
Headstart
Coding Club, IIT Guwahati
< WEEK 1 >

*Join the discord server for clearing all your doubts

What is Competitive Programming or CP?

- You are given a set of problems and you have to solve these problems in a fixed time.
- After solving, depending on your rank/percentile you will be awarded some virtual points.

Why should you do it?

- Mainly because it's competitive and programming :)
- The Competitive part gives you a feel of sports and the Programming part might help you in your **future courses, internships, and placements.**

CONTENT

PREREQUISITES

(Expected Time: 2-3 days)

For this course, there are no prerequisites as such. But to start CP you will need a decent level of knowledge of at least one programming language.

- General Order of Preference:-

  C++ > Java > Python > Others
- It isn't that tough to switch between languages if you have a hold of at least one language.
- We recommend you start with C++ if it is going to be your first programming language.

If you already have knowledge of Python, we have linked resources for CP in python, so you can continue with the course. However, we recommend to learn C++ along the way as it's better for more difficult problems and will help you in your college curriculum as well.

SETTING UP AN IDE (Integrated Development Environment)

In simple words, an IDE is an environment that you will be using to write your code, compile it and view the output. A suitable and well-equipped IDE is very important, and not setting it up in the right way may cause trouble in the future.
(you may spend days fixing it).

Two of the most popular ones are:-

1. **VS Code**
   a. **Windows**
   b. **Mac**
2. **Sublime Text**

Don't mess up while setting it up otherwise you will easily waste days fixing it!!

STARTING WITH YOUR FIRST PROGRAMMING LANGUAGE

What we are going to recommend is the fastest and preferred way to learn C++ / Python. These are short videos but are more than sufficient.

Resources for C++

1. **Bucky's Tutorial**: Watch lectures 1 to 41 and 71 to 73.
2. **CP Course by Luv**: 4 to 12 lectures. (For those comfortable with Hindi)

Resources for Python

1. **Programming with Mosh**: The first three hours would be sufficient.
2. **Corey Schafer - STL**

Solve the First 10-20 problems from **Hackerrank** just to get comfortable with the language you chose.

Note: **Do not waste much of your time learning the language** (getting into tiny details).

TOOLS FOR COMPETITIVE PROGRAMMING

Note: You don't need to set these up to begin with CP, however, along the way these tools become really helpful to your journey. You may revisit this when you get comfortable with CP.

1. **VS Code extensions:**
   To install these extensions, look for the "Extensions" button on the left sidebar. Then you can search for and install them
   - **Code Runner**: With this, you can directly run the code without having to compile in the terminal . Caution: Sometimes, though rarely, this extension does not work properly.

- **Competitive Programming Helper:** It automatically reads test cases from the website which you won't have to write again and again. To work with this you also have to install the **Competitive Companion Chrome extension** .

2. Sublime Text Packages:

Similar to the cph extension in VS Code, you can find **FastOlympicCoding** in Sublime which works with Competitive Companion .

3. User Code Snippets/Template code:

These are reusable code pieces that can help you avoid re-writing the same code. You can make them in both VS Code and Sublime Text.

4. Header Files

Including all libraries required can be a tedious task. So we prefer including *<bits/stdc++.h>* file instead, which already contains all the libraries. It is not used in usual C++ codes as it takes a long time to compile. However, that is not important in CP. It is available from **C++14 and above** and it is **not directly available for Mac users.**

Mac users can refer **this** video. The folder with include files may not be the same. To find that, you can make a C++ file, type *#include <iostream>* and Ctrl + Click on it to see the file location.

5. Speeding up I/O

If you are using C++, add the following lines in your main function:

```
ios::sync_with_stdio(0);
cin.tie(0);
```

The python equivalent for the same is :

```
import sys
input = lambda: sys.stdin.readline().rstrip("\r\n")
```

By adding "ios::sync_with_stdio(0);", you are essentially telling the C++ streams not to synchronise with the C I/O functions. This can lead to a speedup, especially when there's a lot of I/O in your program.

Warning: You won't be able to use C functions like scanf and printf in your C++ code properly thereafter since we have desynchronized C++ input and output streams with the standard C I/O functions.

In C++, standard input (cin) and standard output (cout) streams are tied by default. This means that whenever you perform an input operation using cin, it automatically flushes the output buffer of cout.

By using cin.tie(0);, you break the tie between cin and cout. This means that input operations (cin) won't automatically flush the output buffer of cout. This can lead to better performance in scenarios where you need to optimize I/O operations.

