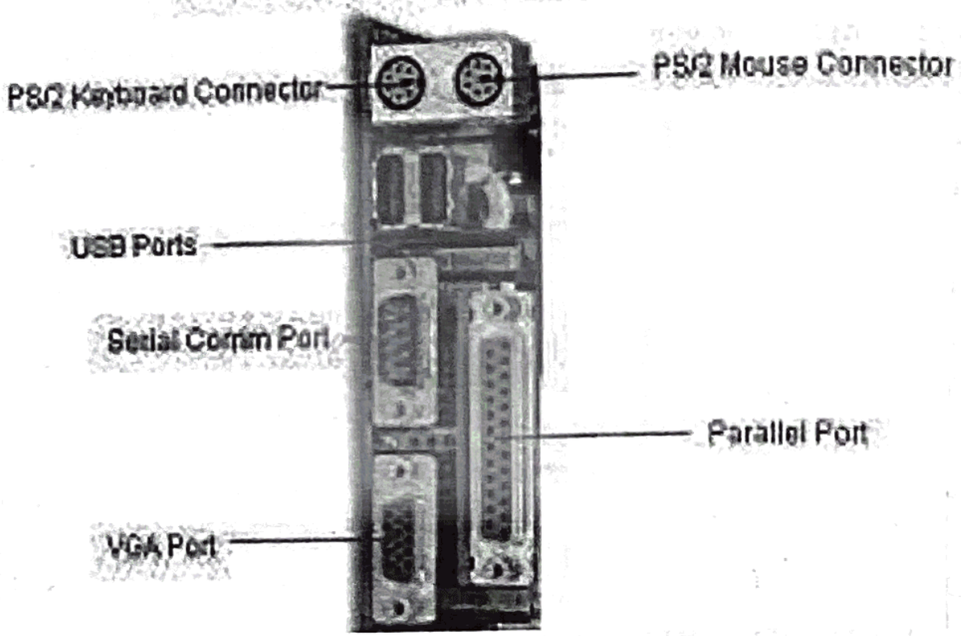
4) Output -

This is the proces of producing results from the data for getting useful information. Similarly the output produced by the computer after processing must also be kept somewhere inside the computer before being given to you in human reddable form. Again the output is also stored inside the computer for further procesing.

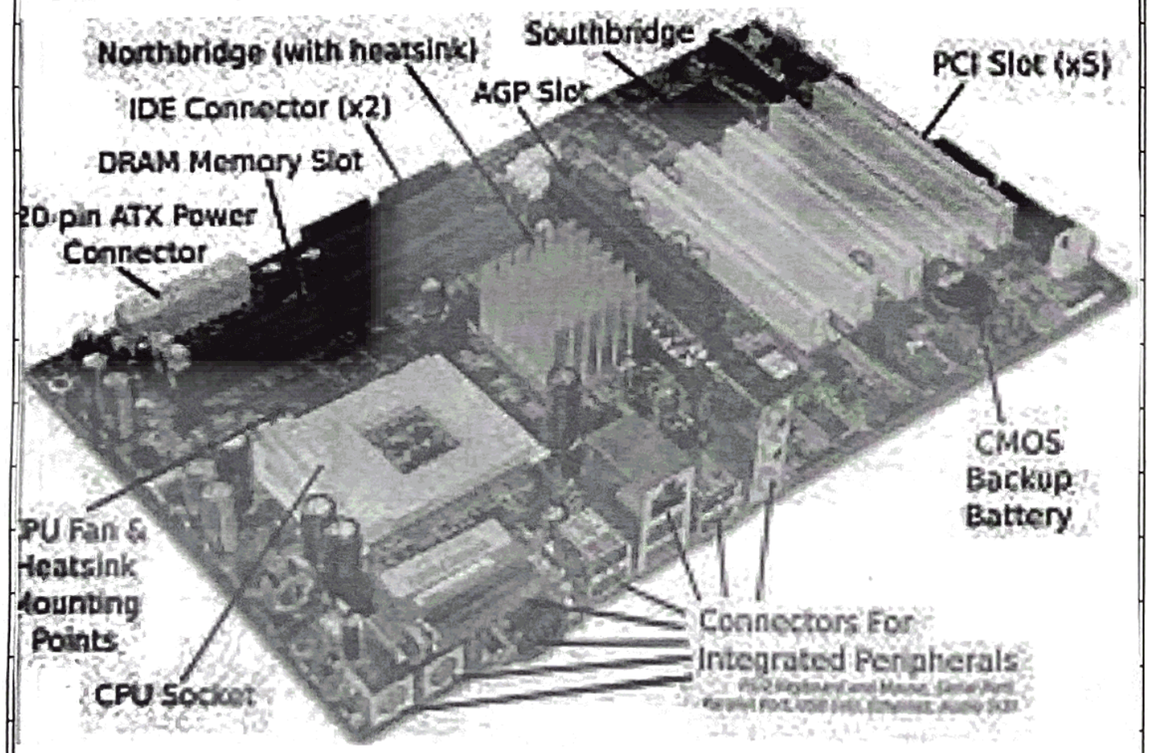
* Monitors
* Printers
* Speakers

5) Control Unit -

The Control Unit is the central nervous system of the entire computer. It manges and controls all the components of the computer system. Controlling of all



MOTHERBOARD



operations inside the computer.

Buses -

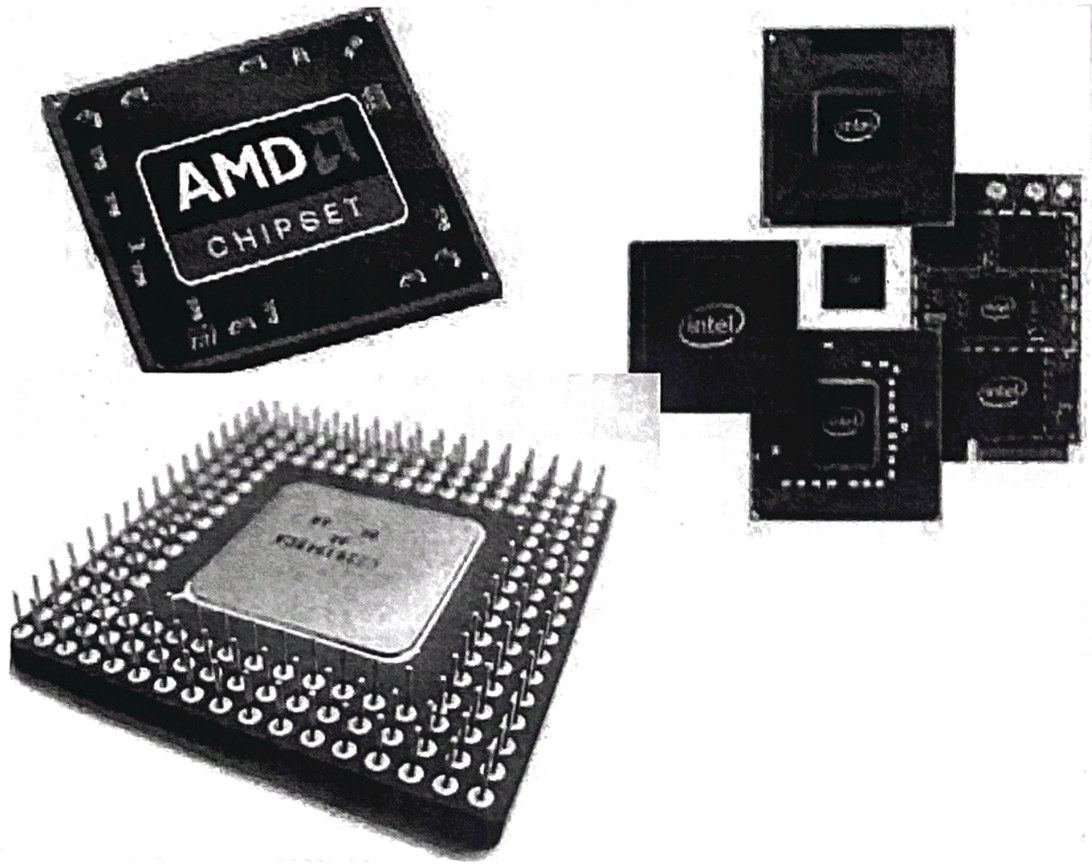
A bus is a collection of wires through which data is transmitted from one part of the computer to another. It connects physical components such as cables, printed circuits, CPU, memory, peripherals etc for sharing of information and communication with one another.

Types of Buses -

* Data Bus - It is used to transfer the data between procesor, and I/o devices. It is bidirectional in nature.
* Address Bus - It is used to transfer of data and instructions stored in memory. It is unidirectional in nature.
* Control Bus - It is used to transfer the control signals between CPU, memory and I/O devices. It is unidirectional or bidirectional in nature.

Mother Board -

The motherboard serves to connect all the parts of a computer together. The CPU, memory, hard drives, video card, sound card and other ports and expansion cards all connect to the motherboard directly or



via cables.

Chip -

A small piece of semiconducting material on which an integral circuit is embedded. A typical chip is less than square inches and can maintain contain millions of electronic components. Computers consists of many chips placed on electronic boards called Printed Circuit boards.
There are different types of chips. For example, CPU chips (microprocesors) contain an entire procesing unit, whereas the memory chips contain blank memory.

Chip Set -

A number of integrated circuits designed to perform one or more related functions. For example, one chipset may provide the basic functions of a modern while another provides the CPU functions for a computer. Newer chipsets generally include functions provided by two or more older chipsets. In some cases, older chipsets that required two or more physical chips can be replaced with a chipset on one chip.

DATE .....
EXP. NO. .....

EXPT. TITLE :

PAGE NO. 7

Operating System and Types of OS -

The operating system is system software that controls and supervises the hardware components of a computer system and it provides services to computer users. Every general-purpose computer must have an operating system to run other programs. Operating systems perform basic tasks, such as recognizing input from the keyboard, sending output to the display screen, keeping track of files and directories on the disk.

The four general kinds of operating systems are-

* Real time OS - This kind of OS controls machinery, industrial equipment and scientific instruments. Its main purpose is to ensure than an operation executes in exactly the same way, in the same amount of time, every time it happens.
* Multiuser - This OS lets many people do many things, all at the same time. It has to balance the needs of each other and keep them separate, so that they dont interfere with each other.
* Single user, single task - This kind of OS is designed so that a computer executes

a user's tasks one at a time, such as with early personal digital assistants.

* Single user - multitasking - Most PCs use this kind of OS, such as Windows or Mac OS. It lets a single user do many things at once.

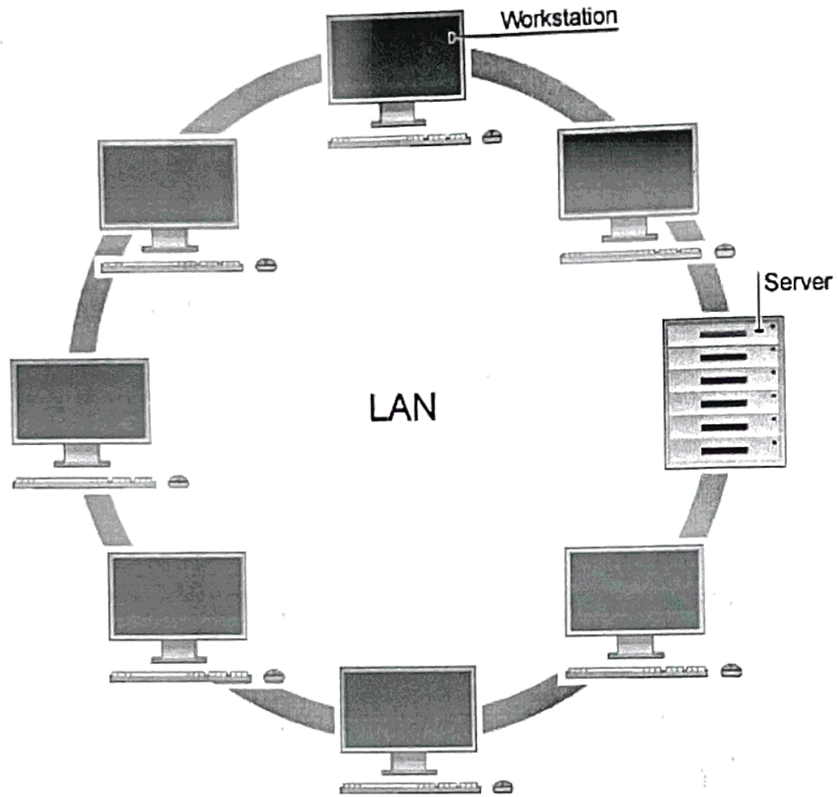
Basics of Networking -

A network is a group of two or more computer systems linked together. A network is a set of technologies including hardward, software and media that can be used to connect computers together, enabling them to communicate exchange information and share resources in real time.

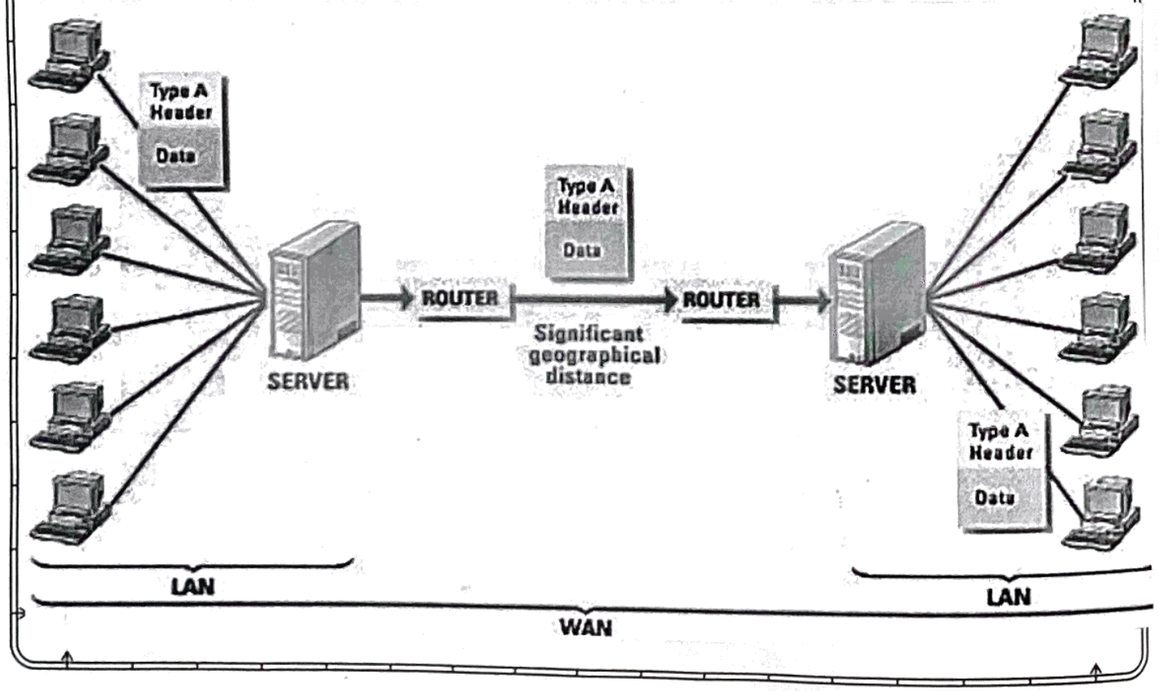
Benefits of a Network -

* Allows simultaneous accen to critical programs and data.
* Allows people to share peripheral devices, such as printers and scanners.
* Streamlines personal communication with email.
* Makes the backup proces easier.

LAN-



WAN-



DATE .....
EXP. NO. .....

EXPT. TITLE :

PAGE NO. 9

Types of Network -

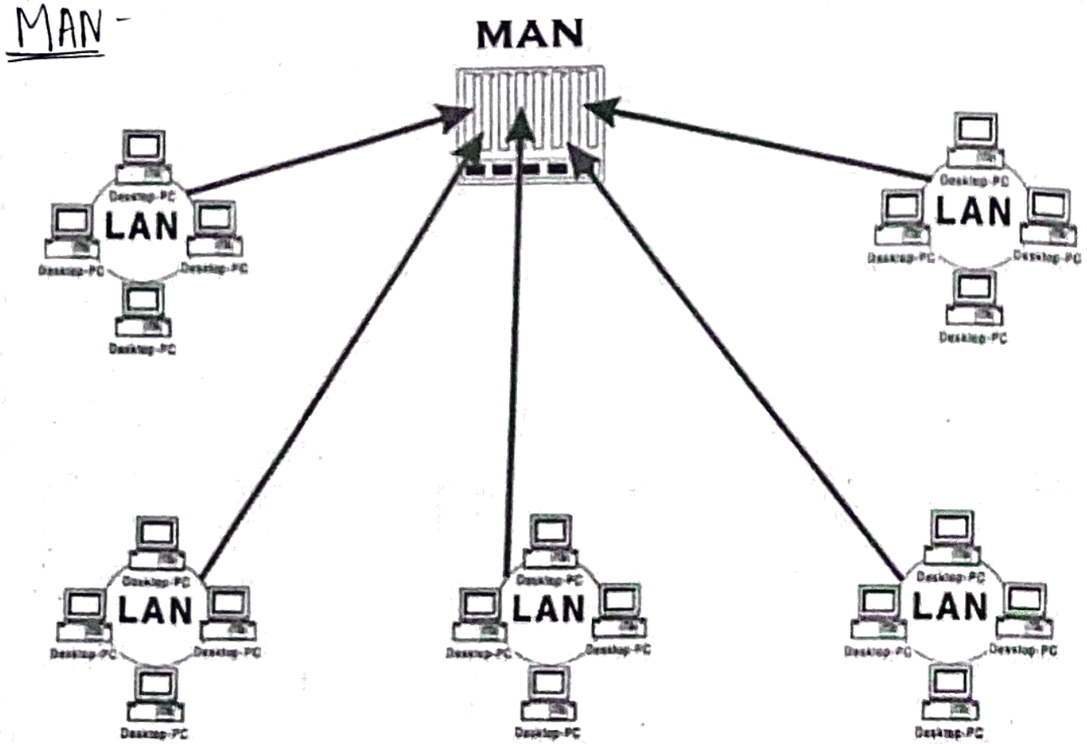
Local Area Network (LAN) -

A network of computers located relatively near each other and connected by a cable. A lan is a data communication system consisting of several devices such as computers and printers. A lan can consist of just two or three PC's connected together to share resources, or it can include hundreds of computers of different kinds. Any network that exists within a single building or even a group of adjacent buildings is called LAN.

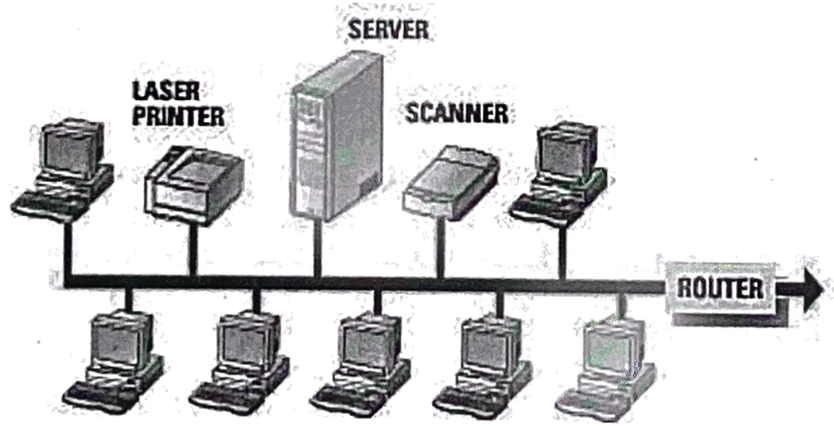
Wide Area Network (WAN) -

Two or more LAN's connected together generally across a wide geographical area using high-speed or dedicated telephone line. For example, a company may have its corporate headquarters and marketing office in another. Each site needs resources, data and programs locally, but it also needs to share data with the other sites. To accomplish this feat of data communication, the company attach devices that connect over public utilities to create a WAN.

