

C++ Basic Coding

BS Computer Science

Submitted by:

**Nayaab Abdul Khaliq**

Submitted to:

**Ms. Sardar-un-Nisa**

# Introduction to C++:

## C++ is a general-purpose programming language that is widely used for developing software applications. It is an extension of the C programming language and supports object-oriented programming. The following are some of the essential components of a C++ program:

## Essentials of C++ Program:

1. **Comments**: Comments are used to add notes to the code that explain what it does. They are not executed by the compiler and do not affect the program’s output.

* Single-line comments: // This is a single-line comment
* Multi-line comments: /\* This is a multi-line comment \*/

1. **Preprocessor Directives**: Preprocessor directives are used to include header files in the program. They are executed before the compilation process and help in defining constants, macros, and conditional compilation.

**Example: #define, #ifdef, #ifndef, #endif.**

1. **Namespaces**: Namespaces are used to avoid naming conflicts between different parts of a program. They help in organizing code and make it easier to read and maintain.
2. **Variables**: Variables are used to store data in a program. They have a name, a data type, and a value.
3. **Data Types**: Data types define the type of data that can be stored in a variable. C++ supports several built-in data types such as int, float, double, char, etc.

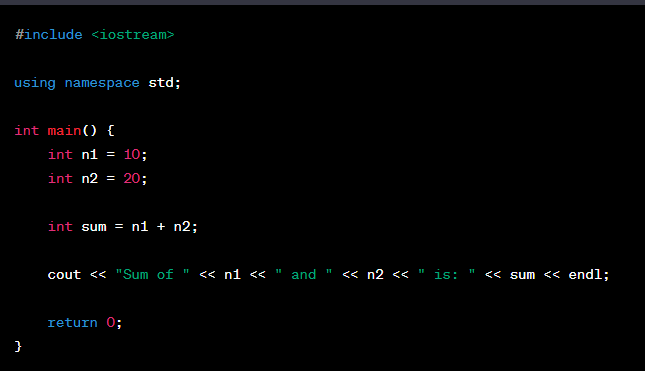
* **int**: Integer data type.
* **float**: Floating-point data type.
* **double**: Double-precision floating-point data type.
* **char**: Character data type.
* **bool**: Boolean data type.
* **void**: Represents no data type (used for functions)

1. **Operators**: Operators are used to perform operations on variables and values. C++ supports several types of operators such as arithmetic, relational, logical, bitwise, etc.
2. **Control Structures**: Control structures are used to control the flow of execution in a program. They include if-else statements, loops, switch statements, etc.
3. **Functions**: Functions are used to group related code together and make it reusable. They have a name, a return type (optional), and parameters (optional).
4. **Classes and Objects**: Classes and objects are used to implement object-oriented programming in C++. A class is a blueprint for creating objects, while an object is an instance of a class.

# Findings:

## General Syntax of C++ Program:

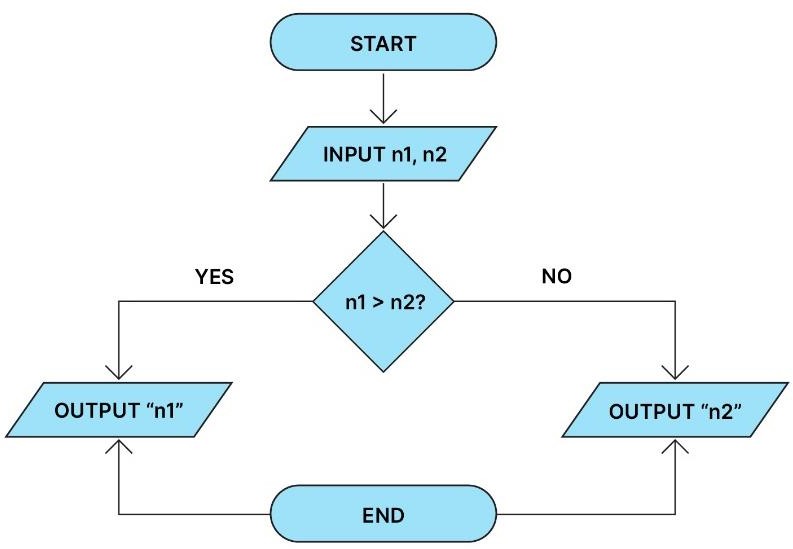
Here is an example of simple C++ program:



