Competitive Programming  
Stage 1👶🏻

horizontal line

# Introduction

Hi there, people! :)

These ladders were made with the intent to make you guys self sufficient to learn the basics of C++ and Competitive Programming. The ladders will be divided into several Stages which will include links to theory resources that you need to read and after that, some Problems related to the topic.

The best strategy is to always understand something first before proceeding to the Problems. If at any point of time you feel that the theory resources provided by us are not sufficient or not easy enough, look out for that particular topic on Google. You’ll be amazed at how many tutorials (videos or simple text) you’ll find related to a specific topic. If it still does not ring a bell, you can always contact us, your seniors. :)

*The best way to learn and sustain as a programmer is to try and solve problems on your own. Search for issues on Google. That will save your lives throughout as a programmer. Also remember that the more you struggle with issues/errors, the better a programmer you’ll become.*

*Also never skip stuff just to complete quickly or to come out as a topper in some contest. All of this will be futile in ultimate future. What will stay with you is your learning. So make sure that you are patient throughout and that you completely understand whatever should be learnt.*

*On the image to your left, the third guy will never learn what is important: stability. He’ll be ahead, maybe, but will slip, surely.*

Enough philosophy😂 But, remember all of this. Now, let’s begin with learning C++!

All the best!

# Part 1: Basics of C++

## Basic IDEs

If you need a computer to do some specific task for you, you need to provide specific instructions to the computer. These instructions are delivered in the form of a **program**. And you, you’ll be called a **programmer**.

A program once written by you, needs to be run by the computer on your command. This is called the **execution** of a program.

The writing, editing, debugging (removal of errors) and execution(running) of a program is facilitated easily by software known as **IDEs** (Integrated Development Environment). They’ll make your life a lot easier. ;) There are many IDEs, the popular ones being CodeBlocks, Visual Studio Code etc. You may install any one of these. You’ll be helped in the installation in a tutorial series in the next section, keep going!

## What does a program do?

Whenever you write up a program, it does some sort of task. That task may involve anything like printing something on the screen or maybe calculating something as per your logic and instructions.

Also, throughout Stage 1 of these ladders, you need to follow this tutorial series:

[Video Tutorial Series for C++](https://www.youtube.com/playlist?list=PLAE85DE8440AA6B83)

Since we won’t be able to explain everything in detail, this series will fill in for that. This series covers everything in the basic syntax of C++ as a language. Before proceeding any further, watch the first 4 videos from this series. After watching them, go through this tutorial below. In this link, read **only** the section labelled as **‘C++ Intro’** *(in the panel on left)* for now.

<https://www.w3schools.com/cpp/cpp_intro.asp>

**HELP SECTION:** If someone is stuck on installation of an IDE like CodeBlocks, you can refer to the following steps. Steps to run a C++ code on your computer:

**Windows:**

Using CodeBlocks or Dev C++ is best for Windows as of now.

1. You may refer to this video for steps to install CodeBlocks: <https://www.youtube.com/watch?v=tvC1WCdV1XU&list=PLAE85DE8440AA6B83&index=1>
2. To install Dev C++: <https://www.youtube.com/watch?v=EgwndDGnm5w>

You can also work on text editors like Sublime Text, but in that case, you will require installation of the g++ package, so the above two are preferable as of now.

**Linux:**

1. Install g++ using your distro's package manager. Then run 'g++ FileName.cpp'
2. You may install sublime-text <https://www.sublimetext.com/docs/3/linux_repositories.html> and change its script. To change that you can refer : <https://www.youtube.com/watch?v=bGUiJ8lpvlY>. **To get script used in this link, go to** <https://github.com/shikharkunal99/Sublime-Build-System>

## Basic Syntax

Now that you have an IDE installed, let's move on to learning the rules of how to write a program in C++. These rules are also known as the **‘syntax’** of the program. So, lets begin!

First off, go through this link which will explain the basic structure of a program:

<https://www.w3schools.com/cpp/cpp_syntax.asp>

Now, let’s move further to explain the basic syntax of C++.

### **Preprocessor Directive** *(Complex name, but don’t worry* 😉 *)*

You might be wondering what exactly is meant by the line like #include<iostream.h> on the top of a C++ program. Read this answer for an idea. This would be enough to give you a basic understanding of the statement

<https://www.quora.com/What-means-include-iostream-h-in-C++>

When we include some file/class on the top of our C++ program, we do so because we feel that its contents might be of use to us in the program. For eg, here “iostream.h” is a file which includes some contents that enable a C++ program to carry out input and output. A C++ program cannot do that on its own, therefore this file helps it to do so.