MAN -



Bus -



DATE .....
EXP. NO. .....

EXPT. TITLE :

PAGE NO. 10

Metropolitan Area Network (MAN) -

The MAN is a large scale network that connects multiple corporate LAN together. MAN usually are not owned by a single organization, their communication devices and equipment are usually maintained by a group or single network provider that sells its networking services to coorporate customers. Similar to a WAN network but is confined to a single city or metropolitan area.

Topology -

The physical layout of the cables that connect the nodes of the network

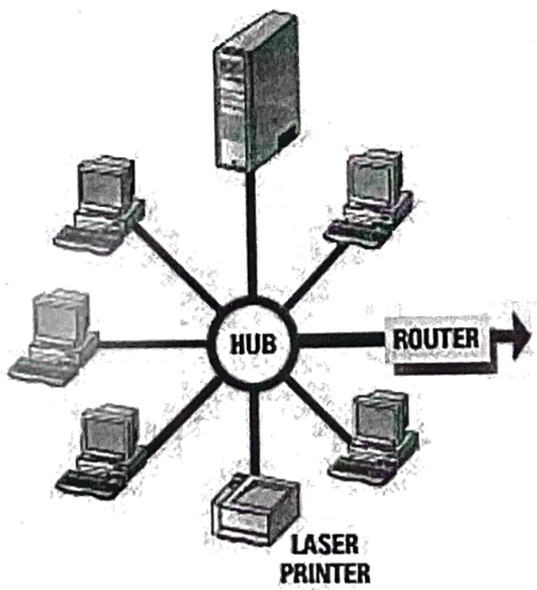
Buss Topology -

In this network structure, a single cable runs in a building or campus. All the nodes (terminals/computers) are connected to this single cable. It is suitable for LAN.

Advantages -

* Failure of one node will not affect the whole
* well suited for quick setup.
* Easy to install and expand
* High rate of data transmission as compared to star and ring topology.

Star -



DATE .....
EXP. NO. .....

EXPT. TITLE :

Disadvantages -

* A cable break can disable the entire network
* Trouble shooting is very difficult.
* Only a single menage can travel at a time.

Star Topology -

In this network, all the computers are connected with a centralized system called Server. The central computer is also called a hub. To transmit information from one node to another node, it should be transmitted through a central hub. The central hub mandges and controls all the functions of network.

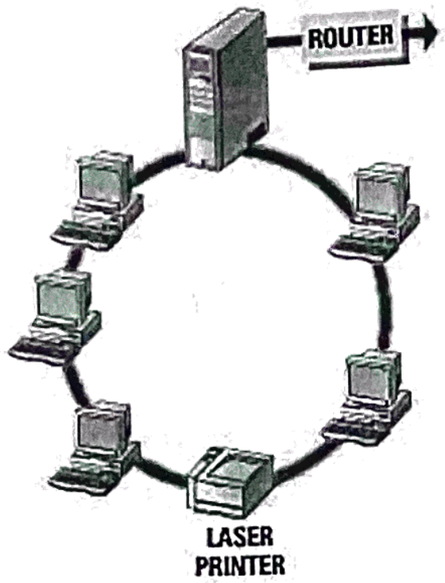
Advantages -

* Easy to install and expand
* Addition or deletion of a node is easier.
* Failure of one node will not affect the entire network.
* Well suited for quick setup
* Easier to debug network problems through a hub.

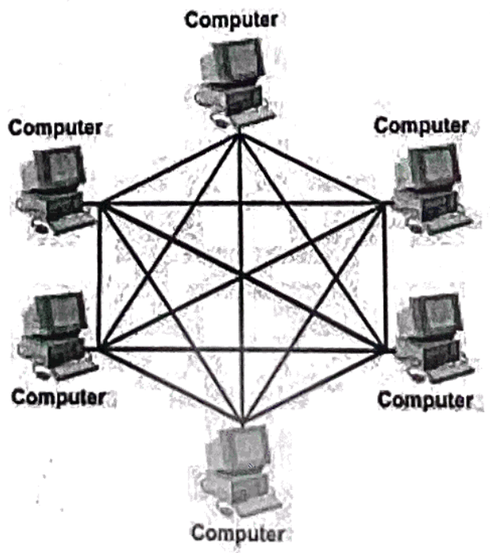
Disadvantages -

* Failure of all hub will affect the whole network
* Cost of hub is expensive

Ring -



Mesh -



DATE .....
EXP. NO. .....

EXPT. TITLE :

PAGE NO. 12

Ring Topology -

In this network all the computers are connected to each other in the form of a ring. It connects the nodes of the network in a circular chain in which each node is connected to the next.

Advantages -

* All the nodes have equal chance to transfer the data.
* These are easily extensible
* It can span longer distance than other types of networks.

Disadvantages -

* Difficult to install
* Difficult to troubleshoot
* Adding or remaing computer can disturb the entire network

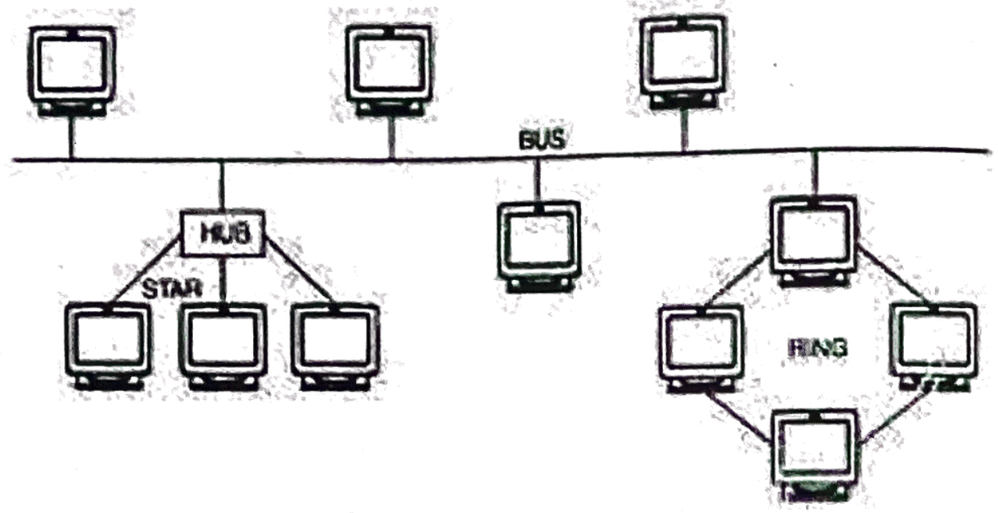
Mesh Topology -

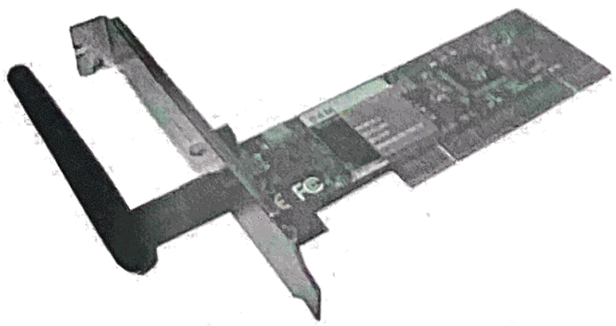
In this network all the computers and network devices are interconnected with one another like a mesh. Every node has connection to every other node in the network.

Advantages -

* Failure of a single node will not affect
* Data transfer rate is very fast because all the nodes are connected to each other

Hybrid -





NIC

DATE ..... EXP. NO. .....

EXPT. TITLE :

PAGE NO. 13

Disadvantages -

* Installation and re configuration is very difficult.

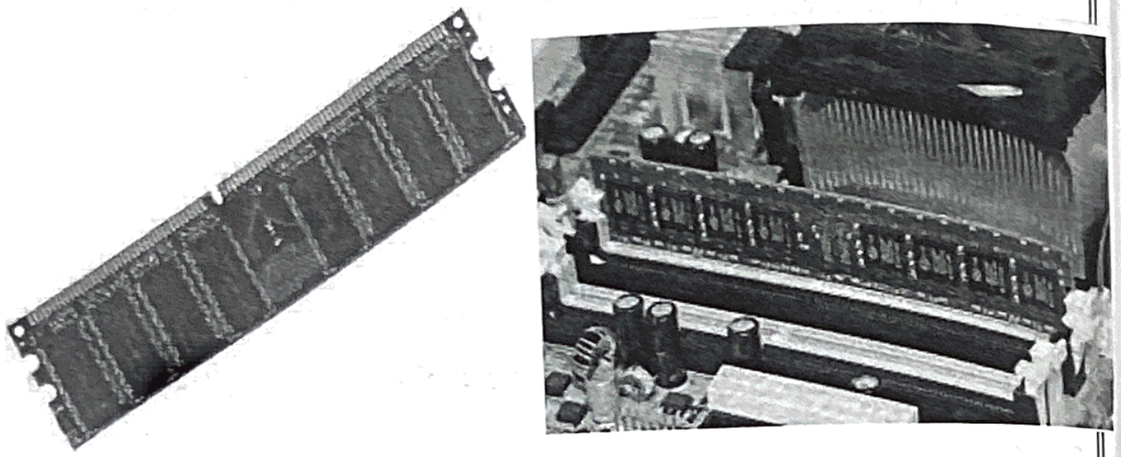
Hybrid Topology -

This topology uses a combination of any two or more topologies in such a that the resulting network does not exhibit one of the standard topologies. Ex - bus, star, ring etc.

Network Interface Cards (NIC) -

A network interface card is used to connect a machine to an computer network. This card provides an interface to the media. This may be either using an external transceiver mounted on the network interface card PCB. The card usually also contains the protocol control firmware and Ethernet Controller needed to support the Medium Access Control (MAC) data link protocol used by Ethernet.

Write-up on RAM, SDRAM, FLASH memory, Hard disks, Optical media, CD-ROM, R, RW, DVDs, Flash drives, Keyboard, Mouse, Printers and Plotters.



RAM fixed on Motherboard

DATE .....
EXP. NO. .....

EXPT. TITLE :

Random Access Memory (RAM) -

It is a volatile storage area within the computer that is typically used to store data temporarily. The information stored in the RAM is basically loaded from the computer's hard disk and includes data related to the operating system and applications that are currently being executed by the processor.

Types of RAM -

* **Static RAM** -

In RAM electrical signals are sent to individual memory locations on the RAM chip for future reading by the processor. If RAM is volatile that means that electricity must be supplied to the chip at all times in order to keep the information. Static RAM keeps this information in a series of transistors which remember the electrical signal given to them with just a standard power signal. Because of the complexity of this type of RAM it is usually more expensive, but there is less chance of information loss. The other great benefit of this type of RAM is in something called the L_2 cache.

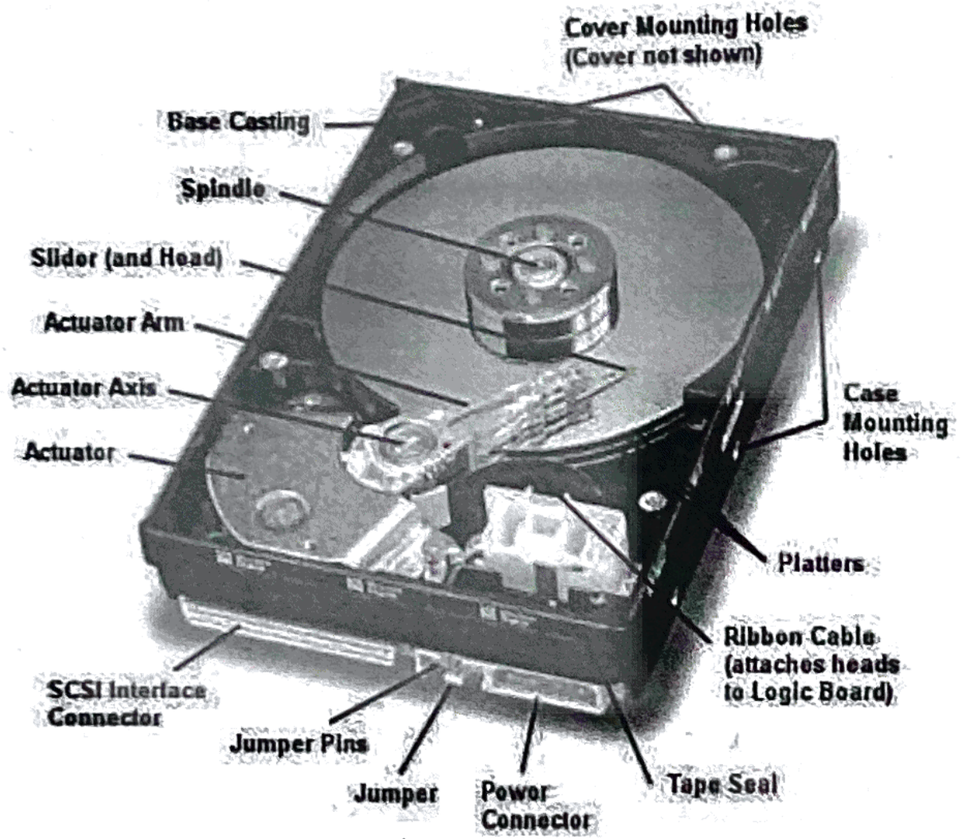
DATE
EXP NO
EXP TITLE
PAGE NO. 15

Dynamic RAM -

Dynamic RAM is different from static RAM in that it needs to be refreshed about every 15 nanoseconds. This slightly less in speed is made up by the reduced cost of this type of RAM. It is the most common form of RAM used in computers today. The most used type is DDR. DDR2 is succesor to DDR SDRAM. The current cutting edge counter computer RAM is DDR3. This manged to double even the speed of DDR2. Unfortunately none of the DDR ranks are compatible with each other and can't replace each other in a computer unless specifically designed for it.

SDRAM -

Synchronous Dynamic Random Acces Memory is an improvement standard DRAM because it retrieves data alternatively between two sets of memory. This eliminated the delay caused when one bank of memory address is shut down while another is prepared for reading. It is also called Synchronous DRAM because the memory synchronous with the clock speed that of computer's CPU has speeds optimized for.



Hard Disc

DATE .....
EXP. NO. .....

EXPT. TITLE :

PAGE NO. 16

Flash Memory -

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

Hard Disk -

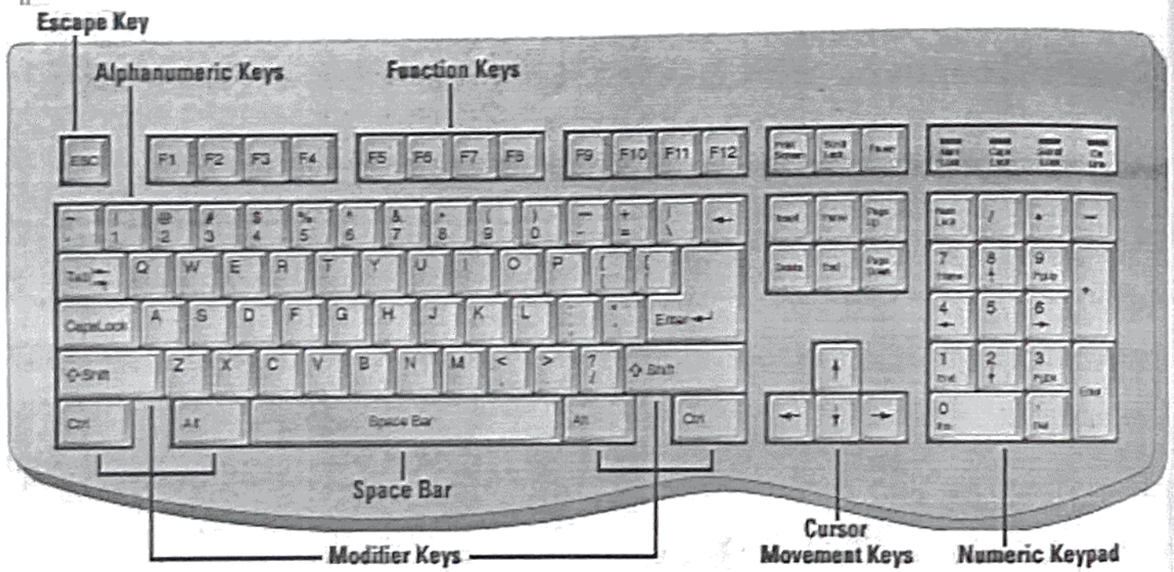
A hard disk is part of a unit called "disk drive" or hard drive or hard disk that stores and provids relatively quick access to large amounts of data on an electromagnetically charged surface or set of surfaces. Today's computers typically come with a hard disk that contains several billion types of storage.

Optical Media -

Optical media such as CD are storage media that hold content in digital form and that are written and read by a laser, these media include various CD and DVD variations, as well as optical jukeboxes and auto changers.

Ex - CD-ROM, DVD etc.

USB



Keyboard

DATE
EXP. NO.
EXPT. TITLE :
PAGE NO. 17

DVD -

Digital Versatile disc it is a type of optical disk technology similar to CD-ROM. A DVD holds a min of 4.7 GB of data, enough for a full-length movie. DVD are commonly used as a medium for digital representation of movies and other multimedia presentations that combine sound with graphics.
The DVD specifications support disks with capacities of from 4.7GB to 17GB and access rates of 600KBps to 1.3MBPS. One of the best features of DVD drives is that they are backward compatible with CD-ROM's meaning they can play old ones.

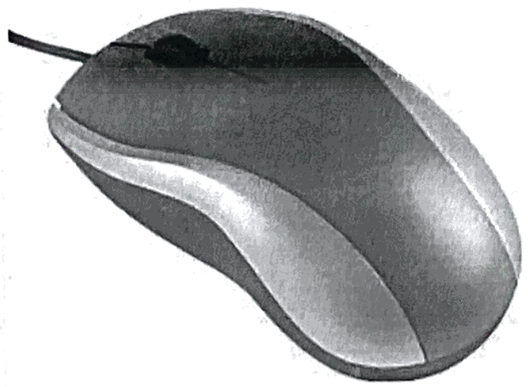
Flash drives -

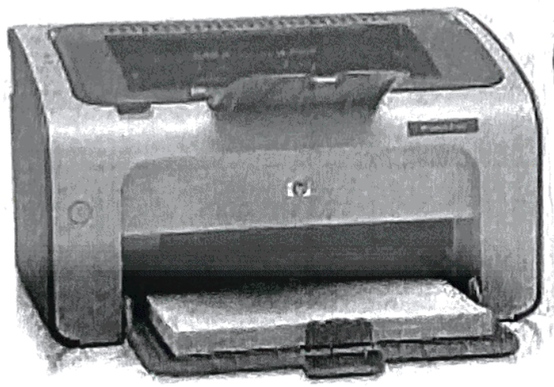
USB flash drives are removable, rewritable and physically much smaller drives weighing even less than 30g. A flash drive consists of a small printed circuit board anything carrying the circuit elements and a USB connector.