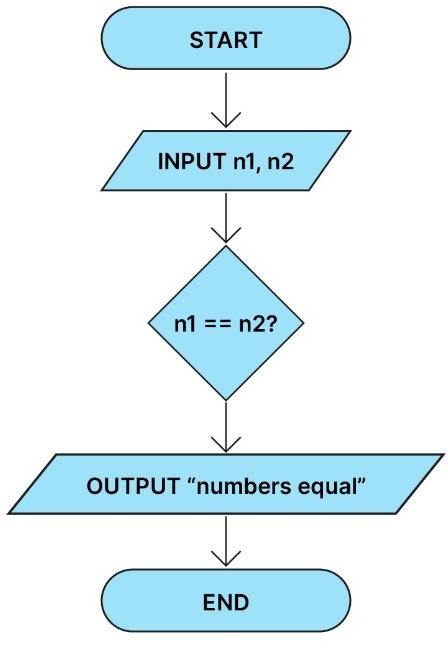
## Tasks:

1. Do the task 2 but take the product instead of sum and show it on the screen.

### PROGRAM # 1 (Coding)

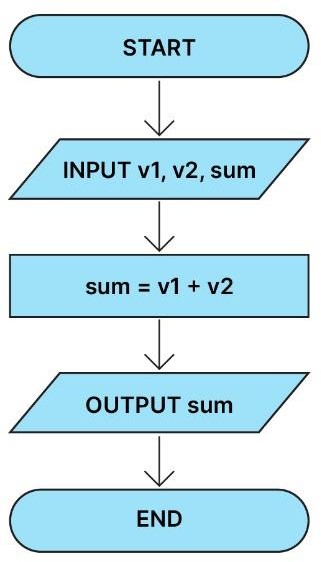


**FLOWCHART DIAGRAM # 01 (A)**



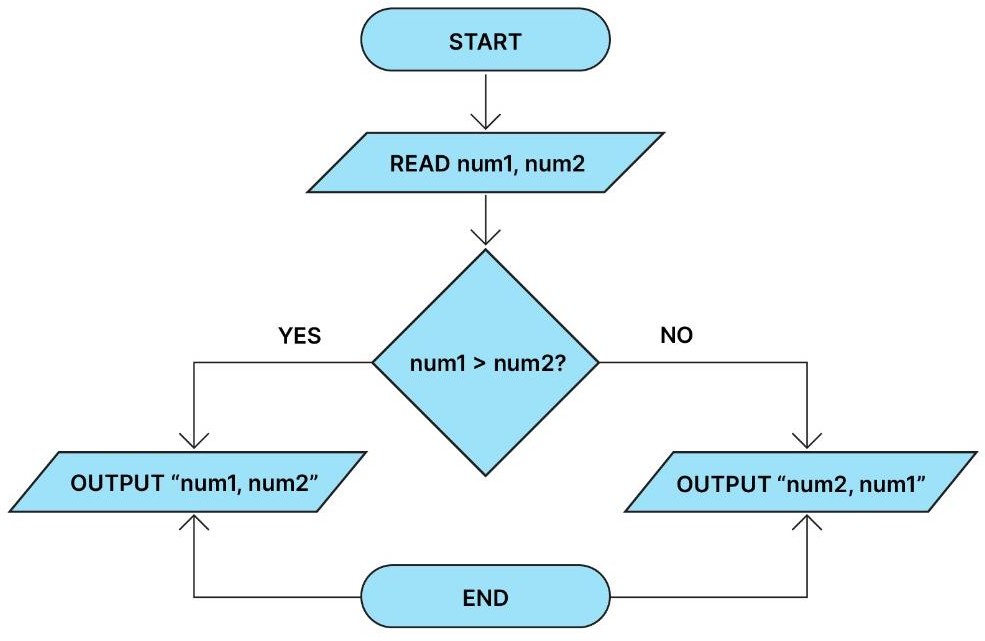
1. Draw a flowchart to take two values from the user as input and display their sum.

