

**Laboratory Manual for**

**PROGRAMMING FOR PROBLEM**

**SOLVING (3110003)**

**B.E. Semester 1**

**Computer Engineering**



**L. D. College of Engineering College, Ahmedabad**



**Directorate of Technical Education, Gandhinagar,**

**Gujarat**



---

# **L. D. College of Engineering College, Ahmedabad**

## **Certificate**

**This is to certify that Mr./Ms. \_\_\_\_\_**  
**Enrollment No. \_\_\_\_\_ of B.E. Semester \_\_\_\_\_ Computer**  
**Engineering of this Institute (GTU Code: 028) has satisfactorily completed the**  
**Practical / Tutorial work for the subject \_\_\_\_\_ for the**  
**academic year 202\_\_-2\_\_.**

**Place: \_\_\_\_\_**

**Date: \_\_\_\_\_**

**Name and Sign of Faculty member**

**Head of the Department**



---

## Preface

Main motto of any laboratory/practical/field work is for enhancing required skills as well as creating ability amongst students to solve real time problem by developing relevant competencies in psychomotor domain. By keeping in view, GTU has designed competency focused outcome-based curriculum for engineering degree programs where sufficient weightage is given to practical work. It shows importance of enhancement of skills amongst the students and it pays attention to utilize every second of time allotted for practical amongst students, instructors and faculty members to achieve relevant outcomes by performing the experiments rather than having merely study type experiments. It is must for effective implementation of competency focused outcome-based curriculum that every practical is keenly designed to serve as a tool to develop and enhance relevant competency required by the various industry among every student. These psychomotor skills are very difficult to develop through traditional chalk and board content delivery method in the classroom. Accordingly, this lab manual is designed to focus on the industry defined relevant outcomes, rather than old practice of conducting practical to prove concept and theory.

By using this lab manual students can go through the relevant theory and procedure in advance before the actual performance which creates an interest and students can have basic idea prior to performance. This in turn enhances pre-determined outcomes amongst students. Each experiment in this manual begins with competency, industry relevant skills, course outcomes as well as practical outcomes (objectives). The students will also achieve safety and necessary precautions to be taken while performing practical.

This manual also provides guidelines to faculty members to facilitate student centric lab activities through each experiment by arranging and managing necessary resources in order that the students follow the procedures with required safety and necessary precautions to achieve the outcomes. It also gives an idea that how students will be assessed by providing rubrics.

C programming is a general-purpose, procedural, imperative computer programming language. C is robust language and has rich set of built-in functions, data types and operators which can be used to write any complex program. Program written in C are efficient due to availability of several data types and operators. C has the capabilities of an assembly language (low level features) with the feature of high level language so it is well suited for writing both system software and application software. C is highly portable language i.e. code written in one machine can be moved to other which is very important and powerful feature.

Utmost care has been taken while preparing this lab manual however always there is chances of improvement. Therefore, we welcome constructive suggestions for improvement and removal of errors if any.



### Practical – Course Outcome matrix

#### Course Outcomes (COs):

**CO-1:** Formulate algorithm/flowchart for given arithmetic and logical problem

**CO-2:** Translate algorithm/flowchart into C program using correct syntax and execute it

**CO-3:** Write programs using conditional, branching, iteration, and recursion

**CO-4:** Decompose a problem into function

**CO-5:** Develop an application using the concepts of array, pointer, structure, and file management to solve engineering and/or scientific problems

Sr. No.	Objective(s) of Experiment	CO 1	CO 2	CO 3	CO 4	CO 5
1.	Write an algorithm & draw a flowchart for following problems. (a) accepting two numbers and performing addition, subtraction, division and multiplication on them (b) to check whether given number is even or odd	√				
2.	Write an algorithm & draw a flowchart for following problems. (a) to print first 10 Fibonacci numbers (b) to find out largest number from three numbers	√				
3.	Write a C program for following problems. (a) Write a program to that performs as calculator (addition, multiplication, division, subtraction). (b) Write a program to find area of triangle ( $a = h * b * .5$ ) $a$ = area $h$ = height $b$ = base		√			
4.	Write a C program for following problems. (a) Write a C program to interchange two numbers. (b) Write a program to compute Fahrenheit from centigrade ( $f = 1.8 * c + 32$ )		√			
5.	Write a C Program for following problems using branching, iteration and recursion. (a) Write a program to read marks of a student from keyboard whether the student is pass or fail( using if else) (b) Write a program to read three numbers from keyboard and find out maximum out of these three. (nested if else)			√		
6.	Write a C Program for following problems using branching, iteration and recursion. (a) Write a C program to read no 1 to 7 and print			√		



	<p>relatively day Sunday to Saturday. (Switch statement)</p> <p>(b) Write a C program to find factorial of a given number.</p> <p>(c) Write a program to print following patterns :</p> <p>i) *                      ii) *                      iii) * * *</p> <p>  * *                      * *                      * *</p> <p>  * * *                      * * *                      *</p>					
7.	<p>Write a C Program using concept of function.</p> <p>(a) Write a program that defines a function to add first n numbers.</p> <p>(b) Write a program using function to find maximum number from two numbers.</p>				√	
8	<p>Write a C Program using concept of function.</p> <p>(a) Write a program to exchange two numbers by using function</p> <p>(b) Write a program to reverse the number using function.</p>				√	
9	<p>Write a C Program using the concept of an array.</p> <p>(a) Write a C program to find out the Maximum and Minimum number from given 10 numbers in an array.</p> <p>(b) Write a program to sort the elements of an array in ascending order.</p>					√
10	<p>Write a C Program using the concept of pointer, structure &amp; File.</p> <p>(a) Write a program to access elements using pointer.</p> <p>(b) Design a structure student_record to contain name, branch and total marks obtained. Develop a program to read data for 10 students in a class and print them.</p> <p>(c) A file named data contains series of integer numbers. Write a c program to read all numbers from file and then write all odd numbers into file named "odd" and write all even numbers into file named "even". Display all the contents of these file on screen</p>					√



---

### Industry Relevant Skills

The following industry relevant competency are expected to be developed in the student by undertaking the practical work of this laboratory.

1. Will be able to solve any problem by writing an algorithm & drawing a flowchart.
2. Will be able to develop an application software by using the concepts of functions, arrays, file management, loops, branching etc...

### Guidelines for Faculty members

1. Teacher should provide the guideline with demonstration of practical to the students with all features.
2. Teacher shall explain basic concepts/theory related to the experiment to the students before starting of each practical
3. Involve all the students in performance of each experiment.
4. Teacher is expected to share the skills and competencies to be developed in the students and ensure that the respective skills and competencies are developed in the students after the completion of the experimentation.
5. Teachers should give opportunity to students for hands-on experience after the demonstration.
6. Teacher may provide additional knowledge and skills to the students even though not covered in the manual but are expected from the students by concerned industry.
7. Give practical assignment and assess the performance of students based on task assigned to check whether it is as per the instructions or not.
8. Teacher is expected to refer complete curriculum of the course and follow the guidelines for implementation.

### Instructions for Students

1. Students are expected to carefully listen to all the theory classes delivered by the faculty members and understand the COs, content of the course, teaching and examination scheme, skill set to be developed etc.
2. Students shall organize the work in the group and make record of all observations.
3. Students shall develop maintenance skill as expected by industries.
4. Student shall attempt to develop related hand-on skills and build confidence.
5. Student shall develop the habits of evolving more ideas, innovations, skills etc. apart from those included in scope of manual.
6. Student shall refer technical magazines and data books.
7. Student should develop a habit of submitting the experimentation work as per the schedule and s/he should be well prepared for the same.