Keyboard -

A keyboard is a primary input device used in all computers. Keyboard has a group of switches resembling the keys on the ordinary typewriter machine. Normally

Mouse





Laser Printer

Inkjet Printer

keyboard has around 101 keys. The keyboard includes key that allows us to type letters, numbers and various special symbols.

Mouse -

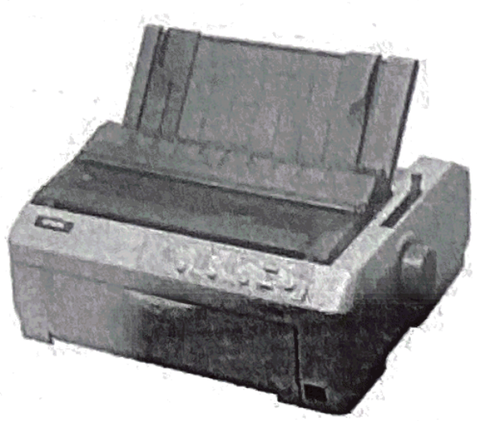
The mouse is the key input device to be used in a Graphical User Interface. The users can use the mouse to handle the cursor pointer easily on the screen to perform various functions like opening a program or file. When with mouse, the users no longer need to memorize commands, which was earlier a necessity when working with text-based command line environment such as MS-DOS.

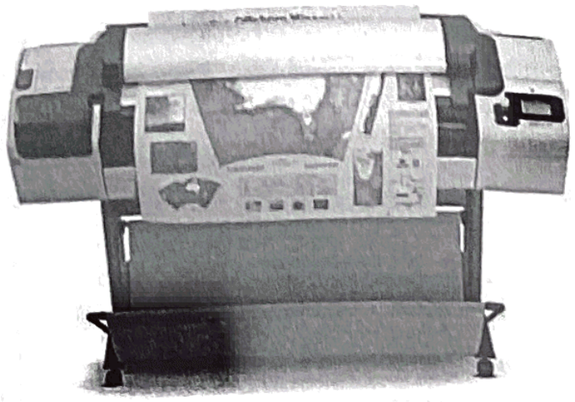
Printers :

The printer is an output device, which is used to get hard copy of the text displayed on the screen. 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) & Pages per minute (PPM) produced by the printer.

Types of printers -

1. Impact Printers: Impact printers are those printers in which a physical contact is established between the print head, ribbon & paper.





Dot Matrix Printer

DATE .....
EXP. NO. .....

EXPT. TITLE :

2. Non-Impact Printers: No physical contact is established b/w the print head, ribbon & paper. Ex: Inkjet printer & laser printer.

Plotters - A plotter is similar to printer that produces hard-copy o/p with high-quality color graphics. Plotters are generally more expensive than printers, ranging from about $100 to $5000

Problem Solving Techniques:

There are three approaches to solve a given problem:

- Algorithm.
- Flowchart.
- Pseudo code.

Algorithm: An algorithm is a step-by-step to be followed to solving a problem. It provides a scheme to solve a particular problem in finite number of unambigious steps.

Features of an algorithm:

Sequence: Each step of the algorithm is executed in specified order.

Decision: Decision statements are used to the outcome process

Repitition: Executing one or more steps for a number of times.

Example: To compute sum of two numbers.

Algorithm: Sum_Two-numbers.

Step1: [Initialize]
Start

Step2: [Input two numbers]
Read number1, number2

Step3: [compute sum of two numbers]
Sum ← number1 + number2

Step4: [Display the sum]
Print Sum

Step5: [Finished]
Stop.

➢ What is Flowchart?

A Flowchart is a pictorial representation of an algorithm.

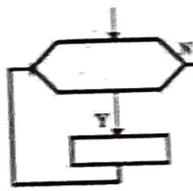
Flowchart Symbols: Meaning

Control Flow

Start/ Stop

Process

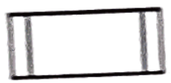
Input /Output



For / While loop

Flowchart Symbols: Meaning

Decision

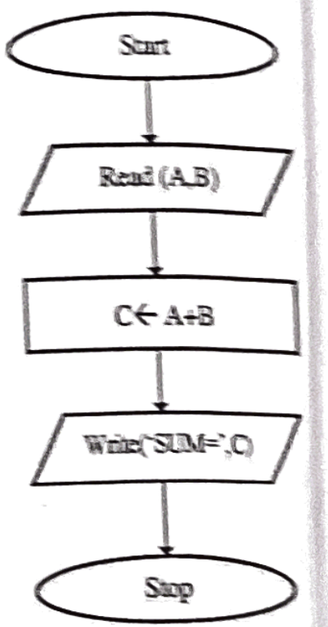


Subroutine

Off Page Connector

On Page Connector

Flowchart for the algorithm to find SUM of 2 Numbers



DATE	EXPT. TITLE:	PAGE NO. 20
EXP. NO.		

Flowcharts: A flowchart is a graphical or symbolic repr^n of an algorithm. They are basically used to design & develop simpler programs to help the users to visualize the logic of the program so that they can gain a better understanding of the program & 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.

-What is a flowchart?

A flowchart is a pictorial repr^n of an algorithm.

Pseudo code:

It is a form of structured English that describes algorithm. Pseudocode is a compact and informal high-level description that uses the structural conventions of a programming language.

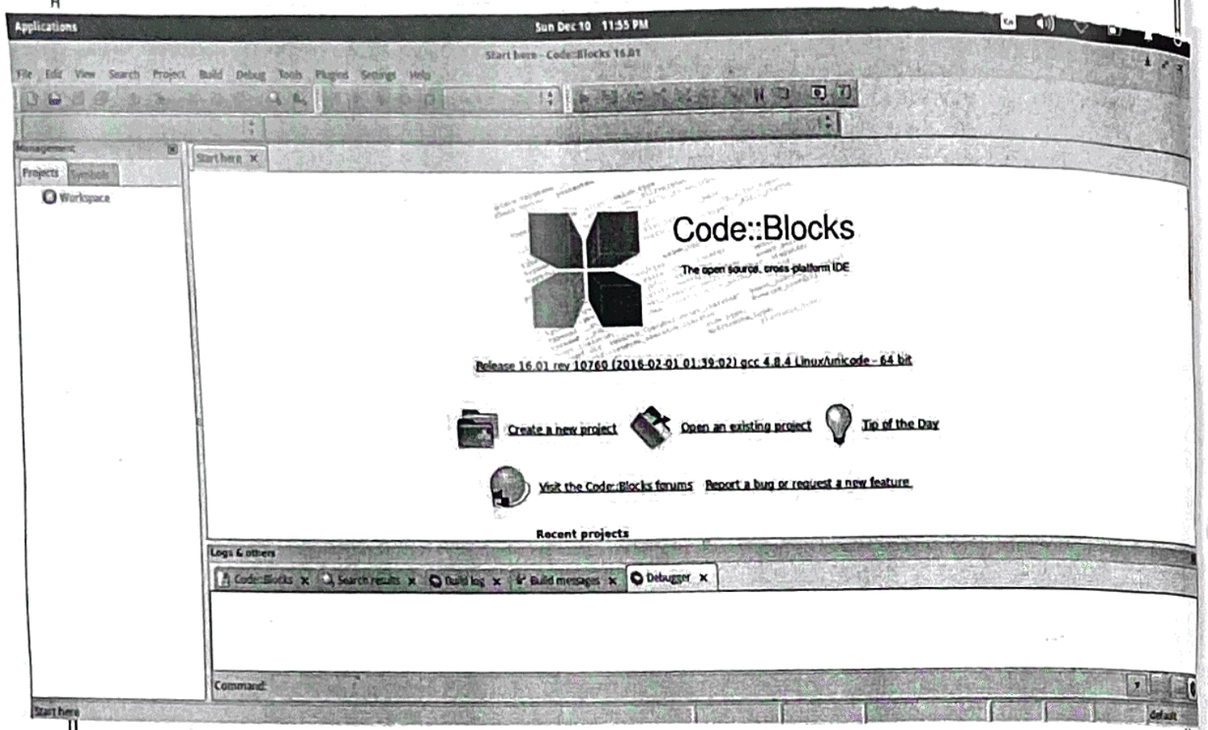
Eg: To compute sum of two numbers:

Input number 1, number 2

Sum = number 1 + number 2

Print Sum

Observation - 10
Record - 10
Viva - 10
Total 30



DATE 1/09/21	EXPT. TITLE :	PAGE NO. 21
EXP. NO. 2		

Laboratory Session-2

Part-A

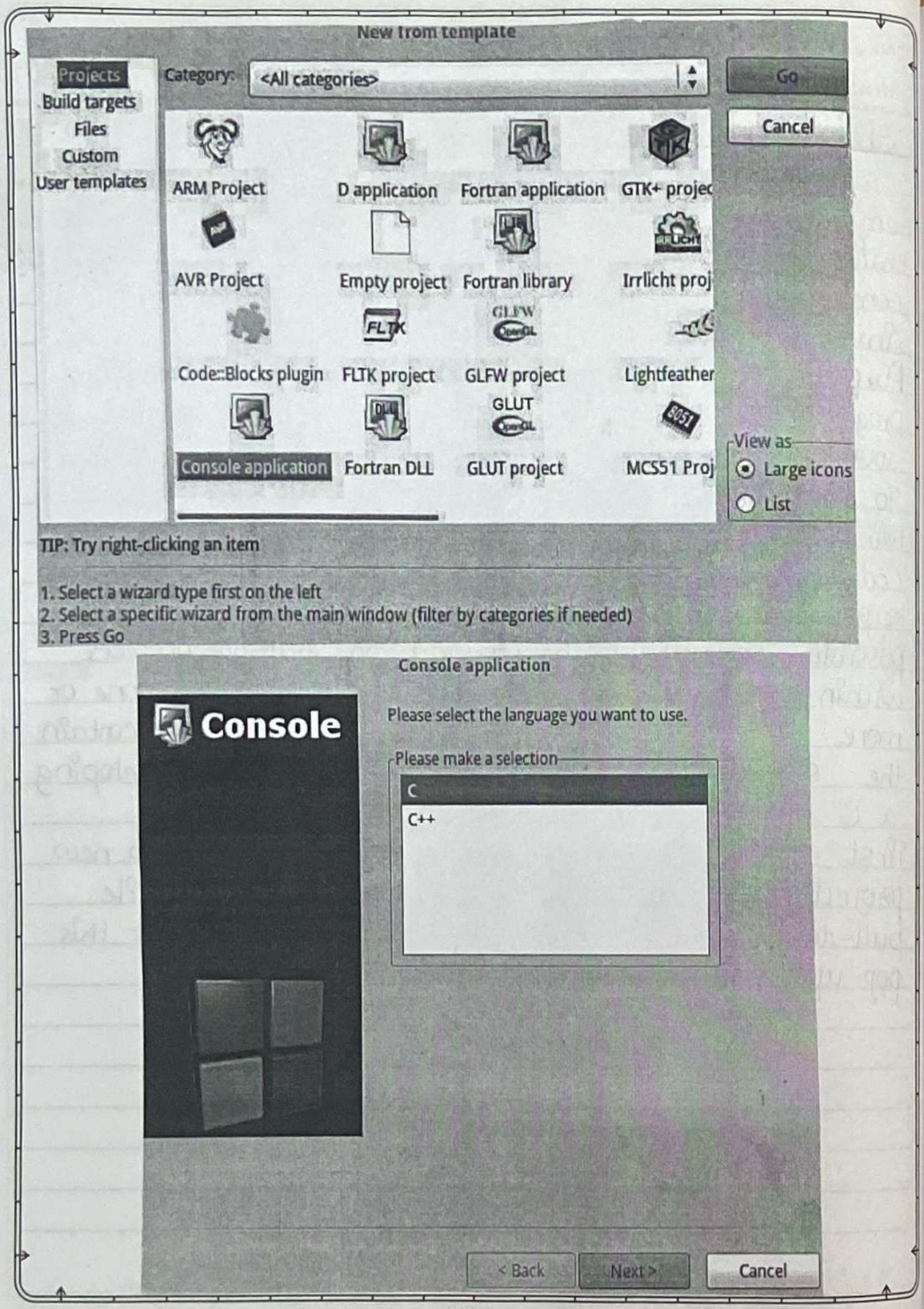
Familiarization with computer hardware and programing environment, concept of naming the program files, storing compilation, execution and debugging. Taking any simple C-code.

Purpose- This program demonstrates naming and saving program files, compilation, execution and debugging of source file.

To launch Code::Blocks IDE, Click on you will get a window as shown below:

Code::Blocks creates what is called a workspace to keep track of the project you are working on. It is possible for you to be working on multiple projects within your workspace. A project is a collection of one or more source files. Source files are the files that contain the source code for your program. If you are developing a C program, you are writing C source code.

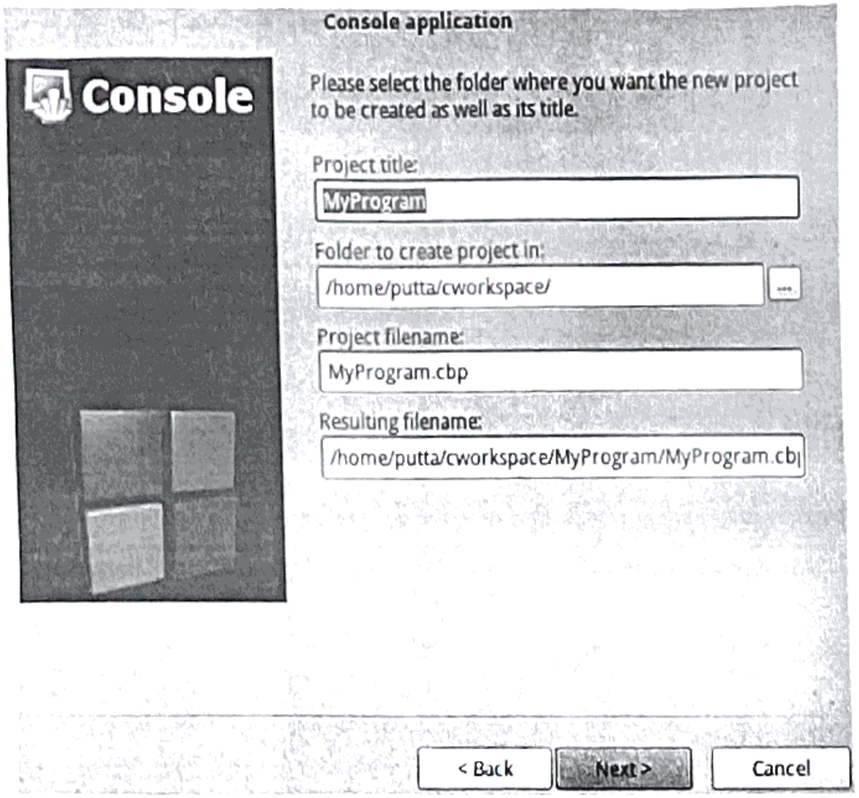
First start a new project by clicking on create a new project. OR to create a project, click on the File pull-down menu, open New and then Project. You get this pop-up window.



DATE
EXP. NO.
EXPT. TITLE :
PAGE NO. 22

Choose Console application
Then in the next window select the programming language C (and not C++)

In the next step give the project title and specify the folder where you want to save your project.



DATE .....
EXP NO. .....

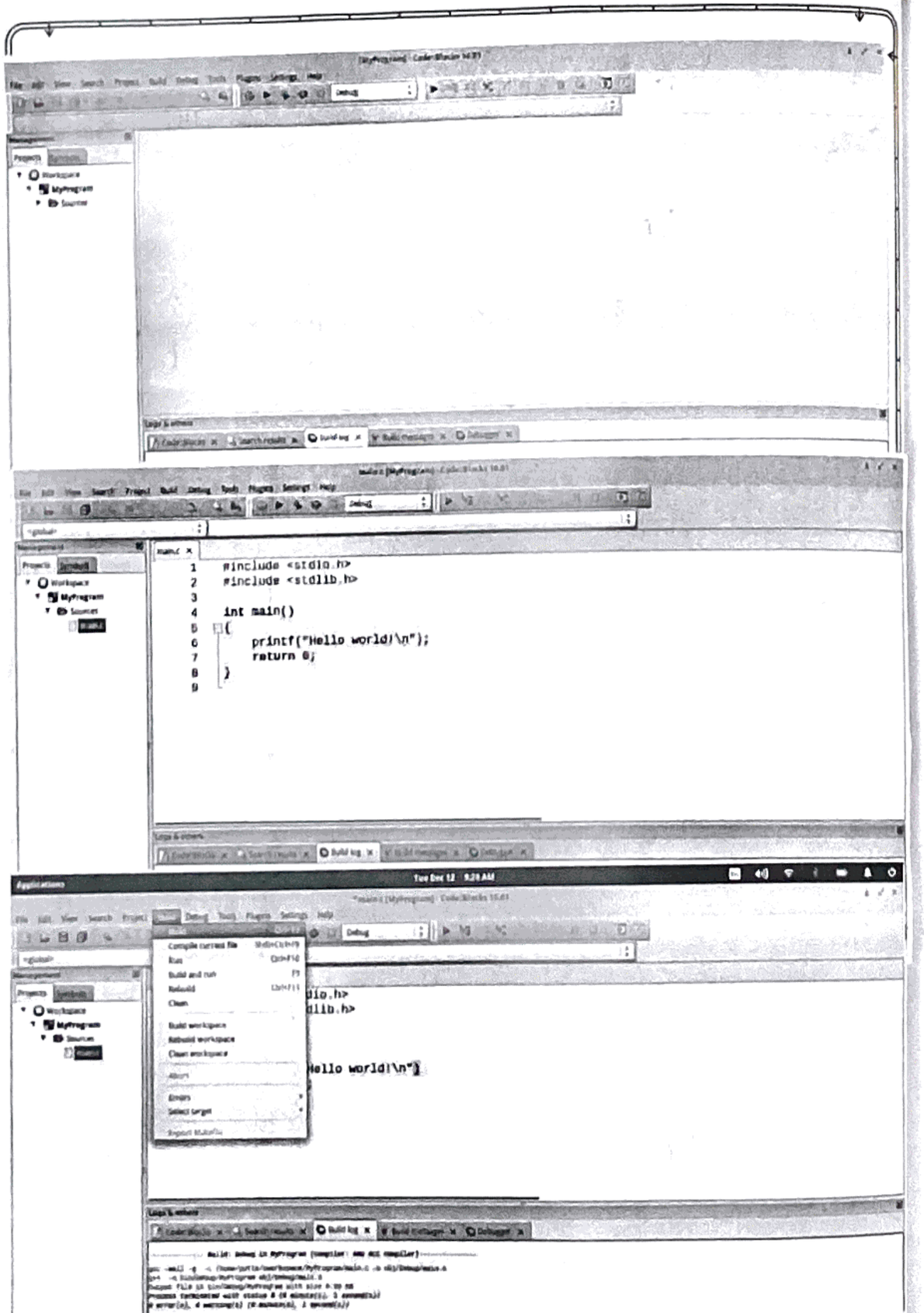
EXPT. TITLE :

PAGE NO 23

Finally you will be prompted to choose the compiler. Just choose the default options here. You should be using GNU GCC Compiler. Click finish to create the new project.