### Input and Output using C++

* ***Theory***
  + The most basic form of operation that we need to carry out in any C++ program is input and output. This is carried out in various ways. You’ll mostly be using ‘cout’ and ‘cin’ for output and input respectively. Have a look at these articles for a detailed knowledge:

<https://www.w3schools.com/cpp/cpp_output.asp>

<https://www.w3schools.in/cplusplus-tutorial/input-output/>

*(If you don't understand the part headed ’put() and get() functions’, it’s totally okay. Go back to it later, once you complete Part 5 of this ladder. We’ll remind you. Don’t worry :) )*

* + Also, you might have noticed something like ‘\n’ in the tutorial above. There are more such notations and they are known as **‘Escape Sequences’**. They come in handy while printing stuff. Read more about them here:

<https://ciphertrick.com/c-beginners-introduction-to-escape-sequences/>

* + Also read about the difference between ‘endl’ and ‘\n’. They almost have the same use, but there is a difference, which you’ll figure out later : <https://www.geeksforgeeks.org/endl-vs-n-in-cpp/>

### Comments, Variables and Keywords

* ***Theory***
  + A program may become very complex at certain times and a requirement may arise that you wish to leave messages for the future reader of your program so that he/she can easily grasp your code. This facility to leave behind messages is facilitated by the use of **comments** in C++. Read more here: <https://www.w3schools.com/cpp/cpp_comments.asp>
  + Just like variables in Mathematics, we also have variables in C++. They are merely locations or names by which you’ll store your data. And keywords are basically certain words which can **only** be used to denote a special meaning or use in C++. Read more about variables and keywords here: <https://www.geeksforgeeks.org/variables-and-keywords-in-c/>

### Data-types

* ***Theory***

As suggested by name, C++ data types define the type of data that you are going to store in a variable. C++ has some predefined data-types that you’ll be using, mostly, and it also gives you the facility to create data types of your own (to be used later on). Read more on these links:

<https://www.w3schools.com/cpp/cpp_data_types.asp>

<https://www.geeksforgeeks.org/c-data-types/>

***Note:*** *You can also use printf instead of cout and scanf instead of cin for input and output in C++. They were used in C (predecessor of C++), but are still equivalently valid in C++. Read more here:* [*https://www.programiz.com/c-programming/c-input-output*](https://www.programiz.com/c-programming/c-input-output)

### Declaration and Initialization

* ***Theory***

No theory from our side😛. Go here :

<https://tutorialink.com/cpp/declaration-of-variables.cpp>

***Note:*** *Before proceeding any further, go through uptil video number 7 of the* [*video tutorial series mentioned above*](https://www.youtube.com/playlist?list=PLAE85DE8440AA6B83)*.*

### Operators

* ***Theory***

Again, just like Mathematics, we also have operators in C++. Operators are used to perform specific calculations or processing on data. Read more about them here:

<https://www.geeksforgeeks.org/operators-c-c/> *(Also read from the links within this article. After this you should have an idea about Arithmetic, Relational, Logical, Bitwise, and Assignment operators in detail.)*

## Problems on Part 1: Basics of C++

| Sr. No. | Question | Solved or Not  (Y/N) | Problems faced/ Comments(About approach, if any) |
| --- | --- | --- | --- |
| 1. | Temperature of a city in Fahrenheit degrees is input through the keyboard. Write a program to convert this temperature into Centigrade degrees. |  |  |
| 2. | If a five-digit number is input through the keyboard, write a program to reverse the number.  (Hint: Use the modulus operator ‘%’) |  |  |
| 3. | <https://www.hackerrank.com/challenges/cpp-input-and-output/problem> |  |  |
| 4. | <https://www.hackerrank.com/challenges/c-tutorial-basic-data-types/problem> |  |  |
| 5. | If a five-digit number is input through the keyboard, write a program to print a new number by adding one to each of its digits. Example if the number that is input is 123 then the output should be displayed as 234. |  |  |
| 6. | A cashier has currency notes of denominations 10, 50 and 100. If the amount to be withdrawn is input through the keyboard in hundreds, find the total number of currency notes of each denomination the cashier will have to give to the withdrawer. |  |  |

***Note:*** *Before proceeding any further, go through with video number 8,16, 17 and 25 of the* [*video tutorial series mentioned above*](https://www.youtube.com/playlist?list=PLAE85DE8440AA6B83)*. Don’t worry about the left out videos, we’ll cover them later on.*

# Part 2: Conditional Statements (if-else & switch)

We have to deal with Decision making every now and then in our day-to-day lives. Our clarity in a given situation depends on how effective decisions we took. What if a similar situation arises in our program? What if you want to execute a block of code if and only if a particular condition is satisfied? This is where the need for conditional statements creeps in. This sort of task can be accomplished in two ways. Let’s explore them.