---

## Common Safety Instructions

Students are expected to

- 1) Switch on the PC carefully (not to use wet hands)
- 2) Shutdown the PC properly at the end of your Lab
- 3) Carefully handle the peripherals (Mouse, Keyboard, Network cable etc)
- 4) Use Laptop in lab after getting permission from Teacher



## Index (Progressive Assessment Sheet)

Sr. No.	Objective(s) of Experiment	Page No.	Date of performance	Date of submission	Assessment Marks	Sign. of Teacher with date	Remarks
0	Write the Following 1. Vision & Mission of DTE, LDCE and Computer Department 2. Program Outcome of Computer Engineering 3. PSOs and PEOs of Computer Engineering Department 4. Course outcomes of PPS						
1	Write an algorithm & draw a flowchart for following problems. (a) accepting two numbers and performing addition, subtraction, division and multiplication on them (b) to check whether given number is even or odd						
2	Write an algorithm & draw a flowchart for following problems. (a) to print first 10 Fibonacci numbers (b) to find out largest number from three numbers						
3	Write a C program for following problems. (a) Write a program to that performs as calculator (addition, multiplication, division, subtraction). (b) Write a program to find area of triangle ( $a = \frac{h*b}{2}$ ) a = area h = height b = base						
4	Write a C program for following problems. (a) Write a C program to interchange two numbers. (b) Write a program to compute Fahrenheit from centigrade ( $f = 1.8*c + 32$ )						
5	Write a C Program for following problems using branching, iteration and recursion. (a) Write a program to read marks of a student from keyboard whether the student is pass or fail( using if else) (b) Write a program to read three numbers from keyboard and find out maximum out of these three. (nested if else)						





6	<p>Write a C Program for following problems using branching, iteration and recursion.</p> <p>(a) Write a C program to read no 1 to 7 and print relatively day Sunday to Saturday. (Switch statement)</p> <p>(b) Write a C program to find factorial of a given number.</p> <p>(c) Write a program to print following patterns</p> <p>i) *                      ii) *                      iii) * * *</p> <p>  * *                      * *                      * *</p> <p>  * * *                      * * *                      *</p>						
7	<p>Write a C Program using concept of function.</p> <p>(a) Write a program that defines a function to add first n numbers.</p> <p>(b) Write a program using function to find maximum number from two numbers.</p>						
8	<p>Write a C Program using concept of function.</p> <p>(a) Write a program to exchange two numbers by using function</p> <p>(b) Write a program to reverse the number using function.</p>						
9	<p>Write a C Program using the concept of an array.</p> <p>(a) Write a C program to find out the Maximum and Minimum number from given 10 numbers in an array.</p> <p>(b) Write a program to sort the elements of an array in ascending order.</p>						
10	<p>Write a C Program using the concept of pointer, structure &amp; File.</p> <p>(a) Write a program to access elements using pointer.</p> <p>(b) Design a structure student_record to contain name, branch and total marks obtained. Develop a program to read data for 10 students in a class and print them.</p> <p>(c) A file named data contains series of integer numbers. Write a c program to read all numbers from file and then write all odd numbers into file named “odd” and write all even numbers into file named “even”. Display all the contents of these file on screen</p>						

**DTEs' Vision:****Vision:**

- ✓ To provide globally competitive technical education
- ✓ Remove geographical imbalances and inconsistencies
- ✓ Develop student friendly resources with a special focus on girls' education and support to weaker sections
- ✓ Develop programs relevant to industry and create a vibrant pool of technical professionals

**Vision and Mission of LDCE:****Vision:**

- ✓ To contribute for sustainable development of nation through achieving excellence in technical education and research while facilitating transformation of students into responsible citizens and competent professionals.

**Mission:**

- ✓ To impart affordable and quality education in order to meet the needs of industries and achieve excellence in teaching-learning process.
- ✓ To create a conducive research ambience that drives innovation and nurtures research-oriented scholars and outstanding professionals.
- ✓ To collaborate with other academic & research institutes as well as industries in order to strengthen education and multidisciplinary research.
- ✓ To promote equitable and harmonious growth of students, academicians, staff, society and industries, thereby becoming a center of excellence in technical education.
- ✓ To practise and encourage high standards of professional ethics, transparency and accountability.

**Vision and Mission of Computer Department:****Vision:**

- ✓ To achieve academic excellence in Computer Engineering by providing value based education.

**Mission:**

- ✓ To produce graduates according to the needs of industry, government, society and scientific community.



- ✓ To develop partnership with industries, research and development organizations and government sectors for continuous improvement of faculties and students.
- ✓ To motivate students for participating in reputed conferences, workshops, seminars and technical events to make them technocrats and entrepreneurs.
- ✓ To enhance the ability of students to address the real life issues by applying technical expertise, human values and professional ethics.
- ✓ To inculcate habit of using free and open source software, latest technology and soft skills so that they become competent professionals.
- ✓ To encourage faculty members to upgrade their skills and qualification through training and higher studies at reputed universities.

### **Programme Outcomes (POs)**

1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for, sustainable development.
8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.



- 
9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
  10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
  11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
  12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

### **Program Specific Outcomes (PSOs)**

- Graduates will be able to explore and propose effective solutions to the problems in the area of Computer Engineering as per the needs of society and industry.
- Graduates will be able to apply standard practice and strategies to develop quality software products using modern techniques, programming skills, tools & an open ended programming environment and work in a team.
- Graduates will manifest the skills of continuous learning in the fast changing field of Computer Engineering.

### **Program Educational Objectives (PEOs)**

- Provide computing solutions of complex problems as per business and societal needs.
- Procure requisite skills to pursue entrepreneurship, research and development, and imbibe high degree of professionalism in the fields of computing.
- Embrace life-long learning and remain continuously employable.
- Work and excel in a highly competence supportive, multicultural and professional environment which abiding to the legal and ethical responsibilities.



---

## **CO-1: Formulate algorithm/flowchart for given arithmetic and logical problem**

### **Algorithm**

An algorithm is a sequence of instructions that are carried out in a predetermined sequence in order to solve a problem.

### **Features of the algorithm**

It defines several important features of the algorithm, including:

**Inputs:** Algorithms must receive inputs that can be represented as values or data.

**Output:** The algorithm should produce some output. It can be a consequence of a problem or a solution designed to solve it.

**Clarity:** Algorithms must be precisely defined, using unambiguous instructions that a computer or other system can follow unambiguously.

**Finiteness:** The algorithm requires a limited steps. It means that it should be exited after executing a certain number of commands.

**Validity:** The algorithm must be valid. In other words, it should be able to produce a solution to the problem that the algorithm is designed to solve in a reasonable amount of time.

**Effectiveness:** An algorithm must be effective, meaning that it must be able to produce a solution to the problem it is designed to solve in a reasonable amount of time.

**Generality:** An algorithm must be general, meaning that it can be applied to a wide range of problems rather than being specific to a single problem.

### **How to write an algorithm?**

Step 1: First define the problem which you want to solve.

Step 2: Think out the steps to solve the problem, arrange all those steps into sequential order so we can realize the correct solution for the problem.

Step 3: Write an algorithm in pseudo code or a programming language.

Step 4: Test your algorithm to make sure it is correct and efficient.

Example of Algorithm:



---

**Algorithm for finding the average of three numbers is as follows –**

Step 1: Start

Step 2: Read 3 numbers a, b, c

Step 3: Compute  $\text{sum} = a + b + c$

Step 4: Compute  $\text{average} = \text{sum}/3$

Step 5: Print average value

Step 6: Stop

**Algorithm for finding the area of rectangle is as follows –**

Step 1: Start

Step 2: Read Breadth (b) and Length (l) for the Rectangle.

Step 3: Calculate Area (a) = Length \* Breadth

Step 4: Print Area (a)

Step 5: Stop

**Algorithm for finding the larger number from the two numbers is as follows –**

Step 1: Start

Step 2: Read Two Numbers a, b.

Step 3: Compare two numbers. IF (a>b) then go to step 5.

Step 4: Print “b is greater”, go to step 6.

Step 5: Print “a is greater”, go to step 6.

Step 6: Stop



---

## Flowchart

Diagrammatic representation of an algorithm is called flow chart.

### Flowchart Symbols

The different flowchart symbols have different conventional meanings.

The various symbols used in Flowchart Designs are given below.

**Terminal Symbol:** In the flowchart, it is represented with the help of a circle for denoting the start and stop symbol. The symbol given below is used to represent the terminal symbol.



**Input/output Symbol:** The input symbol is used to represent the input data, and the output symbol is used to display the output operation.