The system will then return to the [My Program] window and you are ready to write your program. In the management area of the screen (Shift-F2 toggles the management display), you will see the files that are part of the project in the projects tab. To see the source files, click on the triangle symbol to expand the Workspace and its subdirectories.

Under Sources, there is a file called main.c, which is automatically created for you when you build a console application. Click on main.c.



DATE .....
EXP NO .....

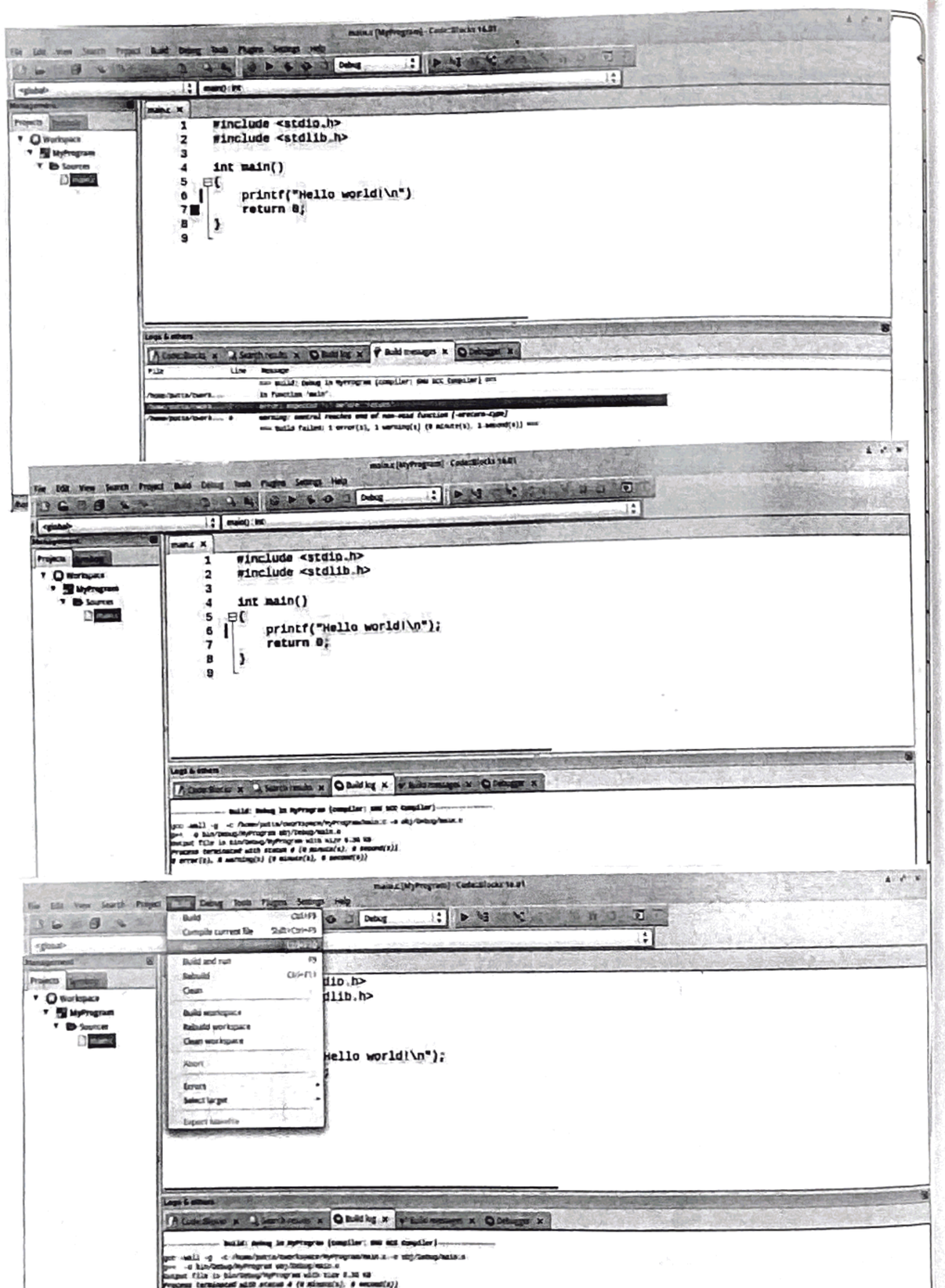
EXPT TITLE :

PAGE NO 24

main.c contains a simple Hello World program which you can edit later to solve a programming problem. Now let us see how to compile and execute this main.c program. Just to understand the process of debugging we knowingly introduce an error in the program. We will now compile the program.

The error messages are shown in the build messages window to the bottom. Let us now try to understand these error msgs.

Build: Debug in MyProgram (compiler: GNU GCC Compiler)
main.c In function main: main.c 7error: expected; before return
main.c 8 warning: control reaches end of non-void function [-Wreturn-type]
=Build failed: 1 error(s), 1 warning(s) (0 minute(s), 1 second(s))===



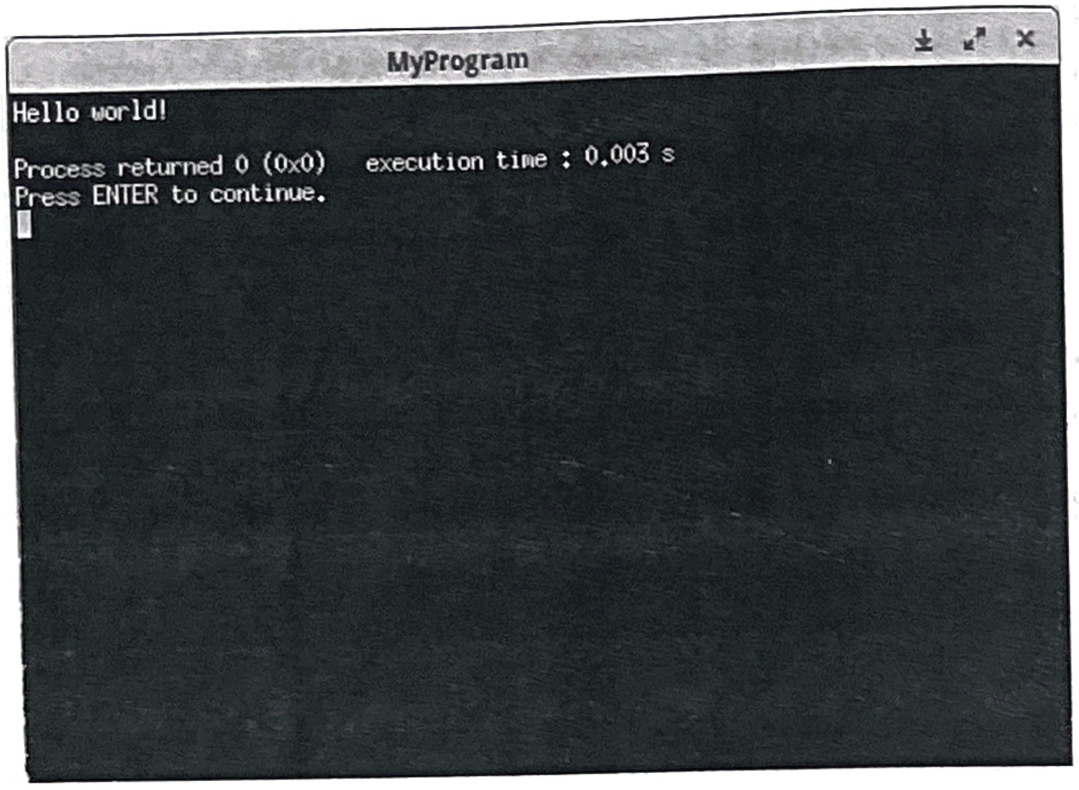
DATE .....
EXP. NO. .....

EXPT. TITLE :

PAGE NO. 25

The error msgs show the errors in syntax. It also indicates the line number & the type of error. Here in this example the error says that before return statement in line no 7, a semicolon is missing. The next msgs is a warning message which has resulted because of the previous error. Now go to line no. 6 and add a ; at the end. Now build your project again.

Now build msg window shows following msg. 0 error(s), 0 warnings, 0 mins, 0 seconds. This means that errors & wrngs have been successfully resolved. Now it is time to run the progrm. You can execute project from build pull-down menu by clicking on Run option [Ctrl+F10]



DATE

EXP. NO.

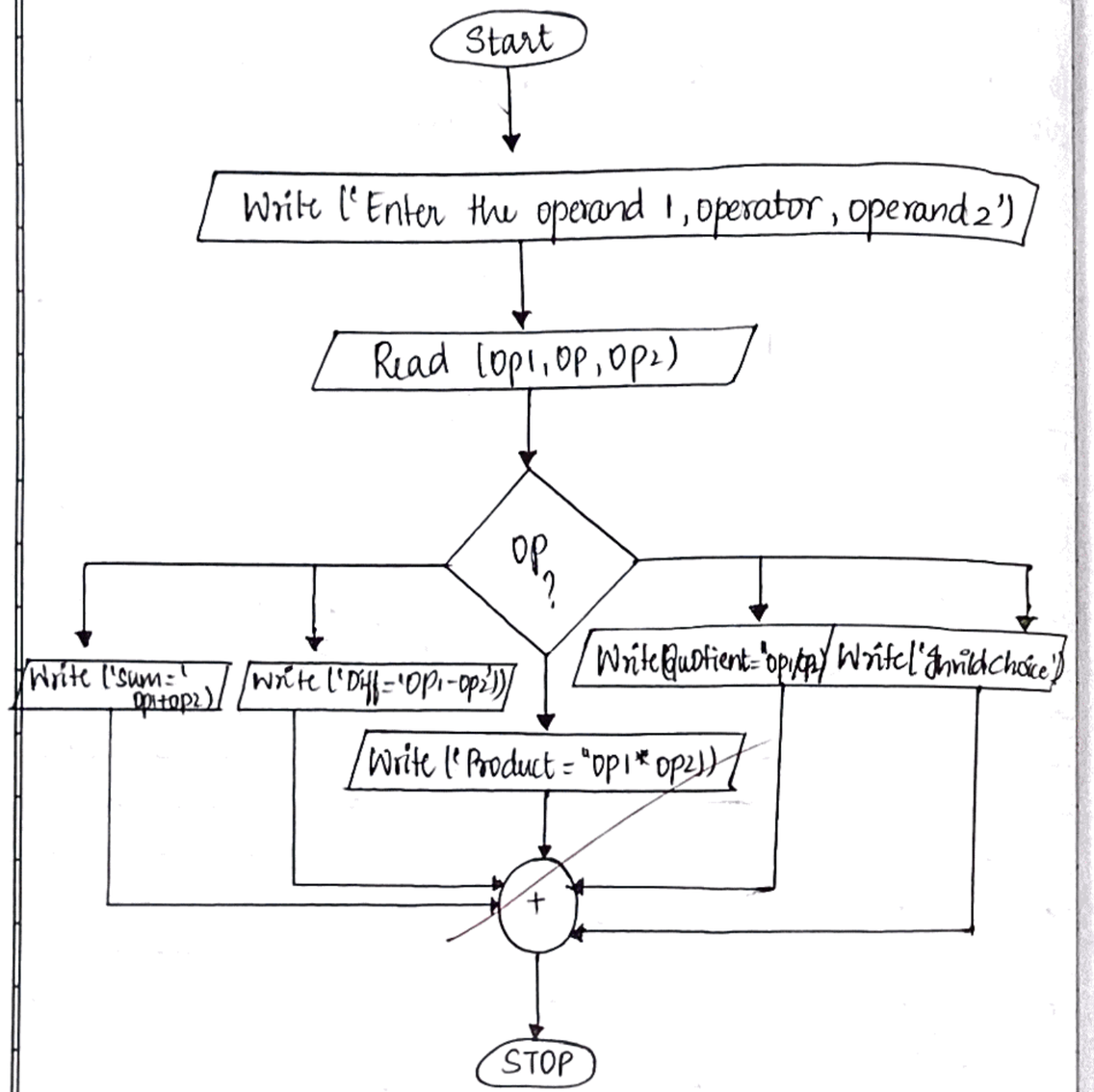
EXPT. TITLE :

PAGE NO. 26

An o/p window pops displaying the op of the prgm. A greeting msg Hello World! is printed to the o/p console.

Observation -10
Record -10
Viva -10
Total -30

Flowchart -



DATE 1/09/21 | EXPT. TITLE : | PAGE NO 27

EXP. NO 3

Develop a program to solve simple computational problems using asithmetic expressions and use of each operator leading to simulation of a commercial calculator.

Algorithm

Step-1 - [Initilize] Start

Step-2 - [Input the values of two operands and operator] read num1, op, num2

Step-3 - [Check the operator and redd] Switch(ch)
Case '+' then Print (sum: num1 + num2)
Case '-' then Print (sub: num1 - num2)
Case '*' then Print (Mult: num1 * num2)
Case '/' then Print (Quo: num1 / num2)
Otherwise Print "Invalid Choice"

Step-4 - [Finished] Stop.

/* Program to design simple commercial calculator */

```
#include <stdio.h>
void main()
{
int op1, op2;
char op;
printf ("Enter Arithmetic operation \n");
scanf ("%d %c %d", &op1, &op, &op2);
Switch (op)
```

Output -

```
Enter arithmetic expresion 1+2
Sum = 3
Enter arithmetic expresion 1-2
Difference = -1
Enter arithmetic expresion 1*2
Product = 2
Enter arithmetic expresion 1%2
Remainder = 1
Enter arithmetic expresion 1%2
Invalid Operator.
```

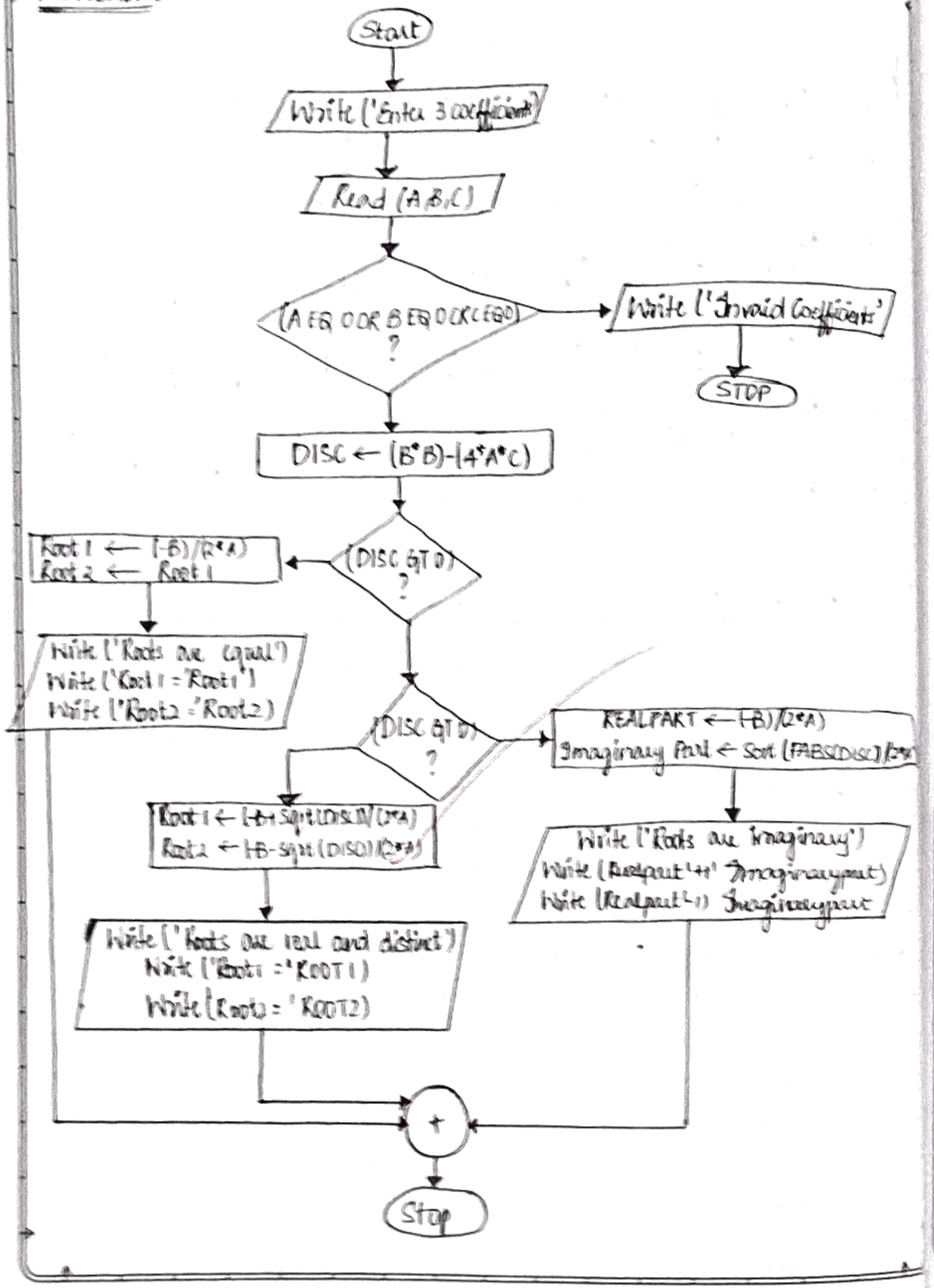


```
case '+': printf ("sum = %d\n", (op1+op2));
break;
case '-': printf ("difference = %d\n", (op1-op2));
break;
case '*': printf ("product = %d\n", (op1*op2));
break;
case '/': printf ("quotient = %d\n", (op1/op2));
break;
case '%': printf ("remainder = %d\n", (op1%op2));
break;

default :
  printf ("Invalid Operator\n");
}
return 0;
}
```

Observation - 10
Record - 10
Viva - 10
Total - 30

Flowchart -



DATE 1/09/21	EXPT. TITLE :	PAGE NO 29
EXP NO 4		

Develop a program to compute the roots of a quadratic equation by accepting the coefficients. Print appropriate messages.

Algorithm

Step 1 : (Start the algorithm)
Start

Step 2 : (Read the coefficients)
Read non zero coefficients a, b, c

Step 3 : (Calculate the discriminant)
$d \leftarrow b*b - 4*a*c$

Step-4 : (Check if roots are real and equal)
if (d = 0)
$x1 \leftarrow -b/(2*a)$
$x2 \leftarrow -b/(2*a)$
Print "Roots are equal"
Print $x1, x2$
Go to Step 7

Step-5 : (Check if roots are real and distinct)
if (d > 0)
$x1 \leftarrow (-b + sqrt(d))/(2*a))$
$x2 \leftarrow (-b - sqrt(d)/2*a))$
Print "Roots are real and distinct"
Print $x1, x2$
Goto Step 7

Step-6 : (Check if roots are imaginary) If d < 0)
$x1 \leftarrow (-b/(2*a))$
$x2 \leftarrow$ sqrt (fabs(d))/(2*a))

Print "The roots are complex"
Print "Root 1 ←", $x_1 + ix_2$"
Print "Root 2 ←", $x_1 - ix_2$
Step-7 : [Terminate the algorithm]
Stop.