## If else

This is the first method to accomplish decision-making. It is almost exclusively used by programmers in preference over the second method.

### *Theory*

* + [Text Theory](https://www.geeksforgeeks.org/decision-making-c-c-else-nested-else/amp/)
  + Video Tutorials:

[1.](https://docs.google.com/document/d/1QP3BUuZvu0FeOcO4jxbZHzOPHwyQm4L2fhPkzx7TZ1s/edit#)

[2.](https://docs.google.com/document/d/1QP3BUuZvu0FeOcO4jxbZHzOPHwyQm4L2fhPkzx7TZ1s/edit#)

[3.](https://docs.google.com/document/d/1QP3BUuZvu0FeOcO4jxbZHzOPHwyQm4L2fhPkzx7TZ1s/edit#)

## Switch

This is the second way for decision-making specially used in case we have to compare a given value with an entity exactly. Hence it is applicable only for *int* and *char* data types.

### *Theory*

[Theory for Switch in C++](https://www.tutorialspoint.com/cplusplus/cpp_switch_statement.htm)

## Problems on Part 2: Conditional Statements

| Sr. No. | Question | Solved or Not  (Y/N) | Problems faced/ Comments(About approach, if any) |
| --- | --- | --- | --- |
| 1. | <https://www.hackerrank.com/challenges/c-tutorial-conditional-if-else/problem> |  |  |
| 2. | <http://codeforces.com/problemset/problem/4/A> |  |  |
| 3. | Given a point (x, y), write a program to find out if it lies on the x-axis, y-axis or at the origin, viz. (0, 0). |  |  |
| 4. | Any year is input through the keyboard. Write a program to determine whether the year is a leap year or not.  (Hint: Use the % (modulus) operator) |  |  |
| 5. | According to the Gregorian calendar, it was Monday on the date 01/01/1900. If any year is input through the keyboard, write a program to find out what is the day on 1st January of this year. |  |  |
| 6. | Given the coordinates (x, y) of the center of a circle and its radius, write a program which will determine whether a point lies inside the circle, on the circle or outside the circle.  (Hint: Use sqrt( ) and pow( ) functions. Read about them on the internet.) |  |  |

***Note:*** *Before proceeding any further, go through with video number 18-24 of the* [*video tutorial series mentioned above*](https://www.youtube.com/playlist?list=PLAE85DE8440AA6B83)*.*

# Part 3: Flow of control (for, while, do-while loops)

## Loops

Ever wondered how would you execute a given task multiple times or repeatedly?

For example: if you were told to print a given statement say 10 times, you would easily come up with an idea of copy-pasting the same statement 10 times, right? But what if you were told to do this maybe for a 1000 times or 10,000 times?

Here’s where the idea of [LOOPS](https://www.geeksforgeeks.org/loops-in-c-and-cpp/) comes in.

### *Theory*

* + [For Loop](https://www.programiz.com/cpp-programming/for-loop)
  + [While Loop](https://www.tutorialspoint.com/cplusplus/cpp_while_loop.htm)
  + [Do-while loop](https://www.tutorialspoint.com/cplusplus/cpp_do_while_loop.htm)

## Jump Statements

These are the statements which are used to abruptly change the flow of control or in other words they transfer the control from one place to another place in a program.

1. [continue](https://www.geeksforgeeks.org/continue-statement-cpp/)
2. [goto](https://www.geeksforgeeks.org/goto-statement-in-c-cpp/)
3. [break](https://www.geeksforgeeks.org/break-statement-cc/)
4. [return](https://docs.microsoft.com/en-us/cpp/cpp/return-statement-cpp?view=vs-2019)

## Problems on Part 3: Flow of Control

| Sr. No. | Question | Solved or Not  (Y/N) | Problems faced/ Comments(About approach, if any) |
| --- | --- | --- | --- |
| 1. | <https://www.hackerrank.com/challenges/c-tutorial-for-loop/problem> |  |  |
| 2. | <https://www.hackerrank.com/challenges/30-loops/problem> |  |  |
| 3. | <https://codeforces.com/problemset/problem/246/A> |  |  |
| 4. | Write a program to print all prime numbers from 1 to 300.  (Hint: Use nested loops, break and continue) |  |  |
| 5. | Write a program to add first seven terms of the following series using a for loop:  (1/1! ) + (2/2!) + (3/3!) + …… |  |  |
| 6. | Write a program to produce the following output: |  |  |
| 7. | Write a program to produce the following output: |  |  |
| 8. | The natural logarithm can be approximated by the following series.    If x is input through the keyboard, write a program to calculate the sum of first seven terms of this series. |  |  |