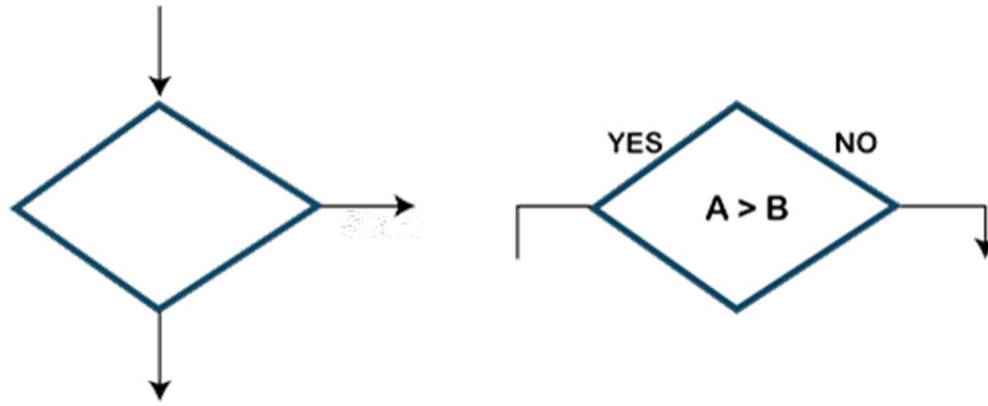


**Processing Symbol:** It is represented in a flowchart with the help of a rectangle box used to represent the arithmetic and data movement instructions. The symbol given below is used to represent the processing symbol.

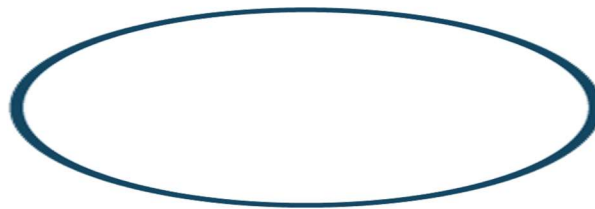




Decision Symbol: Diamond symbol is used for represents decision-making statements. The symbol given below is used to represent the decision symbol.



Connector Symbol: The connector symbol is used if flows discontinued at some point and continued again at another place. The following symbol is the representation of the connector symbol.



Flow lines: It represents the exact sequence in which instructions are executed. Arrows are used to represent the flow lines in a flowchart. The symbol given below is used for representing the flow lines:



Hexagon symbol (Flat): It is used to create a preparation box containing the loop setting statement. The symbol given below is used for representing the Hexagon symbol.





Internal storage symbol: The symbol given below is used to represent the internal storage symbol.



### **Advantages of Flowchart in C:**

Following are the various advantages of flowchart:

Communication: A flowchart is a better way of communicating the logic of a program.

Synthesis: Flowchart is used as working models in designing new programs and software systems.

Efficient Coding: Flowcharts act as a guide for a programmer in writing the actual code in a high-level language.

Proper Debugging: Flowcharts help in the debugging process.

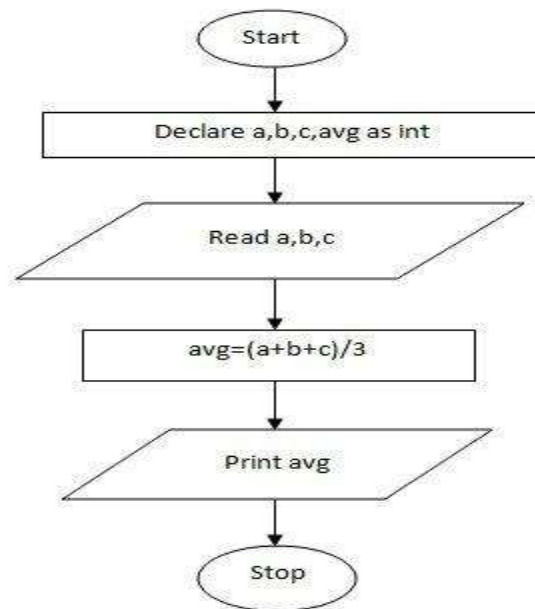
Effective Analysis: Effective analysis of logical programs can be easily done with the help of a related flowchart.

Proper Documentation: Flowchart provides better and proper documentation. It consists of various activities such as collecting, organizing, storing, and maintaining all related program records.

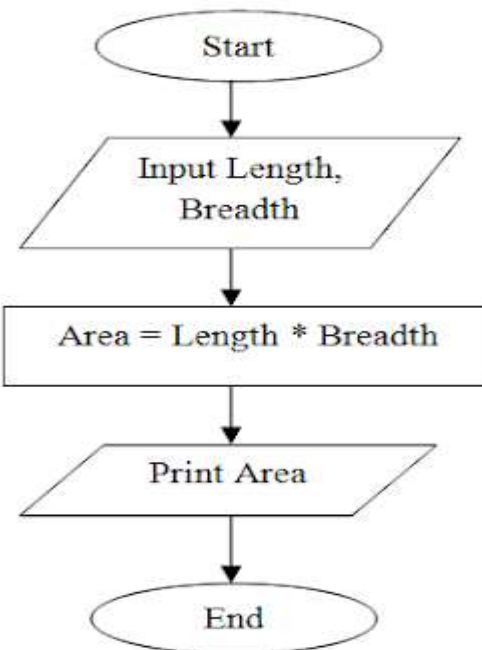
Testing: A flowchart helps in the testing process

### **Examples of flowchart**

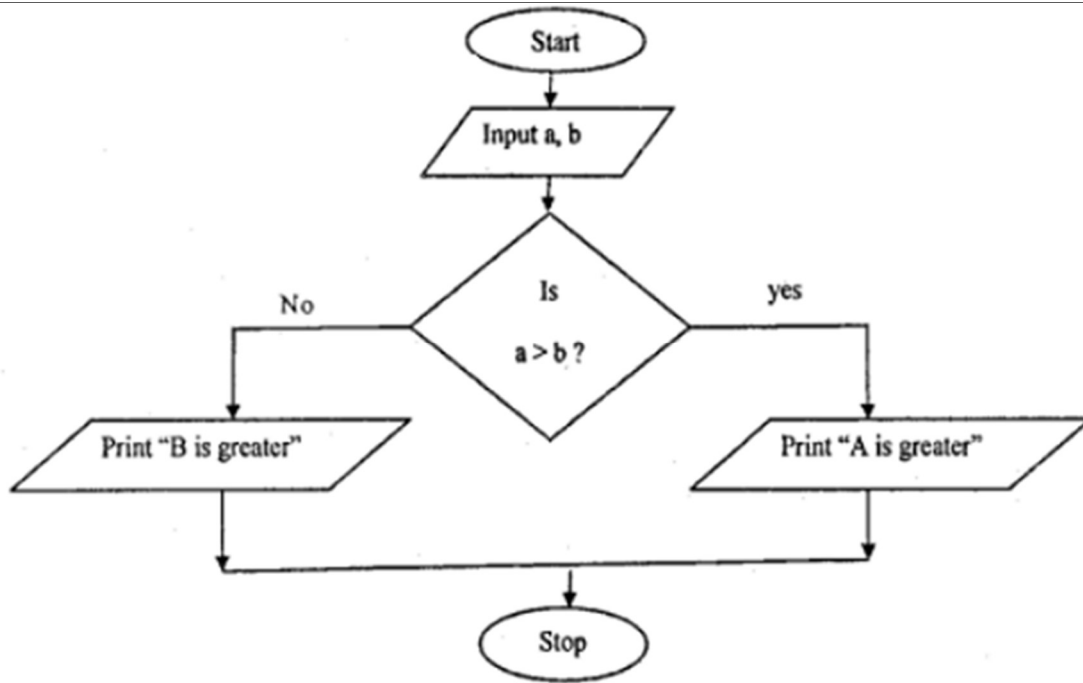
**Draw a flowchart for finding average of 3 Numbers.**



**Design a flowchart for calculating the area of a rectangle.**



**Design a flowchart to find out greater number from the two numbers.**



### Comparison of Algorithm and Flowchart

Algorithm	Flowchart
An algorithm is a procedure or set of rules that defines how a program is to be executed.	A flowchart is a graphical representation of the steps a program takes to process data.
An algorithm does not include any sort of geometrical pattern.	It utilizes different types of geometrical shapes, symbols, and patterns.
An algorithm demands the knowledge of a computer programming language.	A Flowchart doesn't demand the knowledge of a computer programming language.
It is difficult to debug the errors in algorithms.	It is easy to debug the errors in flowcharts.