/* Program to find roots of quadratic equation */

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
void main()
{
float a,b,c,d,x1,x2;
printf("Enter non zero coefficients a,b and c of
 a quadratic equation\n");
scanf("%f%f%f",&a,&b,&c);
if (a==0||b==0||c==0)
{
 printf("Invalid coefficients");
 exit(0);
}
d=b*b-4*a*c;
if (d==0)
{
 printf("Roots are equal\n");
 x1 = -b/(2*a);
 x2 = -b/(2*a);
```

Output -

Sample -1

Enter the three coefficients: 1 4 4
The roots are real and equal
root 1 = -2.0000 root 2 = -2.0000

Sample -2

Enter the three coefficients: 1 5 6
The roots are real and distinct
root 1 = -2.0000 root 2 = -3.0000

Sample -3

Enter the three coefficients : 2 3 4
The roots are imaginary
root 1 = -0.75+i1.19 root 2 = -0.75-i 1.19

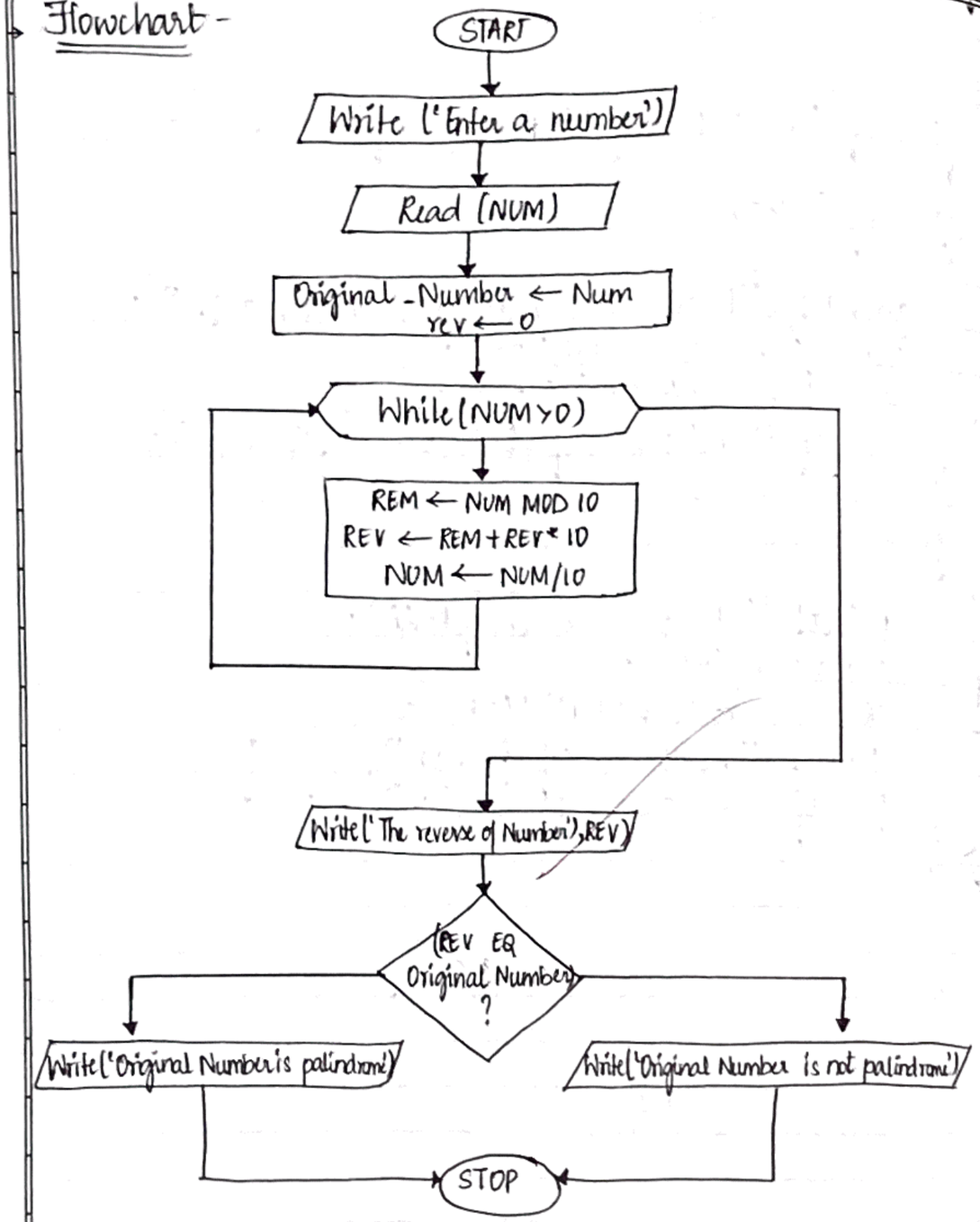
Sample -4 -

Enter the three coefficients : 2 3 0
Invalid coefficients.


```
    printf ("Root 1 = %f \t Root 2 = %f", x1, x2);
}
else if (d>0)
{
    printf ("Roots are real and distinct \n");
    x1 = (-b+ sqrt (d)) / (2*a);
    x2 = (-b- sqrt (d)) / (2*a);
    printf (" Root 1 = %f \t Root 2 = %f", x1, x2);
}
else
{
    printf ("Roots are complex\n");
    x1 = -b/(2*a);
    x2 = sqrt (fabs(d)) / (2*a);
    printf (" Root 1 = %f + i %f \n", x1, x2);
    printf (" Root 2 = %f - i %f \n", x1, x2);
}
}
```

Observation	- 10
Record	- 10
Viva	- 10
Total	- 30

Flowchart -



DATE 6/09/21
EXP. NO. 5
EXPT. TITLE :
PAGE NO. 32

Develop a program to find the reverse of a positive integer and check for palindrome or not.

Algorithm

Step-1 : Start
Step-2 : Read n
Step-3 : temp ← n
rev ← 0
Step-4 : Repeat through step 4
while (n != 0)
rem ← n % 10
rev ← (rev * 10) + dig
n ← n/10
Step-5 : print rev
Step-6 : if (rev == temp)
print ("No is a palindrome")
else
print ("No is not a palindrome"
Step-7 : Stop

Output -

Sample - 1

Enter a number : 5642
5642 is not palindrome

Sample - 2

Enter a number : 2002
2002 is not palindrome

DATE
EXP NO
EXPT TITLE :
PAGE NO 33

```
/* Program to check whether the given number is
   palindrome or not */

#include <stdio.h>
void main()
{
   int n, temp, rev, rem;
   printf("Enter an integer\n");
   scanf("%d", &n);
   temp = n;
   rev = 0;

   while (n!=0)
   {
      rem = n % 10
      rev = rev * 10 + rem;
      n = n/10;
   }
   printf("The reversed number is %d\n", rev);
if(rev == temp)
printf("%d is a palindrome", temp);
else
printf("%d is not a palindrome", temp);
}
```

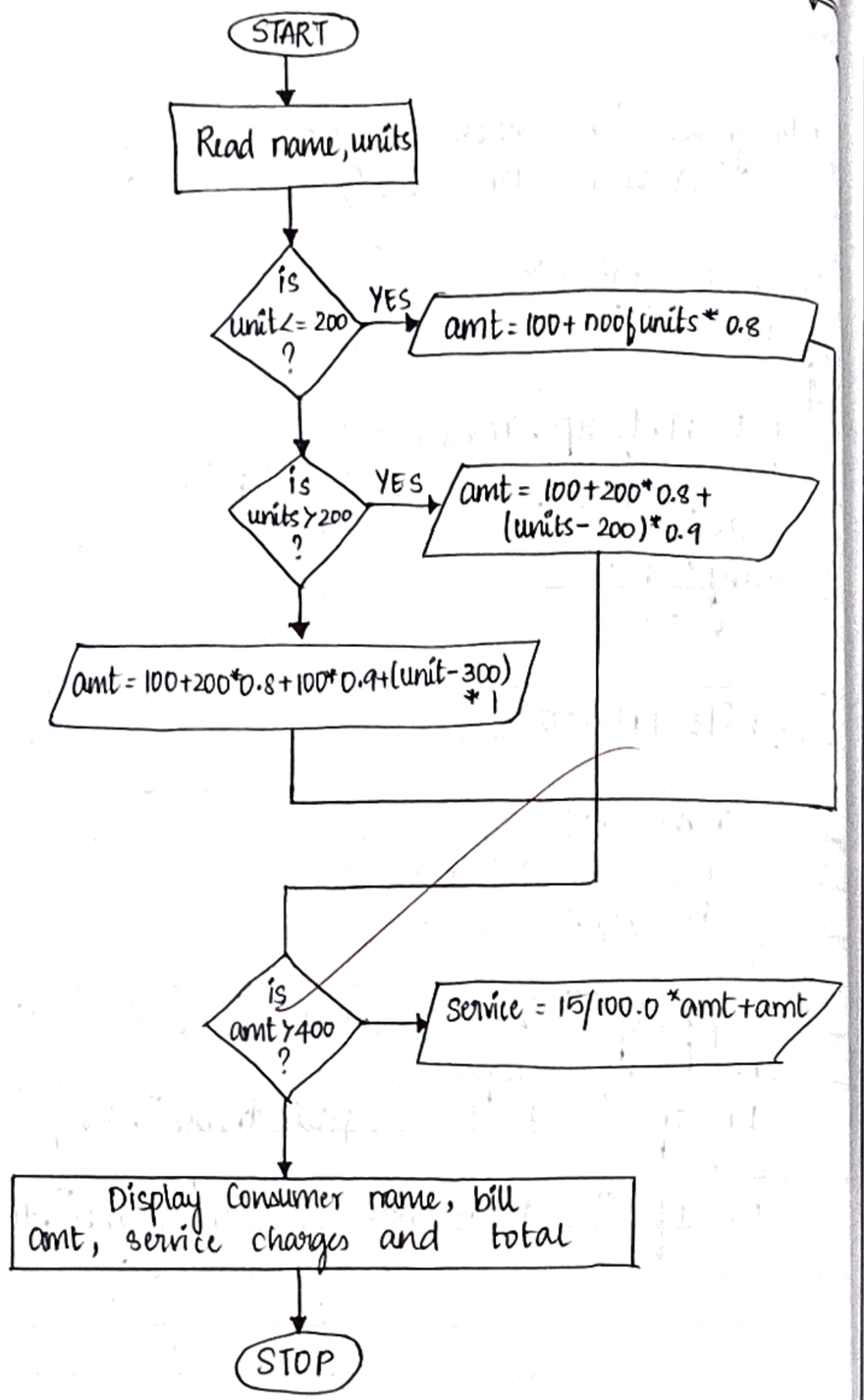
Observation - 10
Record - 10
Viva - 10
Total 30

DATE 6/09/21 | EXPT TITLE : | PAGE NO 34

EXP NO. 6

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 user, number of units consumed and print out the charges.

Flowchart -



Algorithm

Step-1 : Start
Step-2 : Read name
Step-3 : Read units
Step-4 : If (units <= 200)
amt = 100 + no of units * 0.8
else if
(units > 200) and (units <= 300)
amt = 100 + 200 * 0.8 + (units - 200) * 0.9
else
amt = 100 + 200 * 0.8 + 100 * 0.9 + units - 300 * 1
Step-5 : if
(amt > 400) service = 15/100.0 * amt + amt
Step-6 : print "Customer name, bill amount, service charges and total", name, amt

sur, total

Step -7 : Stop

```
/* Program to compute electricity bill */

# include <stdio.h>
void main()
{
   char name[10];
   int unit, m = 100;
   float charge;
   printf("Enter your name and unit consumed:");
   scanf("%s %d", name, &unit);
   if (unit <= 200)
   charge = unit * 0.80+m;

   else if (unit > 200 && unit <= 300)
   {
     charge = (unit -200)* 0.90+200* 0.80+m;
   }
   else
   {
     Charge =(unit-300)*1 + 200* 0.80+100*0.90+m;
   }
    if (charge >= 400)
   {
   charge= charge + charge*0.15;
   }
```

Output -

Sample - 1

```
Enter the customer name : Deejay
Enter number of units consumed : 200
The customer name = Deejay
The number of units consumed = 200
Elecity Bill Amount = 260.00
Service Charge = 0.00
The total Electricity Bill amount : 260.00
```

Sample - 2

```
Enter the customer name : Vinod
Enter the number of units consume : 0
The customer name = Vinod
The number of units consumed = 0
Electricity bill amount = 100.00
Service Charge = 0.00
The total electricity bill amount = 100.00
```

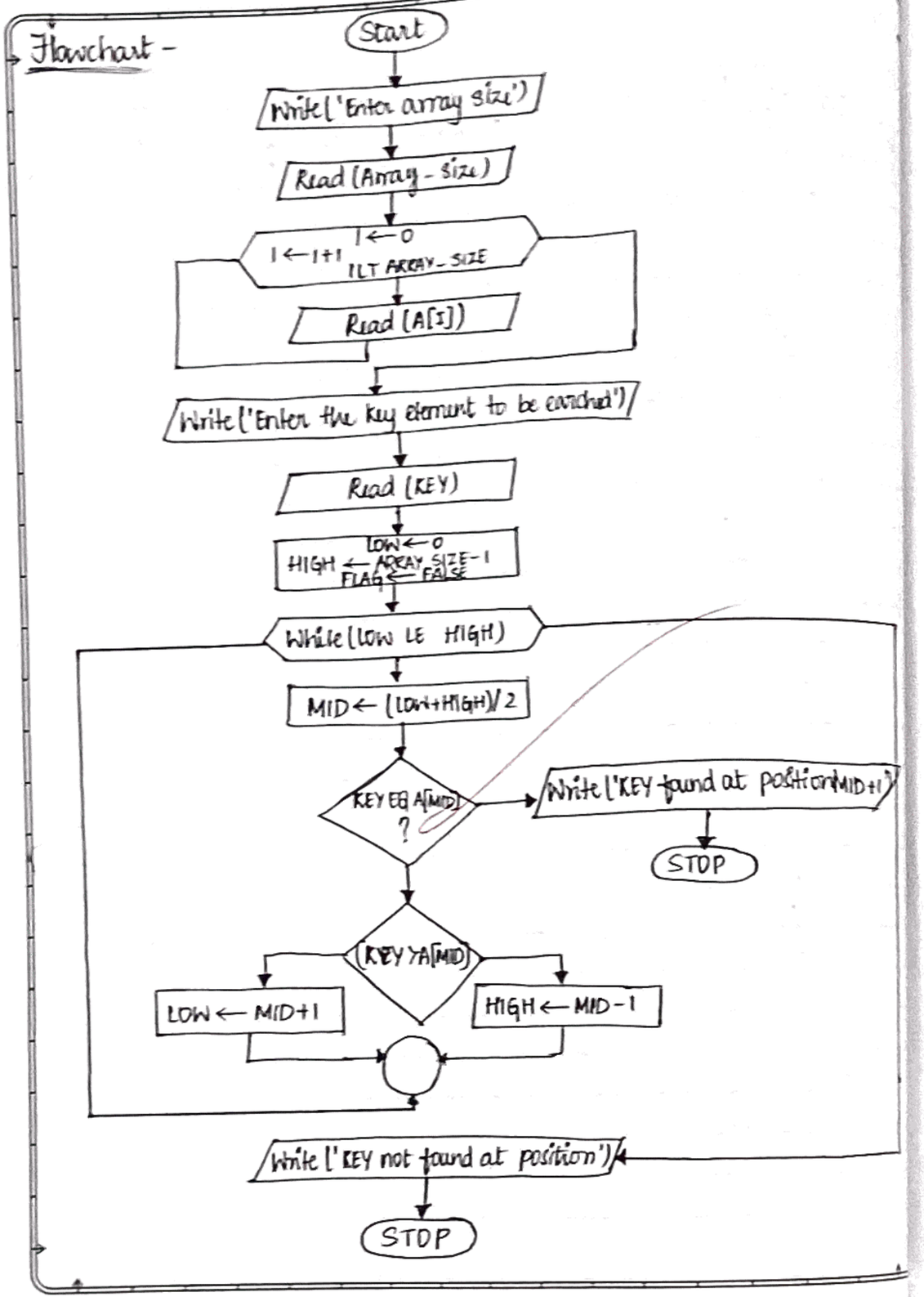


```
printf("Name: %s\n Charge : %f", name, charge);
}
```

Observation - 10
Record - 10
Viva - 10

Total - 30

Flowchart -



DATE 6/09/21
EXP. NO. 7
EXPT. TITLE :
PAGE NO. 37

Introduce 1D array manipulation and implement Binary Search.

Algorithm

Step 1 : Start
Step 2 : Read n
Step 3 : for i=0 to n-1 read A
Step 4 : Read Key
Step 5 : Assume low=0, last = n-1 and mid = (low+high)/2
Step -6 : while (low<=high)
begin while
if (mid == key)
print "element found"
else if (mid > key)
high = mid-1
else
low = mid + 1
end while
Step -7 : print "Element not found"
Step - 8 : Stop.

DATE
EXP. NO.
EXPT. TITLE :
PAGE NO. 38

```
/* Program to search an element in a list
 of elements using binary search*/

#include <stdio.h>
#include <stdlib.h>
void main()
{
   int a[30], i, n, key, low, mid, high;
   printf("\n Enter the no.of elements");
   scanf("%d",&n);
   printf("\n Enter the elements: ");
   for(i=0; i<n; i++)
   {
      scanf("%d",&a[i]);
   }
   printf("\n Enter the key element to be searched\n");
   scanf("%d",&key);
   low=0;
   high=n-1;
   while(low<=high)
   {
      mid=(low+high)/2
      if(a[mid]==key)
      {
         printf("Element %d is found at %d position:",
                                          key,mid+1);
         exit(0);
      }
```

Output -

Sample - 1