### FLOWCHART DIAGRAM # 02



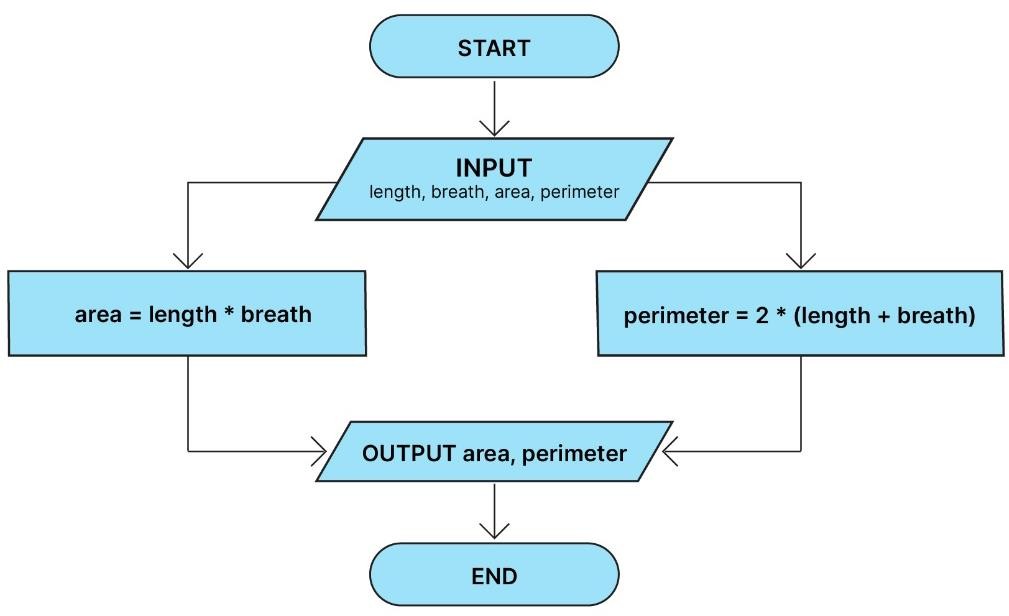
1. Draw a flowchart for a program that reads two numbers and displays the numbers read in decreasing order.