---

## **Practical-1**

**AIM: Write an algorithm & draw a flowchart for following problems.**

**(a) Accepting two numbers and performing addition, subtraction, division and multiplication on them**

**(b) To check whether given number is even or odd.**

Solution: (Write your answer from here)







---

## **Practical-2**

**AIM: Write an algorithm & draw a flowchart for following problems.**

**(a) To print first 10 Fibonacci numbers**

**(b) To find out largest number from three numbers**

Solution: (Write your answer from here)







---


## **CO-2: Translate algorithm/flowchart into C program using correct syntax and execute it**

C is a programming language to communicate a set of instructions to the computer to perform some tasks. The set of instructions is known as a program. The program is written by a human being who is known as a programmer or a developer. The program is then compiled by a compiler to convert the C program into binary executable file. If the program is written properly as per the C programming language convention then the compiler will not generate any error, otherwise it shows all the errors with their location and suggestions to correct those. Here we will use the TURBO C or TURBO C++ compiler to perform our practical tasks. Once the program is compiled successfully, it gets converted into binary executable file known as .exe file. That executable file is known to computer and thus it is also referred as machine code. A binary executable file contains only '0' and '1' within it. The '0' indicates OFF and '1' indicates ON state. Based on the executable instructions in binary format the computer enables and disables its circuitry to perform various operations to fulfil the task mentioned in the program.

The above is just a brief overview of the entire life cycle of a program. To make it simple we the programmer or a developer are just concerned with writing proper C language code using editor of TURBO C compiler.

To start with this we require TURBO C/C++ compiler of 64bits version for Windows operating system of 64bits. Download the same from the link: <https://www.filehorse.com/download-turbo-c/>

Complete the installation and once the installation is done open the TURBO C++ compiler from

the desktop icon:  by double clicking it. The below window will be visible soon.



In above window observe things:

- 1) Dark blue area is the programming or coding section.
- 2) Top left light blue square is to close the coding section.
- 3) Top right light blue arrow is to minimize and maximise the code section.
- 4) The number near the light blue arrow at the top right corner is the code section window number. Which we can call/use by Alt+No. To toggle between open code section windows.
- 5) The name NONAME00.CPP is the default file name of our program which requires to be changed while saving the program.
- 6) The top bar is the menu bar which contains several menu options like, File, Edit, Search, Run, Compile, Debug, etc. These menu Item's name's first letter is of RED color which indicates the shortcut to open that menu item through Alt+Letter(having RED color).
- 7) The below status bar shows the short cuts to see the help thorough F1 key, F2 to Save, F3 to Open file, Alt+F9 to Compile the code and F9 to make and F10 for the menu.

Among all the above features of TURBO C compiler, to make our task simple and easy we are concerned with only the following options.

- |    |              |  |
|----|--------------|--|
| a) | File menu    | {New, Open, Save, Save as, close menu items} |
| b) | Compile menu | {Compile - Alt+F9}                           |
| c) | Run menu     | {Run - Ctrl+F9}                              |



---

A student should follow good habits to save his time, efforts and preserve his/her work.

- a) Always open a new code section, which does not have any past code.
- b) Write your own code on your own without depending upon others.
- c) Do save for even a minor change in program.
- d) Give a good name to the program and save the program to the appropriate location via the navigation window of the TURBO C compiler. Use save as an option to change either the file name or the location of the file.
- e) Once saved, then compile, use shortcut key and make a habit of it.
- f) Once compiled, do run, use a shortcut key and make a habit of it.
- g) See the output. If your output is correct, write down the program in your file pages or take a back up of your code either into pen drive, or online in gmail or g-drive.
- h) Once done, help your friends with their coding. This will help you in terms of revision and better idea of the programing.
- i) Once completed your lab work, SHUT DOWN the machine and SWITCH OFF the machine, lights and fans not in use. Make an entry into LAB master and leave quietly.

### Structure of C Program

The basic structure of a C program is divided into 6 parts which makes it easy to read, modify, document, and understand in a particular format. C program must follow the below-mentioned outline in order to successfully compile and execute.

Documentation

Preprocessor Section

Definition

Global Declaration

Main() Function

User defined Functions

#### 1. Documentation

This section consists of the description of the program, the name of the program, and the creation date and time of the program. It is specified at the start of the program in the form of comments. Documentation can be represented as:

// description, name of the program, programmer name, date, time etc.

#### 2. Preprocessor Section

All the header files of the program will be declared in the preprocessor section of the program. Header files help us to access other's improved code into our code. A copy of these multiple files is inserted into our program before the process of compilation.



---

Example:

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

```
#include<math.h>
```

### 3. Definition

The define section comprises of different constants declared using the define keyword. It is given by:

```
#define a = 2
```

### 4. Global Declaration

The global declaration section contains global variables, function declaration, and static variables. Variables and functions which are declared in this scope can be used anywhere in the program.

### 5. Main Function

Every C program must have a main function. The main() function of the program is written in this section. Operations like declaration and execution are performed inside the curly braces of the main program. The return type of the main() function can be int as well as void too. void() main tells the compiler that the program will not return any value. The int main() tells the compiler that the program will return an integer value.

Example:

```
void main() or int main()
```

### 6. User Defined Functions

The user defined functions specified the functions specified as per the requirements of the user. For example, color(), sum(), division(), etc.

**Example: To find the sum of two numbers given by the user.**

```
/* Sum of two numbers */
```

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

```
int main()
```

```
{
```

```
int a, b, sum;
```

```
printf("Enter two numbers to be added ");
```

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

```
// calculating sum
```



```
sum = a + b;  
  
printf("%d + %d = %d", a, b, sum);  
  
return 0; // return the integer value in the sum  
  
}
```

### Detail Description of C Program

/* Sum of the two numbers */	It is the comment section. Any statement described in it is not considered as a code. It is a part of the description section in a code. The comment line is optional. It can be in a separate line or part of an executable line.
#include<stdio.h>	It is the standard input-output header file. It is a command of the preprocessor section.
int main()	main() is the first function to be executed in every program. We have used int with the main() in order to return an integer value.
{... }	The curly braces mark the beginning and end of a function. It is mandatory in all the functions.
printf()	The printf() prints text on the screen. It is a function for displaying constant or variables data. Here, 'Enter two numbers to be added' is the parameter passed to it.
scanf()	It reads data from the standard input stream and writes the result into the specified arguments.
sum = a + b	The addition of the specified two numbers will be passed to the sum parameter in the output.
return 0	A program can also run without a return 0 function. It simply states that a program is free from error and can be successfully exited.



---

### **Practical-3**

**AIM:** Write a C program for following problems.

(a) Write a program to that performs as calculator (addition, multiplication, division, subtraction).

(b) Write a program to find area of triangle ( $a = h * b * .5$ ) a = area h = height  
b = base

Solution: (Write your answer from here)





---

### **Practical-4**

**AIM: Write a C program for following problems.**

- (a) Write a C program to interchange two numbers.**
- (b) Write a program to compute Fahrenheit from centigrade ( $f=1.8*c + 32$ )**

Solution: (Write your answer from here)







---

## **CO-3: Write programs using conditional, branching, iteration, and recursion.**

### **Conditional Statement in C**

Conditional Statements in C programming are used to make decisions based on the conditions. Conditional statements execute sequentially when there is no condition around the statements. If you put some condition for a block of statements, the execution flow may change based on the result evaluated by the condition. This process is called decision making in 'C.'

### **IF Statement**

It is one of the powerful conditional statement. If statement is responsible for modifying the flow of execution of a program. If statement is always used with a condition. The condition is evaluated first before executing any statement inside the body of If. The syntax for if statement is as follows:  
if (condition)

instruction;

The condition evaluates to either true or false. True is always a non-zero value, and false is a value that contains zero. Instructions can be a single instruction or a code block enclosed by curly braces { }.

Following program illustrates the use of if construct in 'C' programming:

```
#include<stdio.h>
int main()
{
    int num1=1;
    int num2=2;
    if(num1<num2)           //test-condition
    {
        printf("num1 is smaller than num2");
    }
    return 0;
}
```

Output:

num1 is smaller than num2

Description of above program:

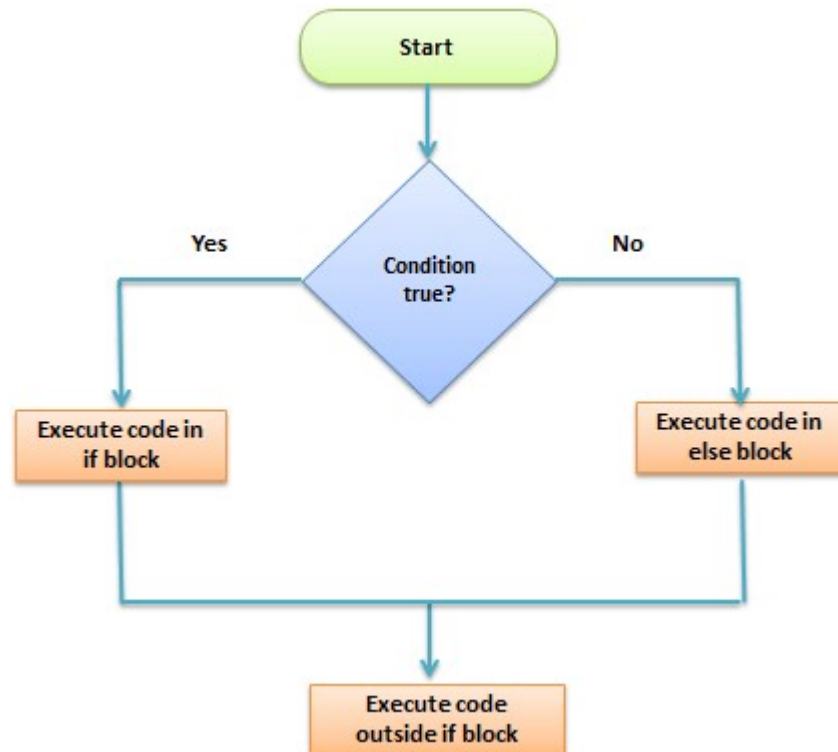
1. In the above program, we have initialized two variables with num1, num2 with value as 1, 2 respectively.