```
Enter the number of elements in an array: 5
Enter the 5 elements in asanding order
4
7
8
9
45
Enter value to find: 9
9 found at location 4.
```

Sample - 2

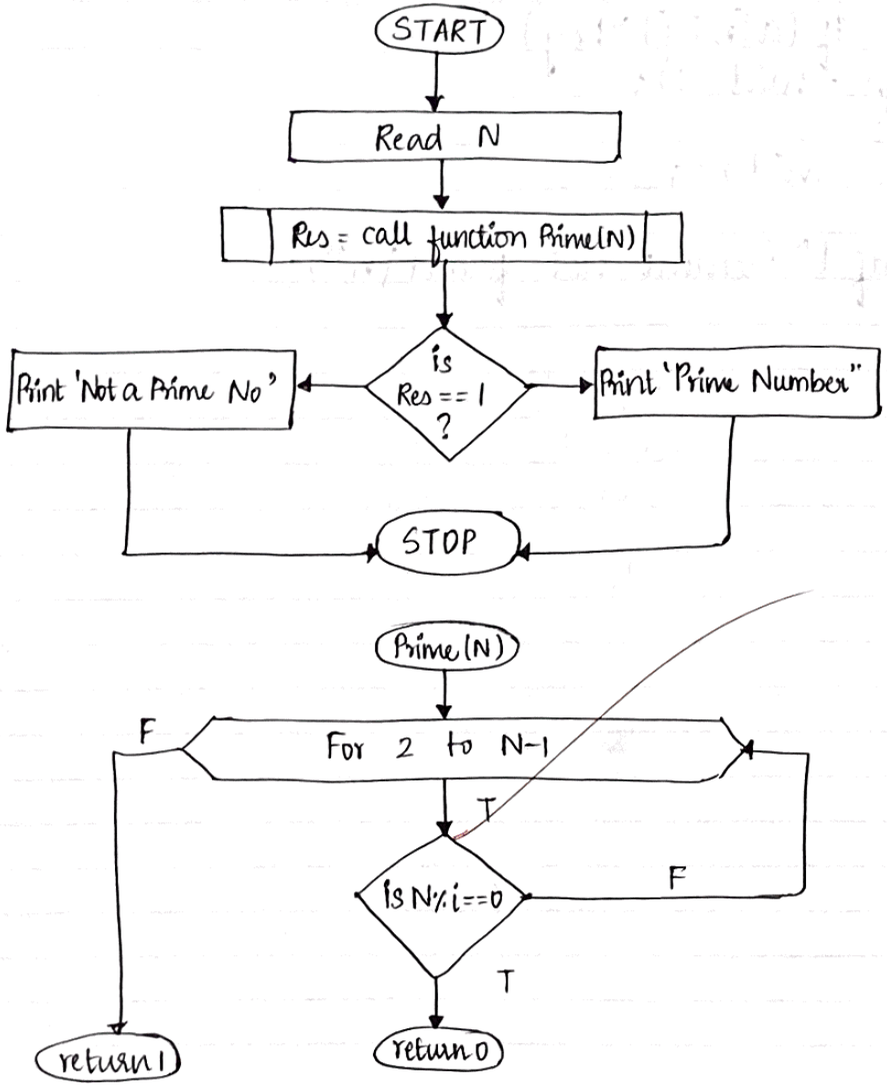
```
Enter the number of elements in an array: 7
Enter the 7 integers in ascending order
9
23
41
90
132
290
301
Enter the value to find: 10
Element not found.
```



```
else if (a[mid] > key)
high = mid - 1);
else
low = mid + 1;
}
printf("Element not found\n");
}
```

Observation - 10
Record - 10
Viva - 10
Total - 30

Flowchart -



DATE 6/09/21 EXP. NO. 8 EXPT. TITLE : PAGE NO. 40

Implement using functions to check whether the given number is prime and display appropriate messages.

Purpose - This program demonstrates USER DEFINED FUNCTIONS

Procedure - Input N array elements and to find whether element is present or not

Input - Array elements

Expected Output - Succesful search or unsuccesful search

Algorithm

Step 1 : Start
Step 2 : Read num
Step 3 : res = prime(num)
Step 4 : if (res=1)
print ("The enterned num is a prime")
else
print ("The entered number is not prime")
Step 5 : Stop

Algorithm : prime (num)

Step-1 : Start
Step-2 : Repeat i ← 2 to n-1

Step 2.1 : If (n mod i == 0) then
return 0
end if
return 1
Step 3 : Stop

```
/* Program to find prime number */

#include <stdio.h>
int prime(int n);
void main()
{
    int num, res = 0;
    printf("\n Enter a number: ");
    scanf("%d", &num);
    res = prime(num);
    if (res == 1)
    printf("\n%d is a prime number", num);
    else
     printf("\n%d is not a prime number", num);
}
int prime(int n)
{
    int i;
    for(i=2; i<=n; i++)
    {
        if(n%i==0)
        return 0;
```

Output -

Sample-1

Enter any positive integer : 7
The given number 7 is prime

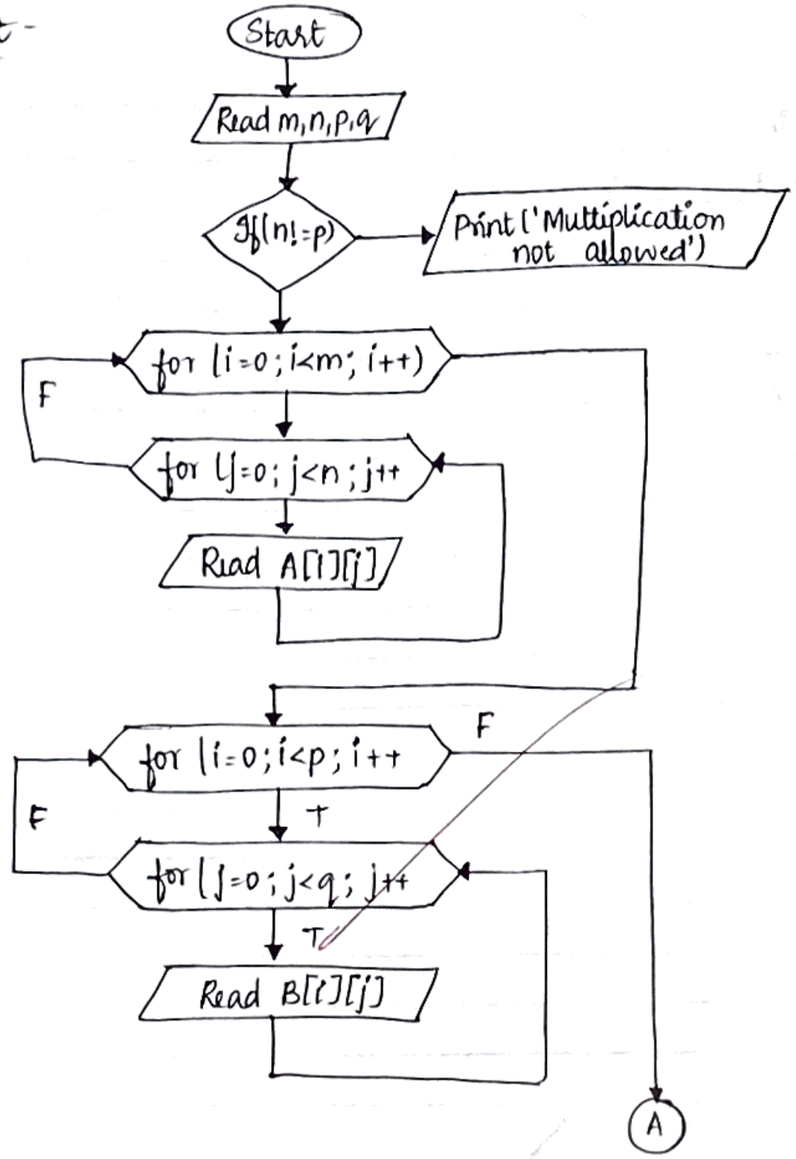
Sample-2

Enter any positive integer: 100
The given number 100 is not prime.


```
}
return 1;
}
```

Observation - 10
Record - 10
Viva - 10
Total - 30

Flowchart -



DATE 6/09/21 | EXPT. TITLE | PAGE NO 43

EXP. NO. 9

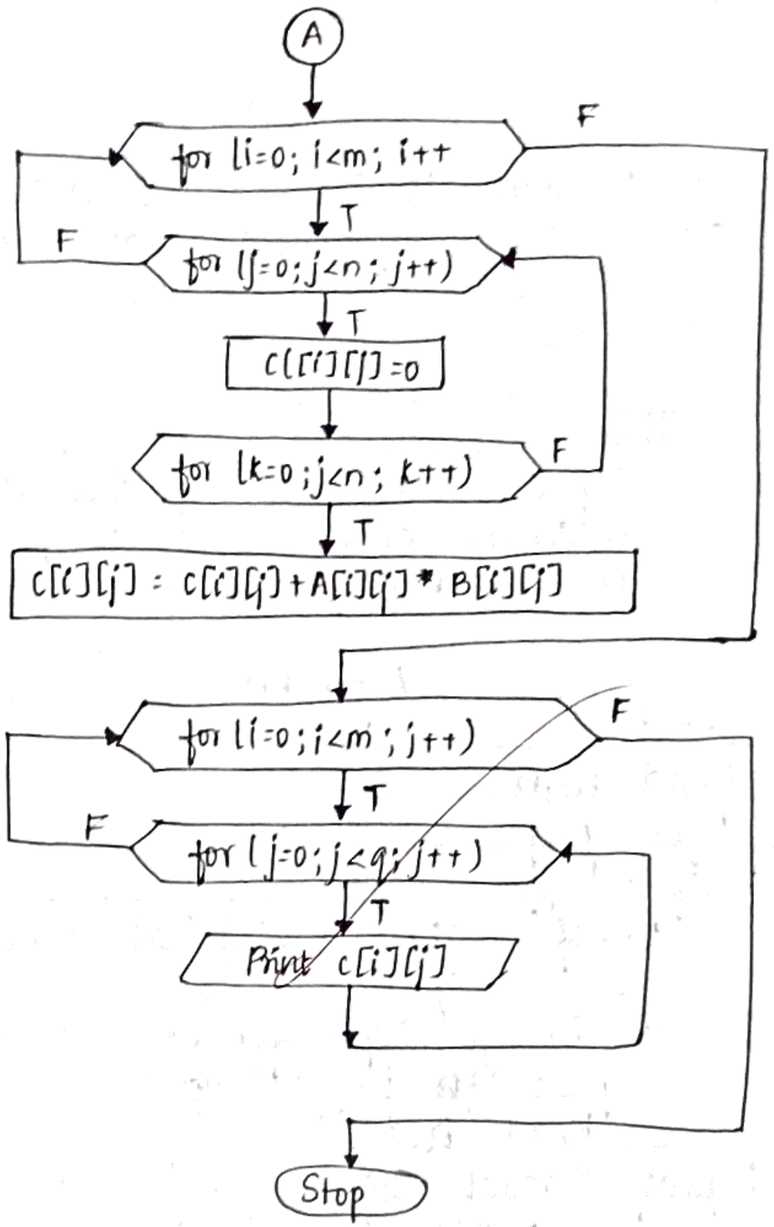
Develop a program to introduce 2D array manipulation and implement Matrix multiplicati -on and ensure the rules of multiplication are checked.

Purpose - This program demonstrates 2D array

Procedure - Input m*n and p*q size of 2 matrices elements to compute matrix multiplication.

Algorithm

Step-1: Start
Step-2: Read m,n
Step-3: Read p,q
Step-4: if (n!=p)
printf " Matrix multiplication not possible"
go to step 11
Step-5: Repeat through step 5 for i ← 0 to m-1
Repeat for i ← 0 to n-1
Read a[i][j]
Step-6: Repeat through step 6 for j ← 0 to q-1
Repeat for j ← 0 to q-1
Read b[i][j]
Step-7: Repeat through step 9 for i ← 0 to m-1
Repeat for j ← 0 to q-1
c[i][j] ← 0
Repeat for k ← 0 to n-1



c[i][j] ← c[i][j] + a[i][k] * b[k][j]

Step-8 : Repeat step 10 for i ← 0 to m-1
Repeat for j ← 0 to n-1
Print c[i][j]

Step-9 : Stop.

/* Program to multiply two matrices */

```
#include <stdio.h>
#include <stdlib.h>
void main()
{
    int i,j,m,n,p,q,k,a[5][5],b[5][5],c[5][5];
printf("Enter the order of matrix A\n");
scanf("%d%d",&m,&n);
printf("Enter the order of matrix B\n");
scanf("%d%d",&p,&q);

int(n!=p)
{
   printf("Multiplication is not possible\n");
   exit(0);
}
printf("Enter the elements of matrix A:\n");
for(i=0; i<m; i++)
{
   for(j=0; j<n; j++)
   {
```

```
        scanf("%d",&a[i][j]);
    }
}
printf("Enter the elements of matrix B\n");
for(i=0;i<p;i++)
{
    for(j=0;j<q;j++)
    {
        scanf("%d",&b[i][j]);
    }
}
for(i=0;i<p;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][i]
    }
}
printf("Product matrix is:\n");
for(i=0;i<m;i++)
{
    for(j=0;j<q;j++)
    {
        printf("%d\t",c[i][j]);
    }
    printf("\n")
```

Output -

Sample - 1

```
Enter the size of matrix A : 2 2
Enter the size of matrix B : 2 2
Enter the elements of matrix A :
1  2
3  4
Enter the elements of matrix B:
2  3
4  5
Matrix A is
1  2
3  4
Matrix B is
2  3
4  5
The product of 2 matrices is
10  13
22  29
```

Sample - 2

```
Enter order of matrix A : 2 3
Enter order of matrix B : 2 3
Multiplication is not possible
```

DATE

EXP. NO.

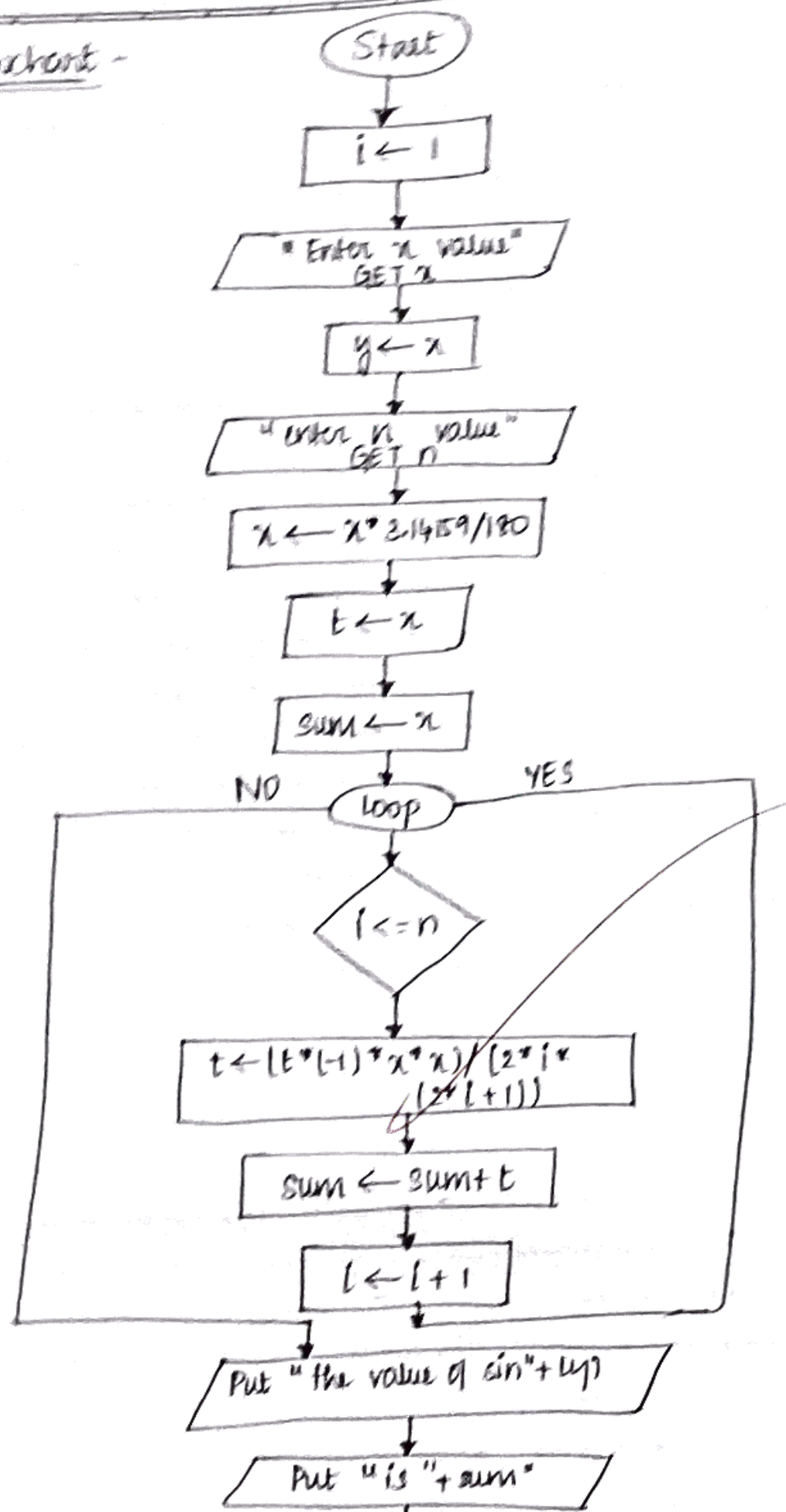
EXPT. TITLE :

PAGE NO 46

Observation - 10
Record - 10
Viva - 10

Total - 30

Flowchart -



DATE 6/09/21
EXP NO 10
EXPT TITLE
PAGE NO 47

Develop a program to compute sin(x) using Taylor series approximation. Compare your result with the built-in library function. Print both the results with appropriate messages.

Algorithm

Step-1: Start
Step-2: Read x
Step-3: degree ← x;
x ← x*(3.142/180.0)
Step-4: Read n
Step-5: term ← x
sum ← t
Step-6: term ← term*(-1)*x*x/2*i*(2*i+1);
sum = sum+term;
Step-7: Print the result
Sum of sine series = sum
sum using library function = sin(x)
Step-8: Stop

Output -

Sample - 1

Enter the value of degree 90
sin(90) = 1.00000 without using built in function sin(90)
= 1.00000 using built in function.

Sample - 2

Enter the value of degree 45
sin(45) = 0.707179 without using built in function
sin(45) = 0.707179 using built in function.

DATE 9/09/21 | EXPT. TITLE | PAGE NO 48
EXP NO.

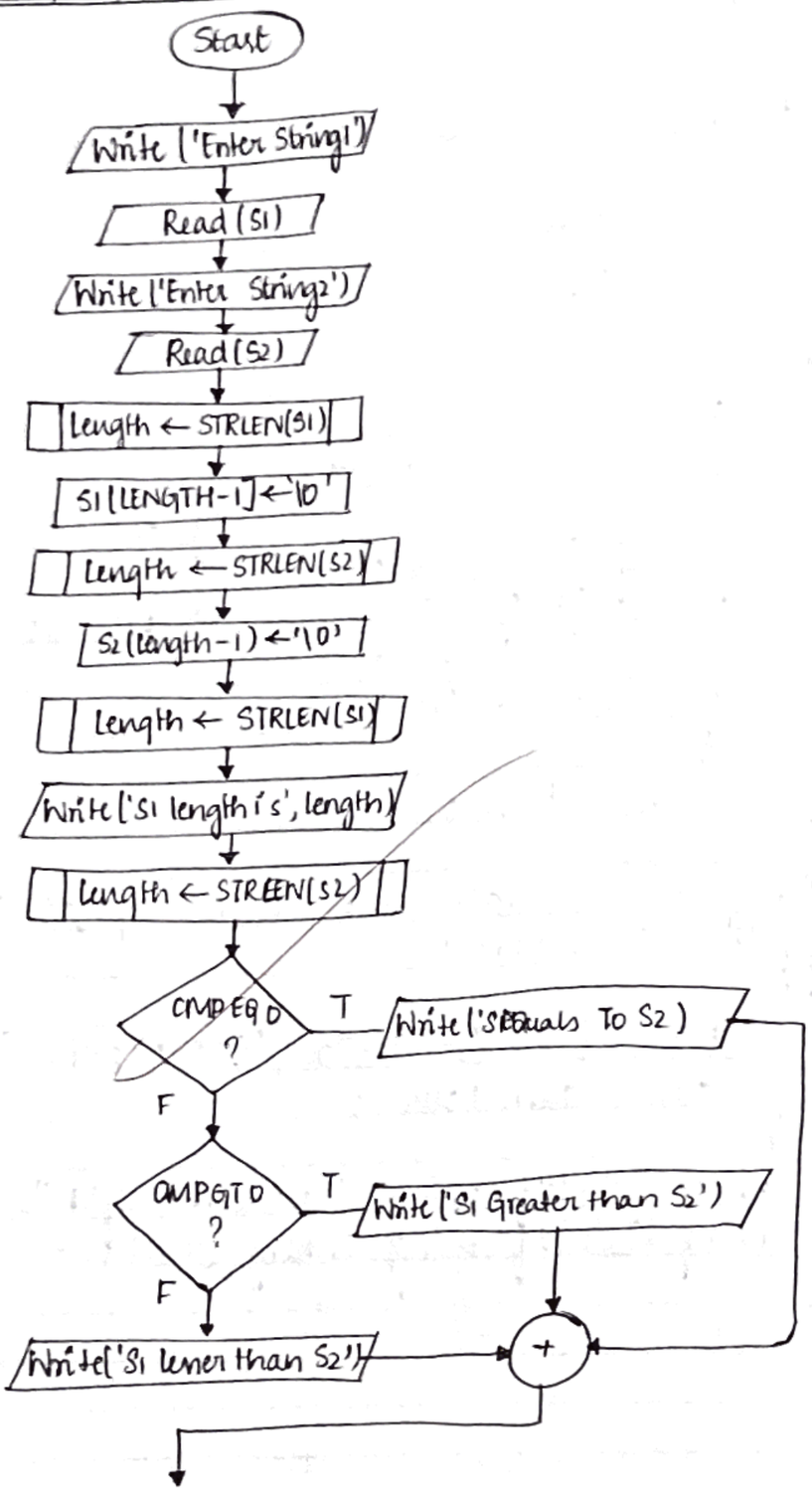
```
/* Program to calculate sin(x) using Taylor series*/