### FLOWCHART DIAGRAM # 03

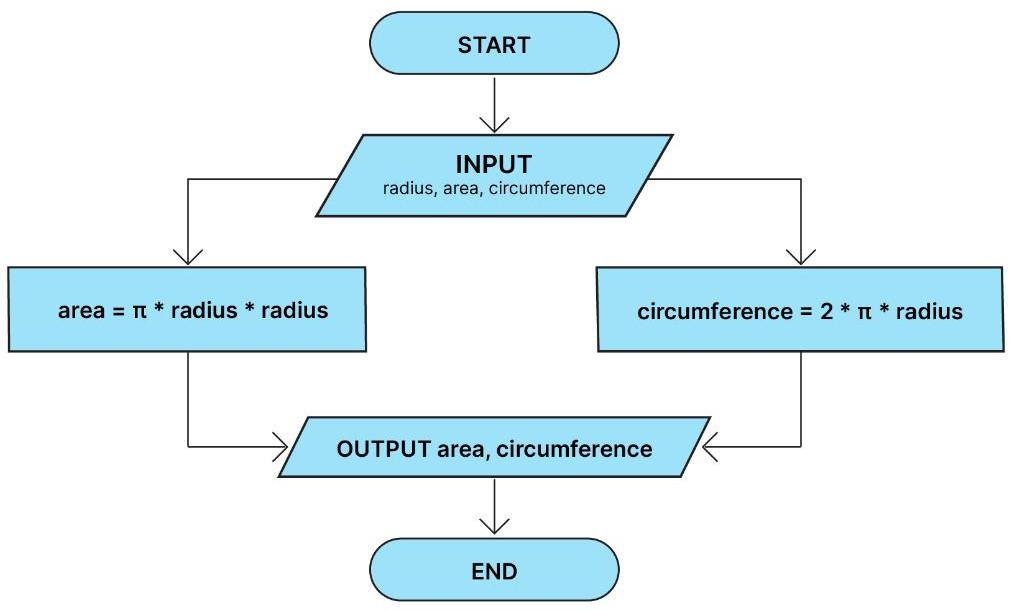


1. The lengths and breadth of a rectangle and the radius of a circle are input through the keyboard. Draw a flow chart to calculate:
   1. The area & perimeter of the rectangle
   2. The area & circumference of the circle

### FLOWCHART DIAGRAM # 04 (a)



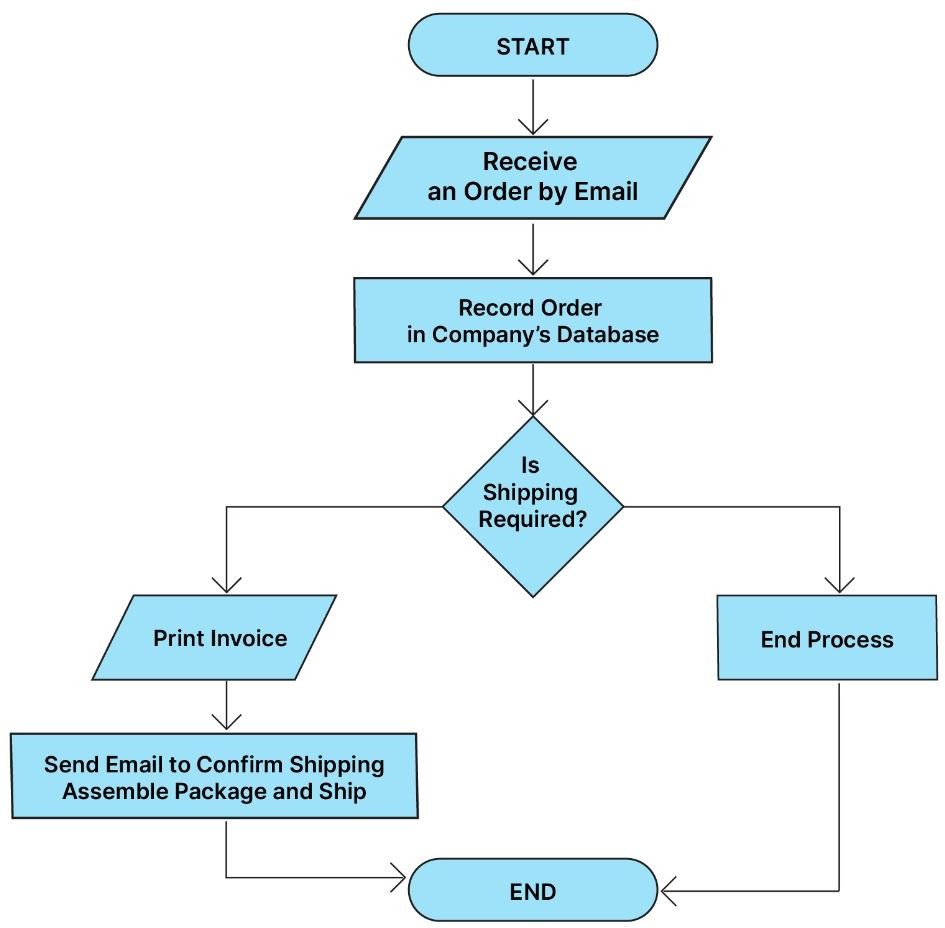
**FLOWCHART DIAGRAM # 04 (b)**



1. Construct the flow chart for the following order processing scenario:
   1. Receive an order by email.
   2. Record it in the company’s database.
   3. Check for shipping.
   4. If yes
      1. Print invoice
      2. Send an email to confirm shipping. iii Assemble package and ship.
   5. If no

i. End Process

### FLOWCHART DIAGRAM # 05



1. Find out a minimum of 5 software (both online and offline) available for flowcharting. Compare them with each other. Select one of them, and state the selection criteria e.g., why have you chosen the software and left the rest. Create the flowchart of Tasks 4 & and 5 using that software.