2. then, we have used if with a test-expression to check which number is the smallest and which number is the largest. We have used a relational expression in if construct. Since the value of num1 is smaller than num2, the condition will evaluate to true.

3. Thus it will print the statement inside the block of If. After that, the control will go outside of the block and program will be terminated with a successful result.

### IF-ELSE Statement:



The if-else is statement is an extended version of If. The general form of if-else is as follows:

if (test-expression)

{

True block of statements

}

Else

{

False block of statements

}

Statements;



In this type of a construct, if the value of test-expression is true, then the true block of statements will be executed. If the value of test-expression is false, then the false block of statements will be executed. In any case, after the execution, the control will be automatically transferred to the statements appearing outside the block of If.

Following programs illustrate the use of the if-else construct:

We will initialize a variable with some value and write a program to determine if the value is less than ten or greater than ten.

```
#include<stdio.h>

int main()
{
    int num=19;
    if(num<10)
    {
        printf("The value is less than 10");
    }
    else
    {
        printf("The value is greater than 10");
    }
    return 0;
}
```

Output:

The value is greater than 10

Description of above program:

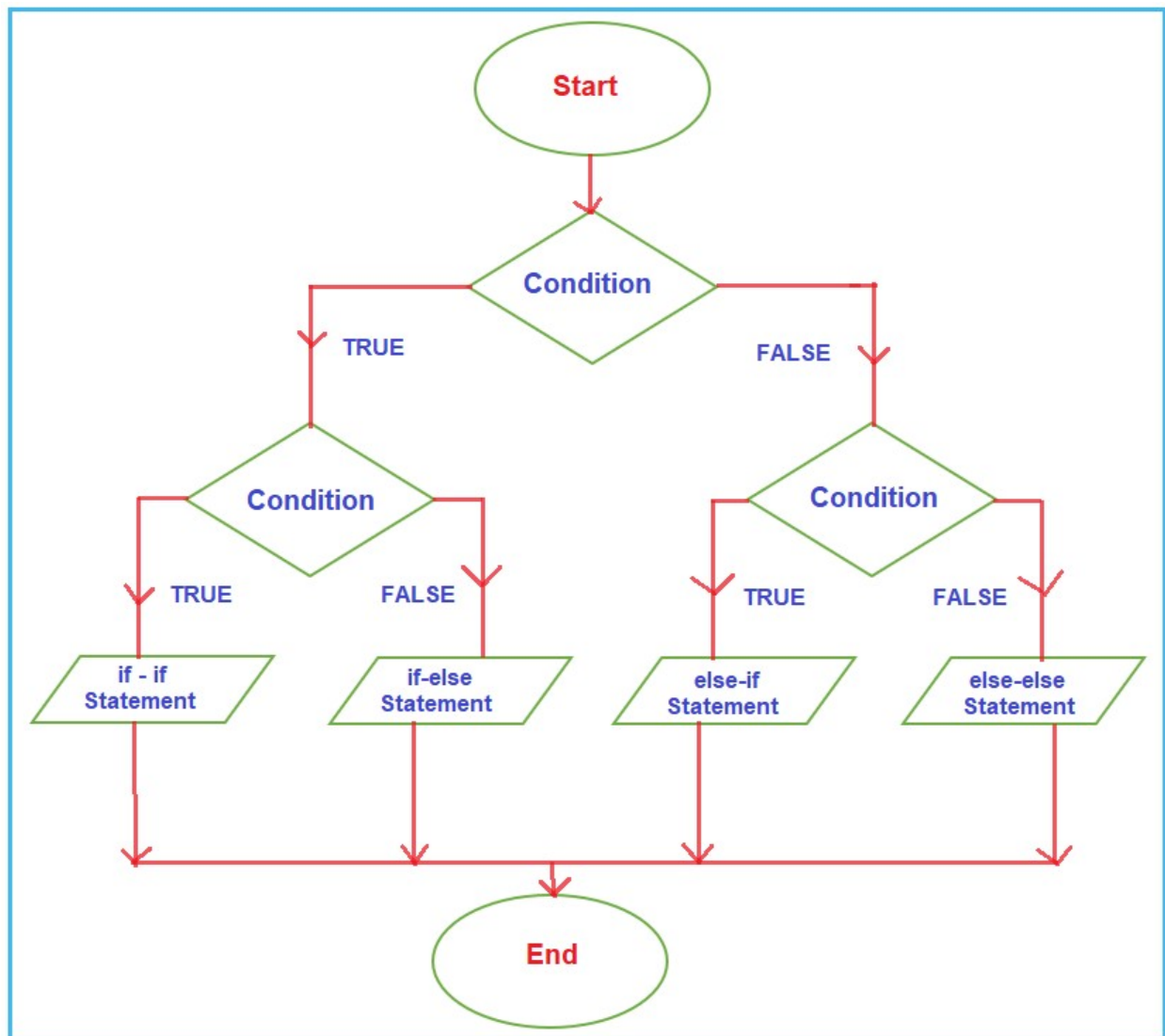
1. We have initialized a variable with value 19. We have to find out whether the number is bigger or smaller than 10 using a 'C' program. To do this, we have used the if-else construct.
2. Here we have provided a condition `num<10` because we have to compare our value with 10.
3. As you can see the first block is always a true block which means, if the value of test-expression is true then the first block which is If, will be executed.



4. The second block is an else block. This block contains the statements which will be executed if the value of the test-expression becomes false. In our program, the value of num is greater than ten hence the test-condition becomes false and else block is executed. Thus, our output will be from an else block which is “The value is greater than 10”. After the if-else, the program will terminate with a successful result.

### **Nested IF-ELSE Statements:**

When a series of decision is required, nested if-else is used. Nesting means using one if-else construct within another one.



Let's write a program to illustrate the use of nested if-else.

The below program checks if a number is less or greater than 10 and prints the result using nested if-else construct.



---

```
#include<stdio.h>

int main()
{
    int num=1;
    if(num<10)
    {
        if(num==1)
        {
            printf("The value is:%d\n",num);
        }
        else
        {
            printf("The value is greater than 1");
        }
    }
    else
    {
        printf("The value is greater than 10");
    }
    return 0;
}
```

Output:

The value is:1

Description of above program:

1. Firstly, we have declared a variable num with value as 1. Then we have used if-else construct.
2. In the outer if-else, the condition provided checks if a number is less than 10. If the condition is true then and only then it will execute the inner loop. In this case, the condition is true hence the inner block is processed.



3. In the inner block, we again have a condition that checks if our variable contains the value 1 or not. When a condition is true, then it will process the If block otherwise it will process an else block. In this case, the condition is true hence the If a block is executed and the value is printed on the output screen.
4. The above program will print the value of a variable and exit with success.