#include <stdio.h>
#include <math.h>
void main( )
{
    int i, n, degree;
    float x, sum, term;
    printf ("Enter the value of x: ");
    scanf ("%f", &x);
    printf ("Enter the value of n: ");
    scanf ("%d", &n);
    x = x*(3.142/180);
    term = x;
    sum = term;

/* Loop to calculate the value of sine*/
for (i=1; i<=n; i++)
{
    term = (term*(-1)*x*x)/(2*i*(2*i+1));
    sum = sum + term;
}
printf ("sin(%f) using sine series = %f\n", degree, sum);
printf ("sin(%f) using library function = %f\n", deg, sin(x));
}
```

Observation - 10
Record - 10
Viva - 10
Total - 30

Flowchart -



DATE 9/09/21 EXPT. TITLE: PAGE NO 49

EXP NO. 11

Write functions to implement string operations such as compare, ~~content~~ concatenate, string length, Convince the parameter paning techniques

Algorithm

Step-1 : Start

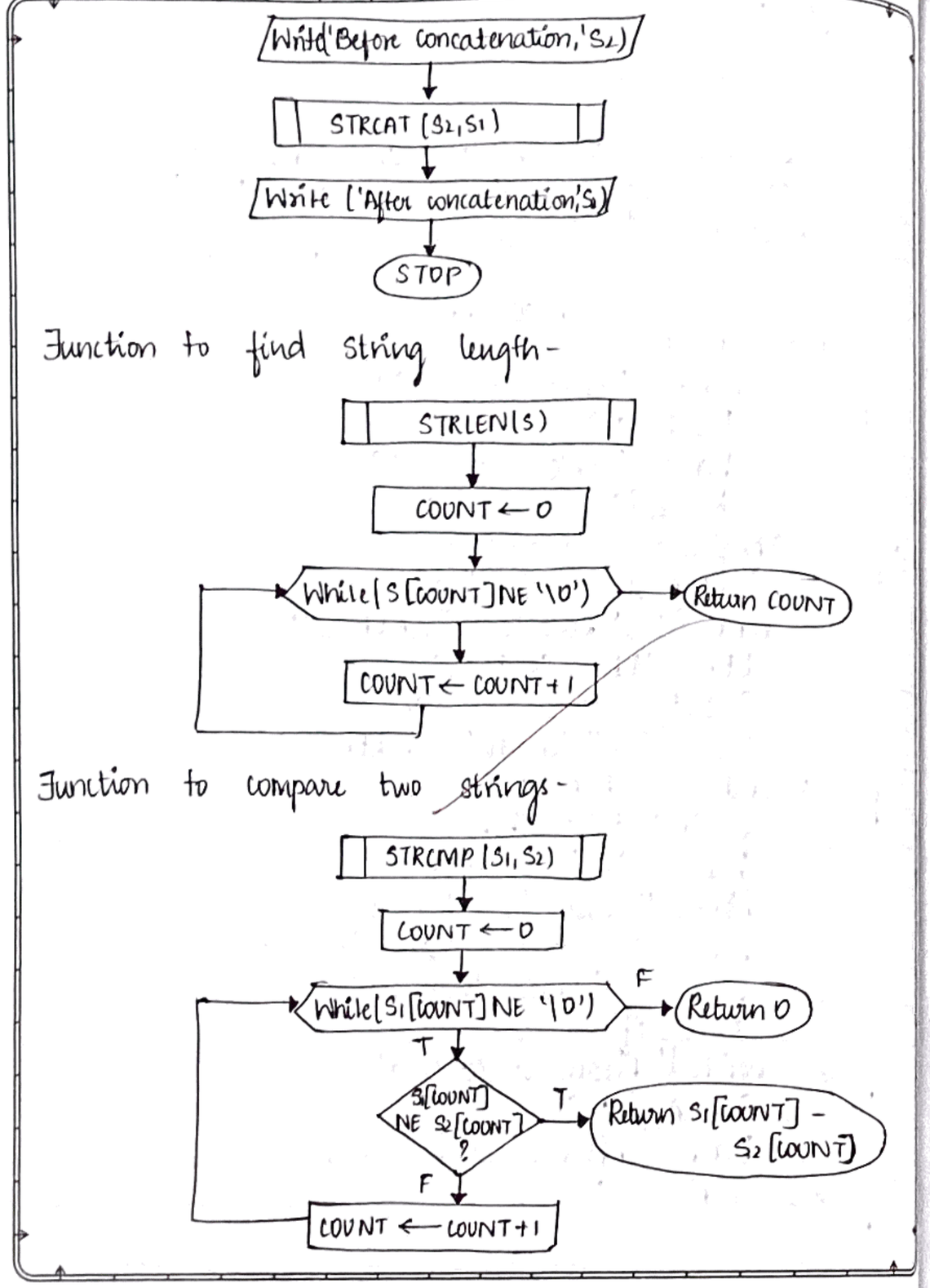
Step-2 : Write ["Enter String1"]
Read (S1)
Write ["Enter String2"]
Read (S2)

Step-3: LENGTH ← STRLEN(S1)
S1 [LENGTH-1] ← "\0"
Write (S1, "Length is", LENGTH)
LENGTH □ STRLEN(S2)
S2 [LENGTH-1] ← "\0"
Write (S1, "Length is", LENGTH)

Step-4: CMP ← STRCMP(S1,S2)
If [CMP EQ 0) Then
Write (S1, "Equals", S2)
~~LENGTH~~ else if (CMP GT 0) then
Write (S1, "Greater than" S2) else
Write (S1, "Lener than", S2)
end if

Step-5: Write ("Before Concatenation", S2)
STRCAT(S2,S1)
Write ("After Concatenation", S2)

Step-6: STOP



Function to find string length -

Function to compare two strings -

DATE
EXP NO
EXPT TITLE :
PAGE NO 50

Sub-Algorithm : STRLEN(S)

This sub-algorithm returns the length of the string S, S is a parameter of type array of characters, the variable COUNT is of type integer.

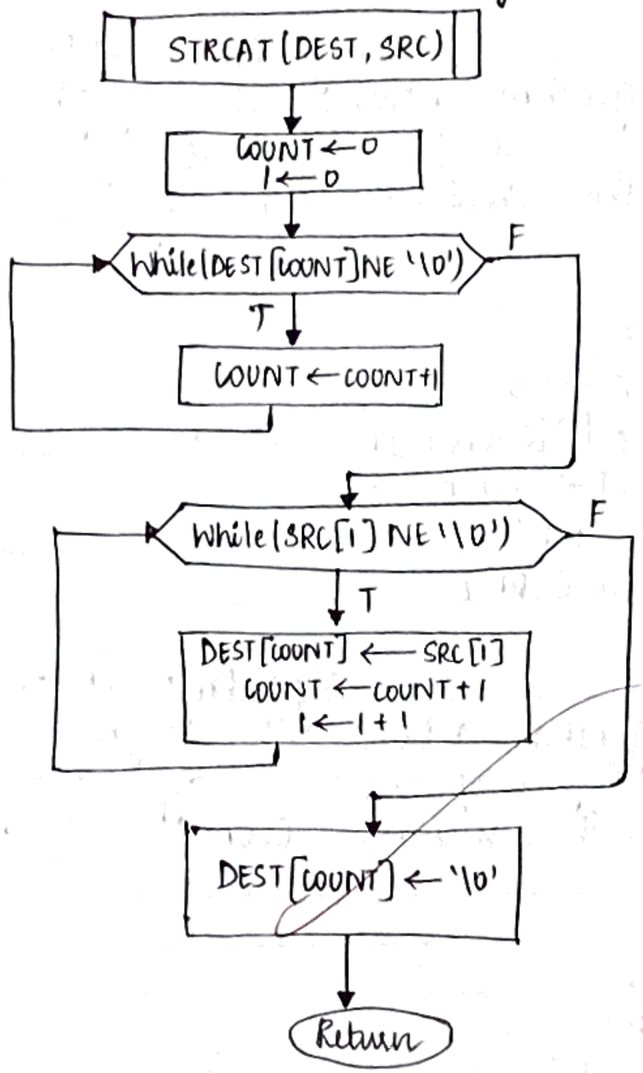
Step-1 : Start
Step-2 : COUNT ← 0
Step-3 : While (S[COUNT] NE NULL)
COUNT ← COUNT + 1
end while
Step-4 : Return COUNT

Sub-Algorithm : STRLEN(S)

The sub-algorithm returns the length of the string S, S is a parameter of type assay of characterers, the variable COUNT is of type integer.

Step-1 : Start
Step-2 : COUNT ← 0
Step-3 : While (S[COUNT] NE NULL)
COUNT ← COUNT + 1
end while
Step-4 : Return COUNT

Function to concatenate two strings -



Sub-Algorithm: STRCMP (S_1, S_2)

The sub-algorithm compares 2 strings S_1 and S_2 character by character and return the difference of the ASCII values of the compared characters of both strings. The variable COUNT is of type integer.

Step-1: Start
Step-2: COUNT ← 0
Step-3: While(S1[COUNT] NE NULL)
If (S1 [COUNT NE S_2 [COUNT]) then
return (S1 [COUNT - S_2 [COUNT])
else
COUNT ← COUNT - 1
end if
end while
return 0.

Sub-Algorithm : STRCAT(DEST, SRC)

This sub-algorithm adds SRC string to DEST string. DEST and SRC are parameters of type array of characters variables COUNT and I are of type integers.

Step-1 : Start
Step-2 : COUNT ← 0
I ← 0
Step-3 : While [DEST [COUNT] NE NULL]
COUNT ← COUNT + 1
end while
While [SRC[I] NE NULL]

DATE
EXP NO.
EXPT. TITLE
PAGE NO

DEST[COUNT] ← SRC(I)
COUNT ← COUNT - 1
I ← I + 1
Step-4 : Stop.

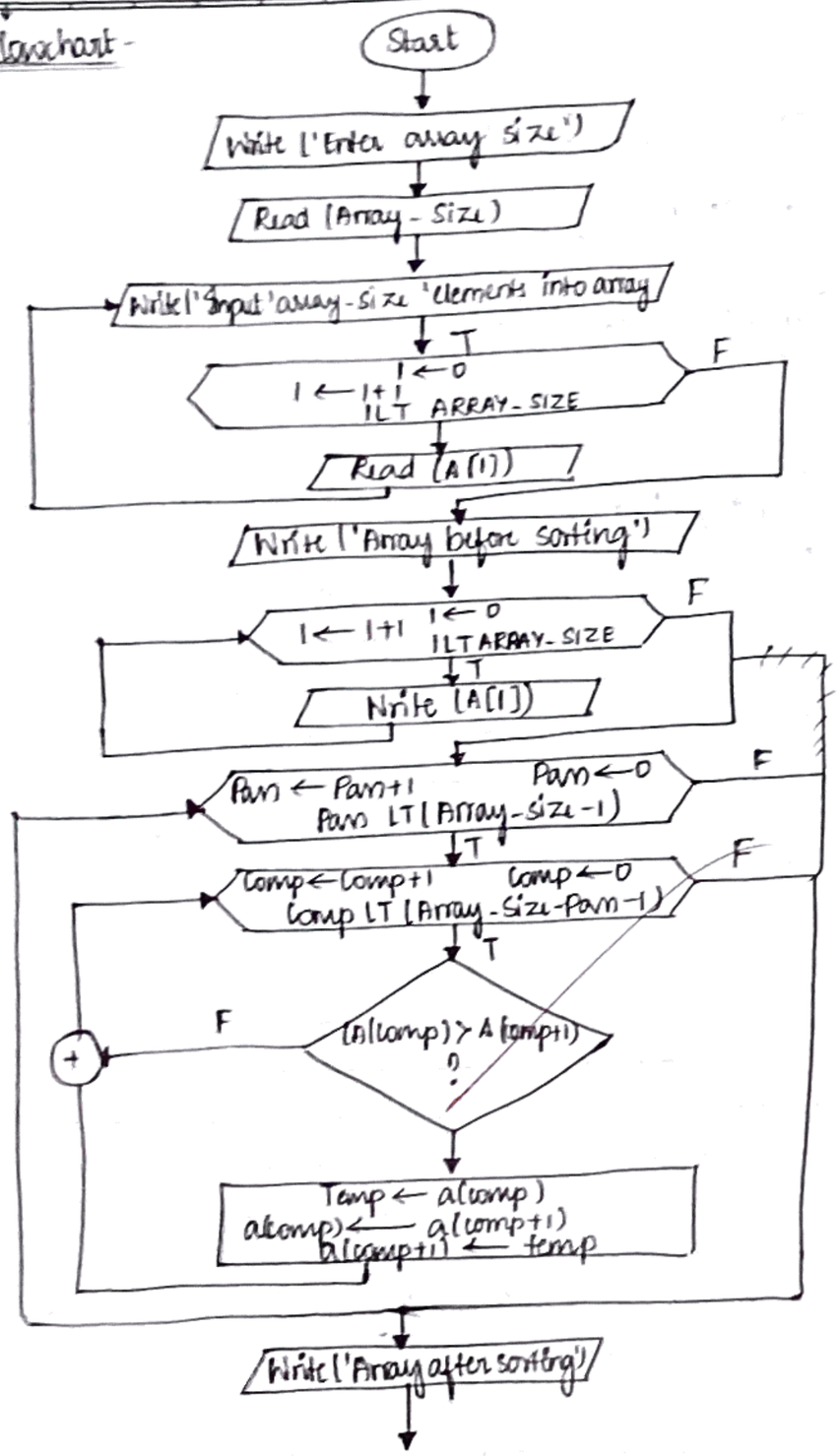
```
#include <stdio.h>
int STRLEN (char S1[]);
int STRCMP (char S1[], char S2[]);
void STRCAT (char dest[], char src[]);
void main()
{
    char S1[20], S2[20];
    printf("Enter the String1\n");
    scanf("%s", S1);
    printf("Enter the String2\n");
    scanf("%s", S2);
    int length;
    length = STRLEN(S1);
    printf("%s length is %d\n", S2, length); int comp;
    comp = STRCMP(S1, S2);
    if (comp==0)
    printf("%s is equal to %s\n", S1, S2);
    else if (comp>0)
    printf("%s is greater than %s\n", S1, S2);
    else
    printf("%s is lesser than %s\n" S1, S2);
    printf("before concatination\n");
```

```
printf("%s\n", s2);
STRCAT(s2,s1);
printf("After concatination\n");
printf("%s\n",s2);
return 0;
}
int STRLEN(char s[])
{
int count=0;
while(s[COUNT]!="\0')
{
count++;
}
return count;
}
int STRCMP(char s1[],char s2[])
{
int count=0;
while(s1[count]!='\0')
{
if(s1[count]!=s2[count]) return(s1[count]-s2[count]);
count++;
}
return 0;
}
void STRCAT(char dest[], char src[])
{
int count=0,i=0;
```

```
while (dest [count] != '\0')
count++;
while (src[i] != '\0')
{
dest [count] = src[i];
count ++;
i++;
}
dest [count] = '\0';
return;
}
```

Observation - 10
Record - 10
Viva - 10
Total - 30

Flowchart -



DATE 9/09/21	EXPT TITLE :	PAGE NO 54
EXP NO. 12		

Develop a program to sort the given set of N numbers using Bubble Sort.

Purpose - This program demonstrates NESTED FOR loop.

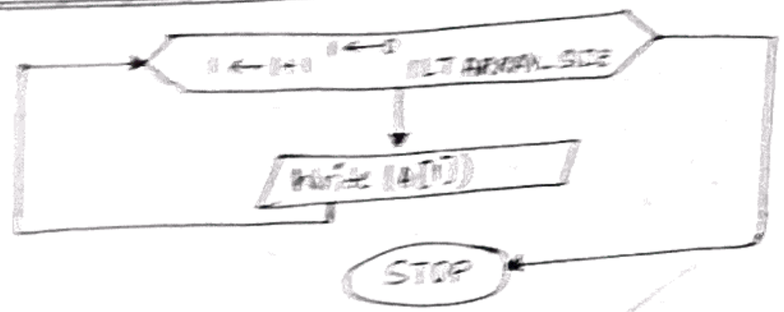
Procedure - To read an array of elements a[]. While iterating, compare each pair of adjacent items in every pass. If the former value is greater than the latter one, their positions are swapped. Over a no. of passes, at most equal to the no. of elements in the list. Then print sorted array elements.

Input - Number of elements - n
An array of unsorted elements - a[]

Expected Output - An array of sorted elements - a[]

Algorithm -

Step-1 : Start
Step-2 : Read n
Step-3 : Repeat for i=0 to n-1
read a[i]
Step-4 : Repeat for i=0 to n-1
print a[i]
Step-5 : Repeat through step 5 ~~for~~
for pass ← 1 to n-1
Repeat for comp ← 0 to n-pass



DATE:
EXP NO:
EXPT TITLE:
PAGE NO: 36

if (a[temp] > a[temp+1])
temp ← a[temp]
a[temp] ← a[temp+1]
a[temp+1] ← temp
Step-6: Repeat for i ← 0 to n-1
print a[i]
Step-7: Stop.

/* Program to sort the given elements using Bubble Sort */

#include <stdio.h>
void main()
{
int a[50], i, temp, n;
printf("Enter the value of n\n");
scanf("%d", &n);
printf("Enter the array elements\n");
for(i=0; i<n; i++)
scanf("%d", &a[i]);
printf("The given array elements are :\n");
for(i=0; i<n; i++)
printf("%d\t", a[i]);

/* Perform Bubble Sort */
for(int pass = 1; pass <n; pass++)
{

Output -

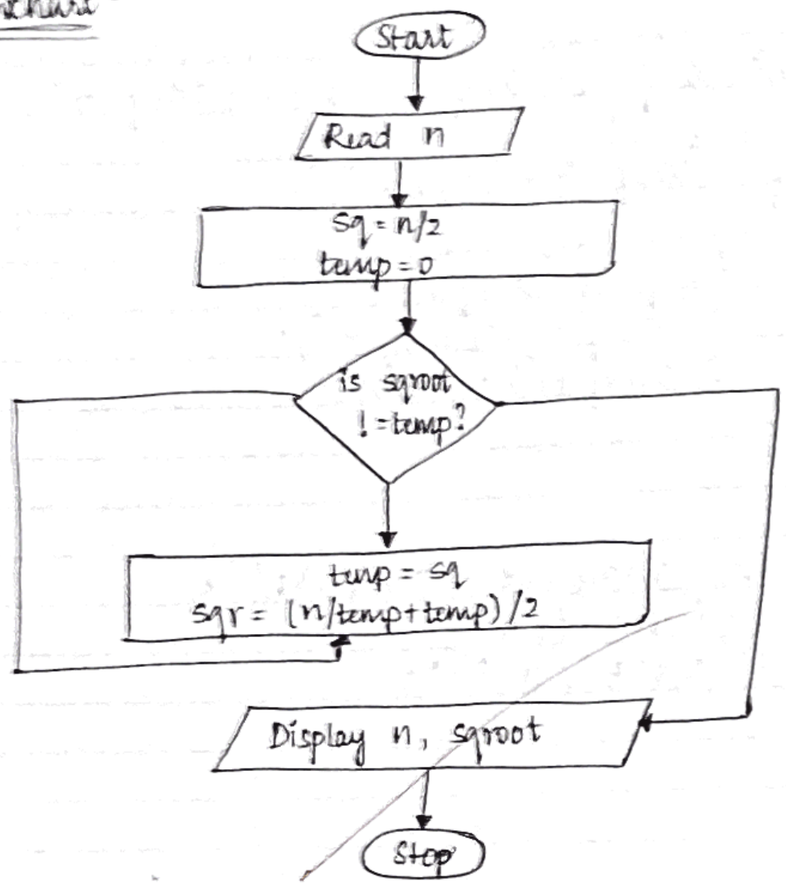
```
Enter the size of an array : 5
Enter the 5 elements of an array :
5 4 3 2 1
The given array before sorting is 5
4
3
2
1
The corted array is :
1
2
3
4
5
```