#### Five Software for Flowcharting:

1. Microsoft Word
2. Microsoft PowerPoint
3. Adobe Illustrator
4. Adobe Express
5. LucidChart

#### Comparison between Flowcharting Software:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Software** | **Suitability for flowcharting** | **Features** | **Ease of use** | **Collaboration** | **Price** |
| Microsoft Word | Good | Templates, shapes, and connectors | Relatively easy | Basic | Freemium |
| Microsoft PowerPoint | Good | Templates, shapes, and connectors | Relatively easy | Basic | Freemium |
| Adobe Illustrator | Good | Comprehensive drawing and editing tools | More difficult | Limited | Paid |
| Adobe Express | Good | Templates, shapes, and connectors | Easy | Basic | Freemium |
| LucidChart | Excellent | Templates, shapes, connectors, collaboration tools, data integration | Easy | Excellent | Paid |

**Software used in this task for Flowcharts:**

#### Adobe Illustrator:

Adobe Illustrator is a vector graphics editor software developed and published by Adobe Inc. It is used for creating digital art, illustrations, and logos. Illustrator is also a popular software for creating flowcharts.

Illustrator has a number of features that make it well-suited for flowcharting, including:

* A wide range of shapes and connectors: Illustrator offers a wide range of pre-made shapes and connectors, which can be used to create flowcharts quickly and easily.
* Customizable templates: Illustrator also offers a variety of customizable flowchart templates, which can be used as a starting point for your own flowcharts.
* Precise drawing and editing tools: Illustrator's precise drawing and editing tools make it easy to create and customize flowcharts to your exact specifications.
* Collaboration tools: Illustrator allows you to collaborate on flowcharts with others in real time. This can be useful if you are working on a flowchart with a team or group of people.

If you are looking for free or low-cost flowcharting software, there are a number of other options available, such as LucidChart, Draw.io, and Creately. These software may not have all of the features of Illustrator, but they are still good options for creating flowcharts.

# Recommendations:

Here are some tips for using flowcharts in programming:

* Start with a clear goal: What do you want to achieve with your flowchart? Are you trying to design a new algorithm, to document an existing algorithm, or to debug a program? Once you know your goal, you can start to draw your flowchart.
* Use the right symbols: Flowchart symbols represent different steps in a process, such as input, output, processing, decision, and branching. Make sure to use the correct symbols in your flowchart.
* Label your flowchart: Label each step in your flowchart with a clear and concise description. This will make it easier to understand your flowchart and to identify any potential problems.
* Test your flowchart: Once you have finished drawing your flowchart, test it to make sure that it works as expected. You can do this by walking through the flowchart step by step.
* Flowcharts are a valuable tool for programmers of all skill levels. By using flowcharts, programmers can improve their communication, understanding, efficiency, and accuracy.

# Conclusion:

Flowcharts are a valuable tool for programmers. They can be used to design and document algorithms, to analyze and debug code, and to communicate complex concepts to others.

Here are some of the benefits of using flowcharts in programming:

* Improved communication: Flowcharts can help to communicate complex programming concepts in a clear and concise way. They can be used to explain code to both technical and non-technical audiences.
* Better understanding: Flowcharts can help programmers to better understand how their code works. By visualizing the flow of a program, it is easier to identify potential problems and areas for improvement.
* Increased efficiency: Flowcharts can help to identify and eliminate bottlenecks in a program. This can lead to increased efficiency and productivity.
* Reduced errors: Flowcharts can help to identify and prevent errors in a program. This can lead to improved quality and reliability.

Flowcharts can be used to represent any algorithm or process, and they can be used at any stage of the programming process. For example, flowcharts can be used to design a new algorithm, document an existing algorithm, or to debug a program that is not working as expected.