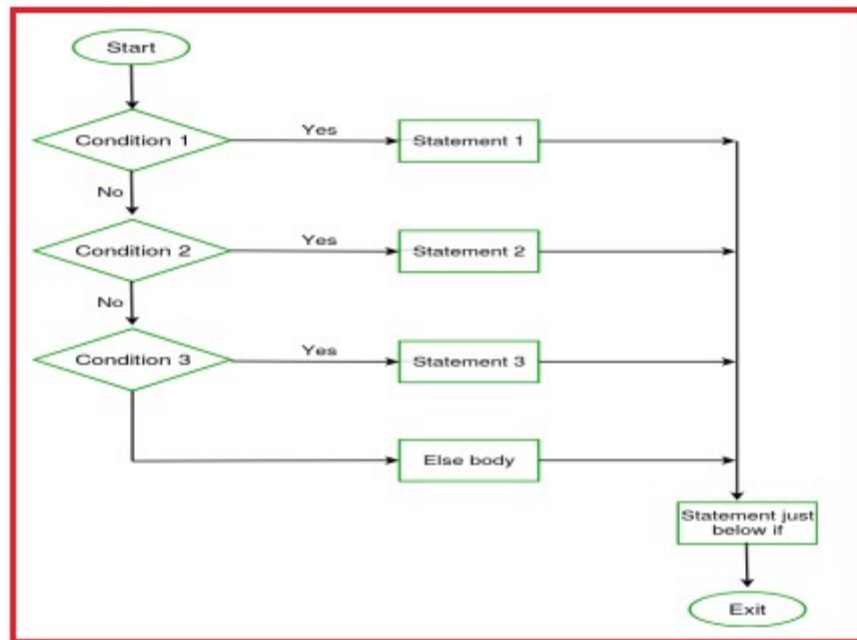
### **Nested Else-if statements (else-if ladders)**

Nested else-if is used when multipath decisions are required.

The general syntax of how else-if ladders are constructed in 'C' programming is as follows:

```
if (test - expression 1)
{
    statement1;
}
else if (test - expression 2)
{
    Statement2;
}
else if (test - expression 3)
{
    Statement3;
}
else if (test - expression n)
{
    Statement n;
}
else
{
    default;
}
Statement x;
```

This type of structure is known as the else-if ladder. This chain generally looks like a ladder hence it is also called as an else-if ladder. The test-expressions are evaluated from top to bottom. Whenever a true test-expression is found, statement associated with it is executed. When all the test-expressions become false, then the default else statement is executed.



Let us see the actual working with the help of a program.

The below program prints the grade as per the marks scored in a test. We have used the else-if ladder construct in the below program.

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

```
int main()
```

```
{
```

```
    int marks=83;
```

```
    if(marks>75)
```

```
    {
```

```
        printf("First class");
```

```
    }
```

```
    else if(marks>65)
```

```
    {
```

```
        printf("Second class");
```

```
    }
```

```
    else if(marks>55)
```

```
    {
```





---

```
        printf("Third class");  
    }  
    Else  
    {  
        printf("Fourth class");  
    }  
    return 0;  
}
```

Output:

First class

Description of above program:

1. We have initialized a variable with marks. In the else-if ladder structure, we have provided various conditions.
2. The value from the variable marks will be compared with the first condition since it is true the statement associated with it will be printed on the output screen.
3. If the first test condition turns out false, then it is compared with the second condition.
4. This process will go on until the all expression is evaluated otherwise control will go out of the else-if ladder, and default statement will be printed.



---

### **Practical-5**

**AIM:** Write a C Program for following problems using branching, iteration and recursion.

- (a)** Write a program to read marks of a student from keyboard whether the student is pass or fail (using if else)
- (b)** Write a program to read three numbers from keyboard and find out maximum out of these three. (Nested if else)

Solution: (Write your answer from here)





---

## Switch Statement:

The switch statement allows us to execute one code block among many alternatives. You can do the same thing with the if...else..if ladder. However, the syntax of the switch statement is much easier to read and write.

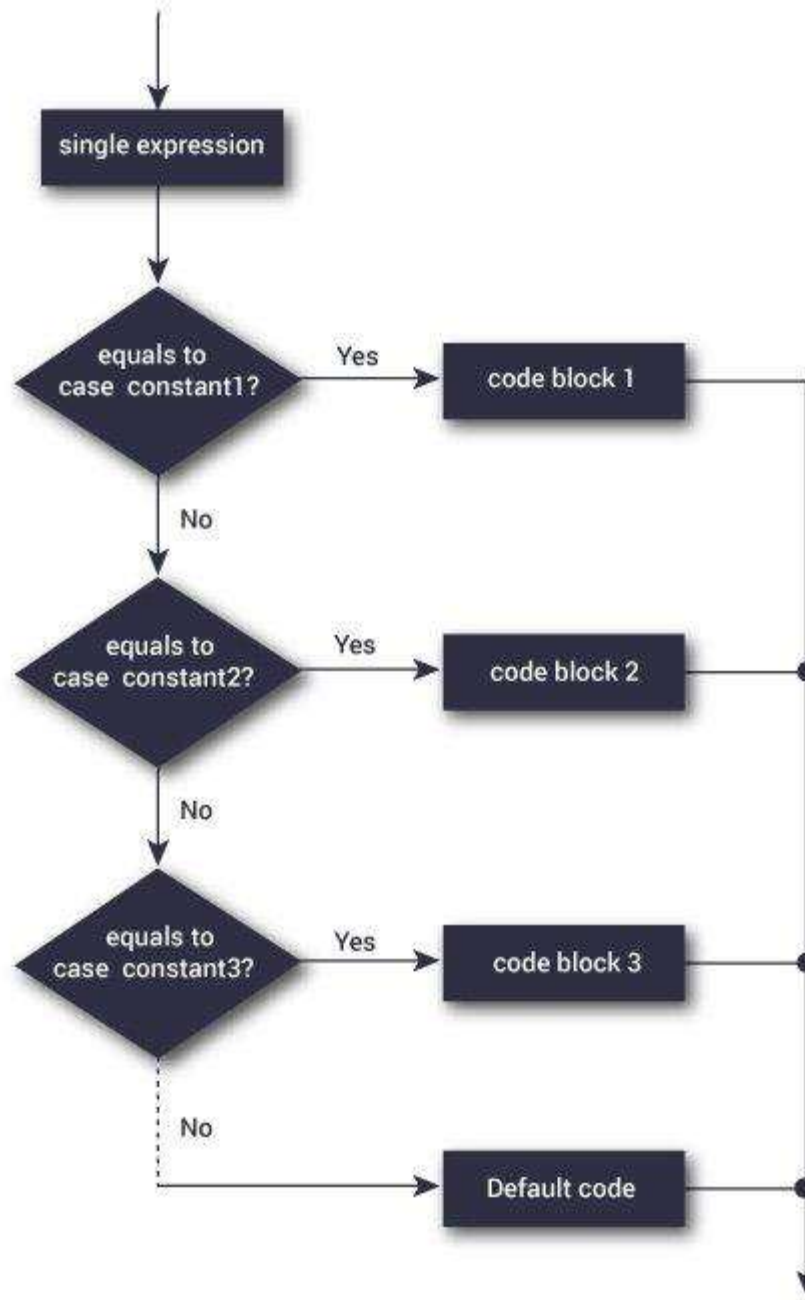
### Syntax of Switch...case is as follows:

```
switch (expression)
{
    case constant1:
        // statements
        break;

    case constant2:
        // statements
        break;
    .
    .
    .
    default:
        // default statements
}
```

### How does the switch statement work?

The expression is evaluated once and compared with the values of each case label. If there is a match, the corresponding statements after the matching label are executed. For example, if the value of the expression is equal to constant2, statements after case constant2: are executed until break is encountered. If there is no match, the default statements are executed.



## Loop Statements

Loops in programming are used to repeat a block of code until the specified condition is met. A loop statement allows programmers to execute a statement or group of statements multiple times without repetition of code.

There are mainly two types of loops in C Programming:



**Entry Controlled loops:** In Entry controlled loops the test condition is checked before entering the main body of the loop. For Loop and While Loop is Entry-controlled loops.

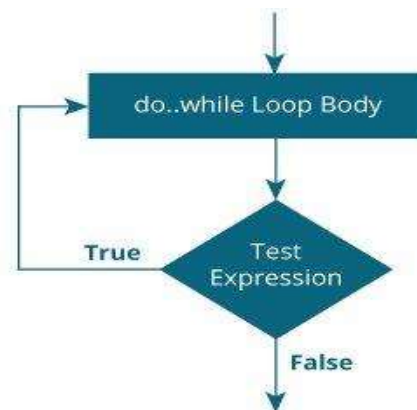
**Exit Controlled loops:** In Exit controlled loops the test condition is evaluated at the end of the loop body. The loop body will execute at least once, irrespective of whether the condition is true or false. do-while Loop is Exit Controlled loop.