```
for (int comp = 0; comp < n - pass; comp++)
{
    if (a[comp] > a[comp+1])
    {
        temp = a[comp];
        a[comp] = a[comp+1];
        a[comp+1] = temp;
    }
}
printf ("The array after sorting\n");
for (i=0; i<n; i++)
printf ("%d\t", a[i]);
}
```

Observation	- 10
Record	- 10
Viva	- 10
Total	- 30

Flowchart -



DATE 9/09/21 | EXPT TITLE : | PAGE NO. 51
EXP NO. 13

Develop a program to find the square root of given number n for and execute all the possible inputs. Dont use library sqrt(n).

Algorithm

Step-1: Start
Step-2: Read n
Step-3: term ← 0;
sq ← n/2;
Step-4: temp ← sq
sq ← (n/temp+temp)/2
end while
Step-7: Print the result
Square root of number is = sq
Square root using library function = sqrt(n)
Step-8: Stop

```
/* Program to find the square root of a number */
#include <stdio.h>
void main()
{
    float sq, temp, n;
    printf("Enter any number\n");
    scanf("%f", &n);
    temp = 0;
    sq = n/2;
```

Output -

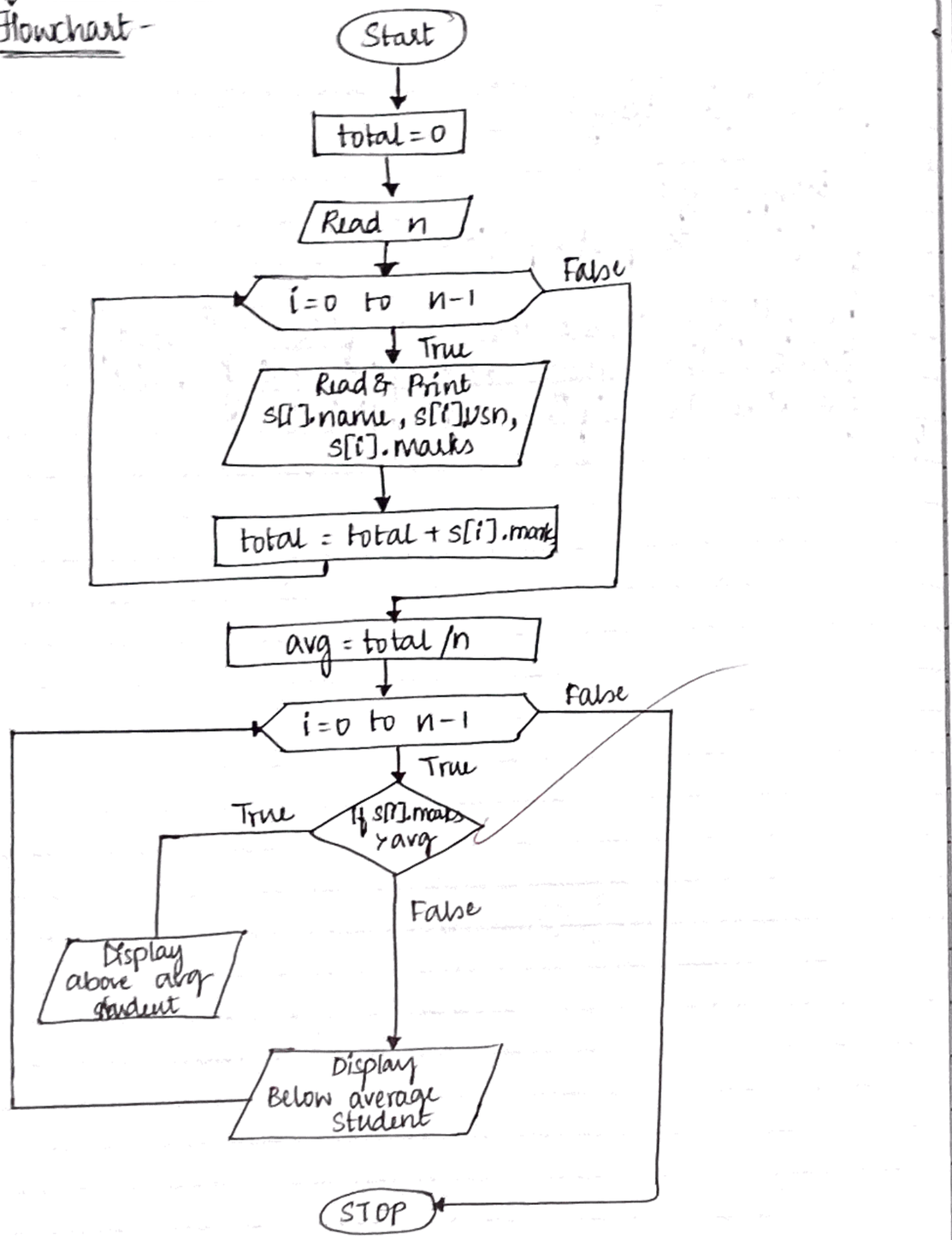
1) Enter a number to which square root to be found: 4
Square root of a real number 4.00 is 2.00

2) Enter a number to which square root is: 8.4
Square root of real number 8.4 is 2.8983

3) Enter a number to which square root : -7
Invalid Number.

```
while (sq != temp)
{
    temp = sq;
    sq = (n/temp+temp)/2;
}
printf("The square root of '%d' is '%f'", n, sq);
printf("The square root of '%d' using built in
function is '%f'", n, sqrt(n));
}
```

Flowchart -



DATE 9/09/21 EXPT. TITLE : PAGE NO. 59

EXP. NO. 14

Implement structures to read, write and compute avg-marks and the students above and below the avg-marks of a class of N students.

Algorithm

Step-1 : Input no. of students, Read n

Step-2 : Initilize total = 0

Step-3 : Input details of students i.e, name, usn and marks for all n students
for i=0 to n-1 do
Read s[i], name, s[i]usn, s[i]marks

Step-4 : Display details of students i.e, name, usn and marks for all n students
for i=0 to n-1 do
print s[i]name, s[i].usn, s[i].marks
end i for loop

Step-5 : For i=0 to n-1 do
total = total/n
end i for loop
end i for loop

Step-6 : Calculate avg, avg = total/n

Step-7 : Display whether a student is below avg or above avg
for i=0 to n-1 do
if (s[i].marks > avg)
Display "Student is above avg"
otherwise

Display "Student is below avg"
end 'i' for loop

Step-8 : Stop.

Program-

```
#include <stdio.h>
struct student
{
    char name[20];
    char usn[10];
    int marks;
} s[10];

void main()
{
int i,n, total =0;
float avg = 0.0;
printf("\n Enter the number of students:");
scanf("%d",&n);
for(i=0;i<n;i++)
{
    printf("Enter student %d details", i+1);
    printf("Enter USN:");
    scanf("%s",s[i].usn);
    printf("Enter name:");
    scanf("%s", s[i].name);
    printf("Enter marks:");
```

Output -

```
Enter the number of students: 3
Enter the details of student 1
Name : Raju
USN : 4AD17CS036
Marks : 67

Enter the details of student 2
Name : Sahana
USN : 4AD17CS405
Marks : 77

The average marks for the class: 75.67.
The student Raju has scored below average. The
student Shana has scored above average
```

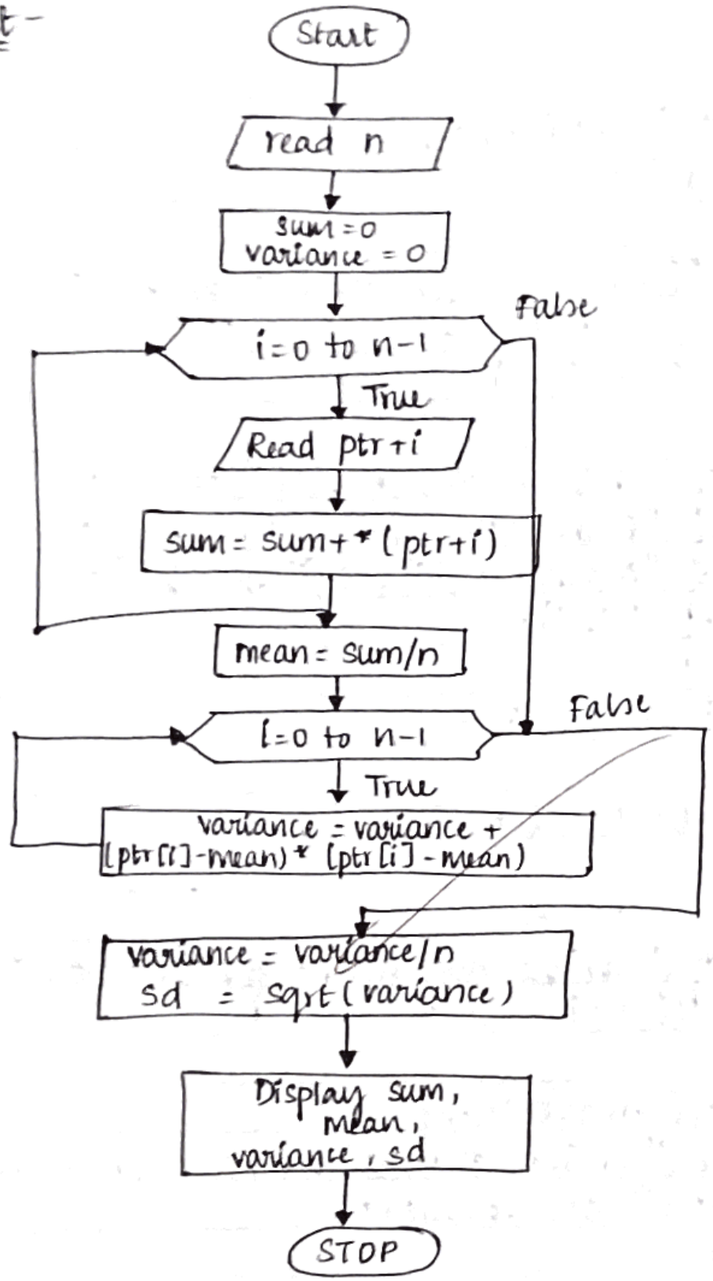


```
scanf("%d", &s[i].marks);
printf("\n");
}
printf("Displaying Information : \n\n");

for(i=0; i<n; i++)
{
    printf("\nUSN : %s\n", s[i].usn);
    printf("Name : %s\n", s[i].name);
    printf("Marks : %d", s[i].marks);
    printf("\n");
}
for(i=0; i<n; i++)
{
    total = total + s[i].marks;
}
avg = total/n;
printf("\n The avg marks for the class is : %f\n", avg);
for (i=0; i<n; i++)
{
if(s[i].marks > avg)
printf("\n The student %s has scored above
avg\n", s[i].name);
else
printf("\nThe student %s has scored below avg
\n", s[i].name);
}
}
```

Observation - 10
Record - 10
Viva - 10
Total - 30

Flowchart -



DATE 9/09/21 EXP. NO. 15 EXPT. TITLE: PAGE NO. 62

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

Algorithm

Step-1 : Start
Step-2 : Input the number of elements, Read n.
Step-3 : Initilize pointer so that it points to the beginning of the array
Step-4 : Initilize varibales, variance = 0, sum = 0
Step-5 : Input array elements
for i=0 to n-1 do
Read array element using pointer, Read a value to be stored at location (ptr+i)
sum = sum + *(ptr+i)
end i for loop
Step-6 : Calculate mean, mean = sum/n
Step-7 : Calculate variance
for i=0 to n-1 do
variance = variance + (ptr[i]-mean)*(ptr[i]-mean)
end i for loop
variance = variance/n
Step-8 : Calculate standard deviation
sd = sqrt(variance)
Step-9 : Display the values of sum, mean, variance

and standard deviation

Step-10: Stop

```
/* Program to compute sum, mean and standard
   deviation of all elements */

#include<stdio.h>
#include<math.h>
void main()
{
    int i,n;
    float mean=0.0, variance=0.0, sd=0.0, num[100],
     sum=0.0; float *ptr;
    ptr = num;
    printf("\nEnter the number of values:");
    scanf("%d",&n);
    printf("\nEnter %d values\n",n);
    for(i=0;i<n;i++)
    {
      scanf("%f",ptr+i);
      sum += *(ptr+i);
    }
    mean = sum/n;
    for(i=0;i<n;i++)
    {
        variance += (ptr[i]-mean)*(ptr[i]-mean);
    }
    variance/n=n;
```

Output -

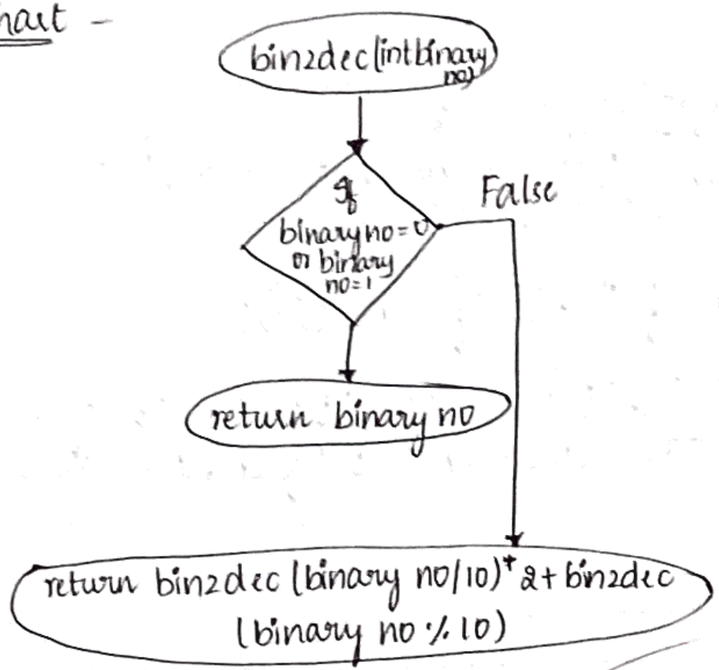
```
Enter the number of values : 4
Enter 4 values
1.1  2.2  3.3  4.4
The values entered are
1.1
2.2
3.3
4.4
    Sum = 11.0
    Mean = 2.75
    Variance = 1.5125
    Standard Deviation = 1.22984
```



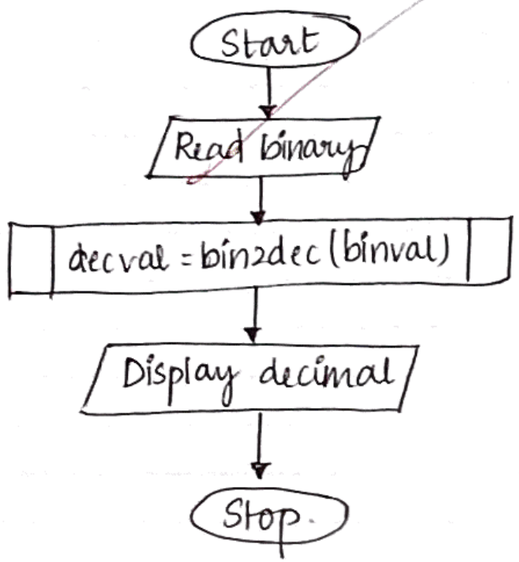
```
    sd = sqrt(variance);
    printf("\nThe values entered are");
    for(i=0; i<n; i++)
    {
        printf("\n\t%f", ptr[i]);
    }
    printf("\n\t sum = %f\n\t mean = \t %f\n\t
    standard deviation = \t%f\n", sum, mean, vari, sd);
}
```

Observation - 10
Record - 10
Viva - 10
Total - 30

Flowchart -



Flowchart for main or driver -



DATE 9/09/21 EXP. NO. 16 EXPT. TITLE : PAGE NO 65

Implement Recursive functions for Binary to Decimal Conversion.

Purpose - This program demonstrates RECURSION
Procedure - Input binary number and call the reursive funktion convert the translating binary number to decimal number
Input - A binary number bin
Expected Output - Decimal number dec

Algorithm

Step-1 : Start
Step-2 : Input binary number, Read binval
Step-3 : Call recursive function that converts binary no to decimal value.
decval = bin2dec(binval)
Step-4 : Display decimal number, Display decval
Step-5 : Stop.

Output -

Sample-1

Enter the binary value:
11111 Decimal equivalent of 11111 is
31.

Sample-2

Enter the binary value:
1111111 Decimal
equivalent of 1111111 is 127

Sample-3

Enter the binary value:
11011 Decimal
equivalent of 11011 is 27

DATE
EXP NO.
EXPT. TITLE :
PAGE NO 66

```
/* Program to convert Binary to decimal */
#include <stdio.h>
int bin2dec(int n);
void main()
{
    int binval, decval;
    printf("\n Enter the binary value:");
    scanf("%d", &binval);
    decval = bin2dec(binval);
    printf("\n Decimal equivalent of %d is %d",
        binval, decval);
}
int bin2dec(int binaryno)
{
    if (binaryno == 0) || binaryno == 1)
    return binaryno;
    else
    return bin2dec(binaryno/10)*2 + bin2dec
        (binaryno % 10);
}
```

Observation - 10
Record - 10
Viva - 10
Total - 30