***Note:*** *Before proceeding any further, go through with video number 32-37****(leaving out 35)*** *of the* [*video tutorial series mentioned above*](https://www.youtube.com/playlist?list=PLAE85DE8440AA6B83)*.*

# Part 4: Arrays

# Suppose you were given the task to store and display the scores of your classmates and then calculate the overall average score of the class. How would you approach? A basic idea would be to declare as many variables as the number of students and then store their scores individually and then add them up. Isn’t it? But just for once consider how cumbersome it would be to do so, declaring numerous variables, taking numerous inputs and then adding them up!

# Don’t worry, we have got a beautiful solution to this problem! Arrays come to your rescue whenever you want to declare numerous variables of the same data type to store similar information.

### *Theory*

* + [What are arrays?](https://www.tutorialspoint.com/cplusplus/cpp_arrays.htm)

***Note:*** *Before proceeding any further, go through with video number 9-15 and 26-31 and 35 of the* [*video tutorial series mentioned above*](https://www.youtube.com/playlist?list=PLAE85DE8440AA6B83)*. (This note assigns too many videos to watch. Don’t get afraid of functions :P. They are pretty simple. They have a wide variety of applications, hence so many videos.)*

# Part 5: Functions

Functions in programming are similar to the functions in mathematics. They take various parameters and return some value (even a void sometimes :P). Just like creating a machine that would take some raw material (input) and return some product (output). Functions are convenient for programming as one would not have to write the whole process again and again for doing the same task. Functions will be the basis for recursion, to be studied later on.

### *Theory*

* + Video Tutorials:
    - Go through [this](https://www.youtube.com/watch?v=R0dGmkVf0b0) link to get more on Functions.
    - Try [this](https://www.youtube.com/watch?v=bsWWHo4KDHE&list=PLAE85DE8440AA6B83&index=9) playlist till video 11
  + [Text Tutorials](http://www.cplusplus.com/doc/tutorial/functions/)

***Note:*** *Go back to ‘Input and Output using C++’ section of Part 1 and understand the put() and get() functions which were left out at that time. :)*

# 

# Task for Stage 1

In the beginning stages, there won't be any algorithms that you'll have to  
learn. The focus will be completely on getting good at writing code and  
developing intuition and logical thinking.

* Register for an account on CodeForces (<http://codeforces.com/>) . Your handle will be your identity in the competitive programming community, so choose a cool one!!
* Create an account on [A2OJ](https://a2oj.com/) and register for an account. After logging in, go to My  
  Account, and enter your codeforces username/handle in 'Codeforces Handle'.
* Go to <https://a2oj.com/ladders> and on the bottom of the page click on the link saying [Codeforces Div. 2, A](https://a2oj.com/ladder?ID=4). You'll have a list that links to Codeforces problems.  
  After solving each problem, a new one will unlock in a few minutes. Solve  
  at least 30 problems before you feel they are easy for you.
* You can further practice problems on hackerrank. Start with Warm-up questions (<https://www.hackerrank.com/domains/algorithms?filters%5Bsubdomains%5D%5B%5D=warmup&badge_type=problem-solving> )
* After this, you can move to implementation part (<https://www.hackerrank.com/domains/algorithms?filters%5Bsubdomains%5D%5B%5D=implementation&badge_type=problem-solving> ).

**HELP SECTION:** Try to solve these in 2-3 weeks. You might get many errors but that happens with everyone while learning. Try Googling the errors you get. Many sites like stackoverflow ( <https://stackoverflow.com/> ) have really good answers. GeeksforGeeks (<https://www.geeksforgeeks.org/c-plus-plus/> ) and cplusplus reference (<http://www.cplusplus.com/reference/> ) contain detailed explanations and examples which may be referred. You can refer to discussion or editorials for a given question but it is highly recommended to first try your best to solve problems on your own before opening editorials.

If you want to know more about competitive programming sites, check out: <https://blog.codingblocks.com/2019/sites-and-tools-for-competitive-programming/>

***Ek concluding meme toh banta hai!*😛😛**



*Don’t just scroll past it! Try and understand that big things take patience and time. So, be patient and make sure that you cover every feat necessary.*

--------------------- Stage 1 Ends ----------------------