### **do-while loop in C**

The do-while loop continues until a given condition satisfies. It is also called post tested loop. It is used when it is necessary to execute the loop at least once (mostly menu driven programs).

The syntax of do-while loop in c language is given below:

```
do {  
    //code to be executed  
} while (condition);
```

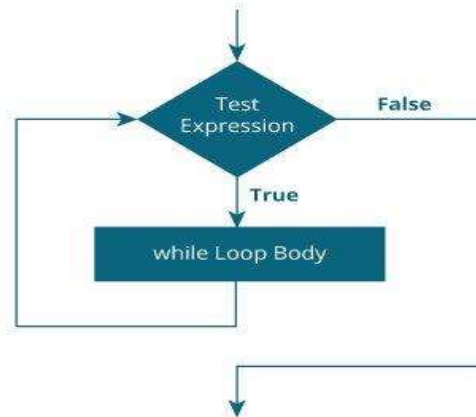


### **while loop in C**

The while loop in c is to be used in the scenario where we don't know the number of iterations in advance. The block of statements is executed in the while loop until the condition specified in the while loop is satisfied. It is also called a pre-tested loop.

The syntax of while loop in c language is given below:

```
while(condition){  
    //code to be executed  
}
```



## For Loop in C

The for loop in C language is used to iterate the statements or a part of the program several times. It is frequently used to traverse the data structures like the array and linked list.

Syntax:

for (initialize expression; test expression; update expression)

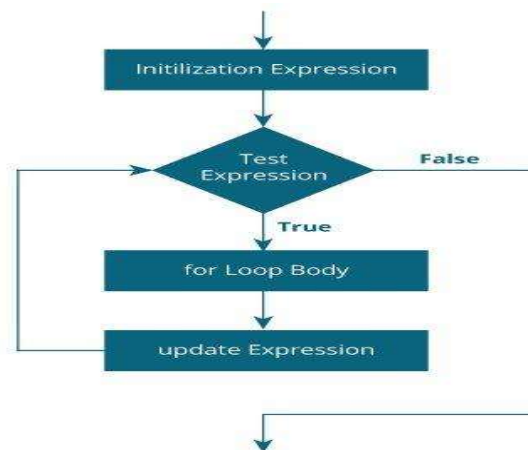
{

//

// body of for loop

//

}





---

## **Practical-6**

**AIM:** Write a C Program for following problems using branching, iteration and recursion.

(a) Write a C program to read no 1 to 7 and print relatively day Sunday to Saturday. (Switch statement)

(b) Write a C program to find factorial of a given number.

(c) Write a program to print following patterns

i) *	ii) *	iii) * * *
* *	* *	* *
* * *	* * *	*

Solution: (Write your answer from here)







---

## **CO-4: Decompose a problem into function.**

### **Functions in C**

In C, we can divide a large program into the basic building blocks known as function. The function contains the set of programming statements enclosed by {}. A function can be called multiple times to provide reusability and modularity to the C program. In other words, we can say that the collection of functions creates a program. The function is also known as procedure or subroutine in other programming languages.

### **Advantage of functions in C**

There are the following advantages of C functions.

- By using functions, we can avoid rewriting same logic/code again and again in a program.
- We can call C functions any number of times in a program and from any place in a program.
- We can track a large C program easily when it is divided into multiple functions.
- Reusability is the main achievement of C functions.
- However, Function calling is always a overhead in a C program.

### **Function Aspects**

There are three aspects of a C function.

**Function declaration** A function must be declared globally in a c program to tell the compiler about the function name, function parameters, and return type.

**Function call** Function can be called from anywhere in the program. The parameter list must not differ in function calling and function declaration. We must pass the same number of functions as it is declared in the function declaration.

**Function definition** It contains the actual statements which are to be executed. It is the most important aspect to which the control comes when the function is called. Here, we must notice that only one value can be returned from the function.

### **Types of Functions**

There are two types of functions in C programming:

**Library Functions:** are the functions which are declared in the C header files such as scanf(), printf(), gets(), puts(), ceil(), floor() etc.



---

**User-defined functions:** are the functions which are created by the C programmer, so that he/she can use it many times. It reduces the complexity of a big program and optimizes the code.

### **Return Value**

A C function may or may not return a value from the function. If you don't have to return any value from the function, use void for the return type.

Let's see a simple example of C function that doesn't return any value from the function.

#### **Example without return value:**

```
void hello(){  
    printf("hello c");  
}
```

If you want to return any value from the function, you need to use any data type such as int, long, char, etc. The return type depends on the value to be returned from the function.

Let's see a simple example of C function that returns int value from the function.

#### **Example with return value:**

```
int get(){  
    return 10;  
}
```

In the above example, we have to return 10 as a value, so the return type is int. If you want to return floating-point value (e.g., 10.2, 3.1, 54.5, etc), you need to use float as the return type of the method.

### **Types of functions in terms of argument and return value:**

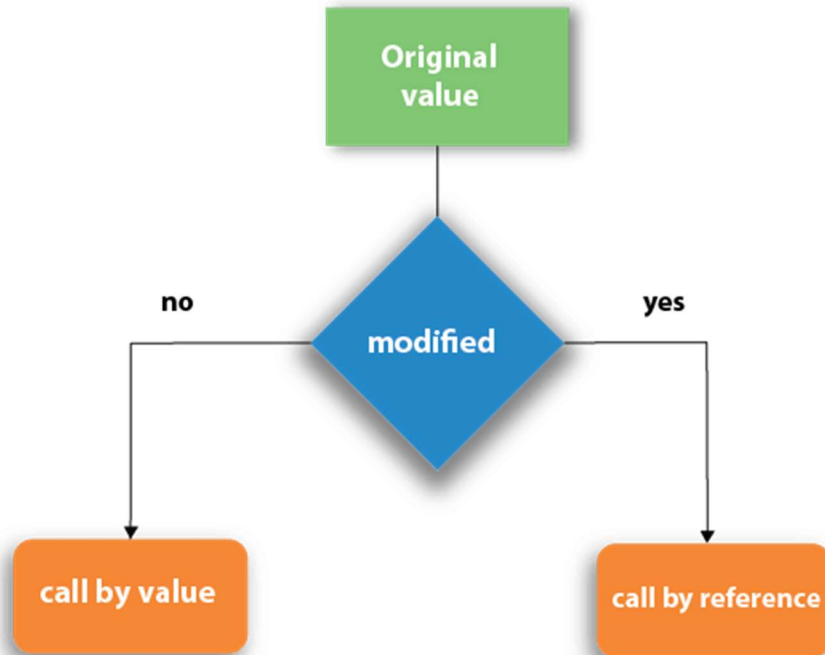
A function may or may not accept any argument. It may or may not return any value. Based on these facts, There are four different aspects of function calls.

- function without arguments and without return value
- function without arguments and with return value
- function with arguments and without return value
- function with arguments and with return value

### **Call by value and Call by reference in C**



There are two methods to pass the data into the function in C language, i.e., call by value and call by reference.



### Call by value and Call by reference in C

There are two methods to pass the data into the function in C language, i.e., call by value and call by reference.

#### Call by value in C

- In call by value method, the value of the actual parameters is copied into the formal parameters. In other words, we can say that the value of the variable is used in the function call in the call by value method.
- In call by value method, we can not modify the value of the actual parameter by the formal parameter.
- In call by value, different memory is allocated for actual and formal parameters since the value of the actual parameter is copied into the formal parameter.
- The actual parameter is the argument which is used in the function call whereas formal parameter is the argument which is used in the function definition.

#### Call by reference in C



- 
- In call by reference, the address of the variable is passed into the function call as the actual parameter.
  - The value of the actual parameters can be modified by changing the formal parameters since the address of the actual parameters is passed.
  - In call by reference, the memory allocation is similar for both formal parameters and actual parameters. All the operations in the function are performed on the value stored at the address of the actual parameters, and the modified value gets stored at the same address.



---

### **Practical-7**

**AIM: Write a C Program using concept of function.**

- (a) Write a program that defines a function to add first n numbers.**
- (b) Write a program using function to find maximum number from two numbers.**

Solution: (Write your answer from here)





---

## **Practical-8**

**AIM: Write a C Program using concept of function.**

- (a) Write a program to exchange two numbers by using function**
- (b) Write a program to reverse the number using function.**

Solution: (Write your answer from here)







---

## **CO-5: Develop an application using the concepts of array, pointer, structure, and file management to solve engineering and/or scientific problems**

### **Arrays in C**

Arrays are used to store multiple values in a single variable, instead of declaring separate variables for each value. To create an array, define the data type (like int) and specify the name of the array followed by square brackets []. To insert values to it, use a comma-separated list, inside curly braces:

```
int myNumbers[] = {25, 50, 75, 100};
```

We have now created a variable that holds an array of four integers.

### **Access the Elements of an Array**

To access an array element, refer to its index number. Array indexes start with 0: [0] is the first element. [1] is the second element, etc.

This statement accesses the value of the first element [0] in myNumbers:

Example

```
int myNumbers[] = {25, 50, 75, 100};
```

```
printf("%d", myNumbers[0]);
```

```
// Outputs 25
```

### **Advantage of C Array**

- 1) Code Optimization: Less code to access the data.
- 2) Ease of traversing: By using the for loop, we can retrieve the elements of an array easily.
- 3) Ease of sorting: To sort the elements of the array, we need a few lines of code only.
- 4) Random Access: We can access any element randomly using the array.



---

## Disadvantage of C Array

- 1) **Fixed Size:** Whatever size, we define at the time of declaration of the array, we can't exceed the limit. So, it doesn't grow the size dynamically like LinkedList which we will learn later.

## Two Dimensional Array in C

The two-dimensional array can be defined as an array of arrays. The 2D array is organized as matrices which can be represented as the collection of rows and columns. However, 2D arrays are created to implement a relational database lookalike data structure. It provides ease of holding the bulk of data at once which can be passed to any number of functions wherever required.

### Declaration of two dimensional Array in C

The syntax to declare the 2D array is given below.

```
data_type array_name[rows][columns];
```

## C Pointers

The pointer in C language is a variable which stores the address of another variable. This variable can be of type int, char, array, function, or any other pointer. The size of the pointer depends on the architecture. However, in 32-bit architecture the size of a pointer is 2 byte. Consider the following example to define a pointer which stores the address of an integer.

```
int n = 10;
```

```
int* p = &n;
```

### Declaring a pointer

The pointer in c language can be declared using \* (asterisk symbol). It is also known as indirection pointer used to dereference a pointer.

```
int *a;//pointer to int
```

```
char *c;//pointer to char
```

### Pointer to array

```
int arr[10];
```

```
int *p[10]=&arr;
```

### Pointer to a function

```
void show (int);
```



---

```
void(*p)(int) = &display;
```

### **Pointer to structure**

```
struct st {  
    int i;  
    float f;  
}ref;  
struct st *p = &ref;
```

### **Advantage of pointer**

- 1) Pointer reduces the code and improves the performance, it is used to retrieving strings, trees, etc. and used with arrays, structures, and functions.
- 2) We can return multiple values from a function using the pointer.
- 3) It makes you able to access any memory location in the computer's memory.

### **Usage of pointer**

There are many applications of pointers in c language.

#### **1) Dynamic memory allocation**

In c language, we can dynamically allocate memory using malloc() and calloc() functions where the pointer is used.

#### **2) Arrays, Functions, and Structures**

Pointers in c language are widely used in arrays, functions, and structures. It reduces the code and improves the performance.

#### **Address Of (&) Operator**

The address of operator '&' returns the address of a variable. But, we need to use %u to display the address of a variable.

### **Why use structure?**

In C, there are cases where we need to store multiple attributes of an entity. It is not necessary that an entity has all the information of one type only. It can have different attributes of different data



types. For example, an entity Student may have its name (string), roll number (int), marks (float). To store such type of information regarding an entity student, we have the following approaches:

Construct individual arrays for storing names, roll numbers, and marks.

Use a special data structure to store the collection of different data types.

## What is Structure

Structure in c is a user-defined data type that enables us to store the collection of different data types. Each element of a structure is called a member. Structures can simulate the use of classes and templates as it can store various information. The `struct` keyword is used to define the structure. Let's see the syntax to define the structure in c.

```
struct structure_name
{
    data_type member1;
    data_type member2;
    .
    .
    data_type memberN;
};
```

Let's see the example to define a structure for an entity employee in c.

```
struct employee
{
    int id;
    char name[20];
    float salary;
};
```

## Declaring structure variable

We can declare a variable for the structure so that we can access the member of the structure easily. There are two ways to declare structure variable:

By `struct` keyword within `main()` function



---

By declaring a variable at the time of defining the structure.

### 1st way:

Let's see the example to declare the structure variable by struct keyword. It should be declared within the main function.

```
struct employee
```

```
{  int id;

    char name[50];

    float salary;

};
```

Now write given code inside the main() function.

```
struct employee e1, e2;
```

The variables e1 and e2 can be used to access the values stored in the structure. Here, e1 and e2 can be treated in the same way as the objects in C++ and Java.

### 2nd way:

Let's see another way to declare variable at the time of defining the structure.

```
struct employee
```

```
{

    int id;

    char name[50];

    float salary;

}e1,e2;
```

### Accessing members of the structure

There are two ways to access structure members:

- 1) By . (member or dot operator)
- 2) By -> (structure pointer operator)

Let's see the code to access the id member of p1 variable by. (member) operator.



---

p1.id

### **C Structure example**

Let's see a simple example of structure in C language.

```
#include<stdio.h>

#include <string.h>

struct employee

{   int id;

    char name[50];

}e1; //declaring e1 variable for structure

int main( )

{

    //store first employee information

    e1.id=101;

    strcpy(e1.name, "Sonoo Jaiswal");//copying string into char array

    //printing first employee information

    printf( "employee 1 id : %d\n", e1.id);

    printf( "employee 1 name : %s\n", e1.name);

    return 0;

}
```

### **File Handling in C**

In programming, we may require some specific input data to be generated several numbers of times. Sometimes, it is not enough to only display the data on the console. The data to be displayed may be very large, and only a limited amount of data can be displayed on the console, and since the memory is volatile, it is impossible to recover the programmatically generated data again and again. However, if we need to do so, we may store it onto the local file system which is volatile and can be accessed every time. Here, comes the need of file handling in C.



File handling in C enables us to create, update, read, and delete the files stored on the local file system through our C program. The following operations can be performed on a file.

- Creation of the new file
- Opening an existing file
- Reading from the file
- Writing to the file
- Deleting the file

### Functions for file handling

There are many functions in the C library to open, read, write, search and close the file. A list of file functions are given below:

No.	Function	Description
1	fopen()	opens new or existing file
2	fprintf()	write data into the file
3	fscanf()	reads data from the file
4	fputc()	writes a character into the file
5	fgetc()	reads a character from file
6	fclose()	closes the file
7	fseek()	sets the file pointer to given position
8	fputw()	writes an integer to file
9	fgetw()	reads an integer from file
10	ftell()	returns current position





11	rewind()	sets the file pointer to the beginning of the file
----	----------	--

### Opening File: fopen()

We must open a file before it can be read, write, or update. The fopen() function is used to open a file. The syntax of the fopen() is given below.

```
FILE *fopen( const char * filename, const char * mode );
```

The fopen() function accepts two parameters:

The file name (string). If the file is stored at some specific location, then we must mention the path at which the file is stored. For example, a file name can be like "c://some\_folder/some\_file.ext".

The mode in which the file is to be opened. It is a string. We can use one of the following modes in the fopen() function.

Mode	Description
R	opens a text file in read mode
w	opens a text file in write mode
A	opens a text file in append mode
r+	opens a text file in read and write mode
w+	opens a text file in read and write mode
a+	opens a text file in read and write mode
rb	opens a binary file in read mode
wb	opens a binary file in write mode
ab	opens a binary file in append mode
rb+	opens a binary file in read and write mode
wb+	opens a binary file in read and write mode
ab+	opens a binary file in read and write mode



---

### **Practical-9**

**AIM:** Write a C Program using the concept of an array.

- (a)** Write a C program to find out the Maximum and Minimum number from given 10 numbers in an array.
- (b)** Write a program to sort the elements of an array in ascending order.

Solution: (Write your answer from here)





---

## **Practical-10**

**AIM:** Write a C Program using the concept of pointer, structure & File.

(a) Write a program to access elements using pointer.

(b) Design a structure `student_record` to contain name, branch and total marks obtained. Develop a program to read data for 10 students in a class and print them.

(c) A file named `data` contains series of integer numbers. Write a c program to read all numbers from file and then write all odd numbers into file named “odd” and write all even numbers into file named “even”. Display all the contents of these file on screen

Solution: (Write your answer from here)