RESOURCES

1. **Handbook (CPH)**:- You can find 90% of the things you want to learn in this book, but reading this book completely and then starting to solve problems is not recommended. Just refer to the book when you come across a topic that you haven't heard about before.

2. **CSES**:- The author of the Handbook has also made a compilation of 300 really good and conceptual problems which covers almost all CP topics. But it doesn't contain problems sorted according to difficulty so you will have to estimate it using solve count.

3. **USACO Guide**:- This is literally the "one-stop shop". This website is generally referred to as USA computing Olympiad aspirants and it has everything in sorted order based on divisions.

4. **CP-Algorithms**:- Contains all CP topics but again not in sorted order of difficulty. So, refer to it only if you are stuck on a topic and couldn't find it in the Handbook

5. **OEIS**:- Short for The On-Line Encyclopedia of Integer Sequences. You may look for a specific formula satisfying the initial integer sequence pattern you devised for a problem.

WEBSITES

Some websites from where you can learn and compete and some important competitions**.**

1. **Codeforces**:- Most popular and widely used platform.
    - Contests are mainly of the following types:- Div 4, Div 3, Div 2, Div 1, Div 1+2, Educational Rounds, and Global Rounds.
    - We recommend participating in all rounds without worrying about rating changes. (Experience matters more than ratings).
    - Do try to solve all the problems whose ratings are <= your rating + 300. If you are not able to solve the problems, refer to their editorial. If you still don't understand the solution, discuss it with your friends or ask your doubts in the competitive-programming-discussion channel on our Discord.
    - Try not to miss Div 3 contest, they contain good problems touching upon various important techniques and problem types.
2. **Atcoder**:- Educational and Weekly contests with quality problems.
    - 3 types of contests:- Atcoder Beginner/Regular/Grand contest
    - Must participate in Atcoder Beginner Contests, as it covers a good variety of problems.
3. **Codechef**:-
    - Monthly contests:- Long challenge and Lunch-time
    - Weekly contests named starters

It is highly recommended to start giving contests regularly once you have some basic knowledge about CP.

IMPORTANT YEARLY COMPETITIONS

* We will send you a reminder before all these contests when to participate.

1. **Meta HackerCup**:- It is a once-a-year contest having various eliminatory rounds.
2. **ACM ICPC**:- It is also held once a year and the Biggest CP contest for college students. It is known as the **Olympics of CP**. It is held in a topological order.
   - Online -> Regionals -> World Finals
3. **Advent of Code**:- Advent of Code is an annual set of Christmas-themed computer programming challenges which offers new programming puzzles each day from December 1 to 25.
4. **TCS CodeVita**:- TCS (Tata Consulting Services) organises one of the largest global programming competitions. According to the new format, there are going to be many rounds in this competition.
5. **LIT**:- LIT (Lexington Informatics Tournament) is a competitive programming tournament for contestants of all levels, hosted by members of the LexMACS club. Participants can participate in teams of three.
6. **Code Gladiators**: Code Gladiators is an annual coding competition by TechGig. The Open Contest is conducted in three rounds: open-round, semi-finals and finale.
7. **Yandex Algorithms**: Yandex Cup is a competition organised in various tracks, including App Development, ML, and CP.
8. **USACO Competitions**: USA Computing Olympiad (USACO) is an online competition for individuals that is held four times a year. In each contest, USACO has the following four divisions such as Bronze, Silver, Gold and Platinum from beginner to advanced. Each division has it's own problem set.

PAST IMPORTANT COMPETITIONS

* These competitions have been discontinued but it is good to know that they existed :)

1. **Google Contests**:- Google used to host a variety of contests which are obsolete now. You can explore the archives of the same provided below.
   a. **Google Kickstart**:
   - Suited for beginners and contains a good variety of problems but some can be irritating too.

   b. **Google Codejam**:
   - Multi level elimination round with increasing difficulty, not very beginner friendly.

   c. **Google Hashcode**:
   - Not your everyday CP contest, you have optimization task for which the optimal answer may be unknown. The more optimized the solution, the more points you get.

2. **TopCoder Open**:- TopCoder used to organise various competitions like Yandex, which included tracks like Design, Data Science, CP etc.

IMPLEMENTATION

(Expected Time: 1 day)

After we have learned a language, we can now get started with actual problem-solving.

Problem-solving requires understanding the problem statement and designing an effective algorithm that successfully solves the task at hand.

The efficiency of algorithms is very important in competitive programming. Usually, it is easy to design an algorithm that solves the problem slowly, but the real challenge is to invent a **fast algorithm**. If the algorithm is too slow, it will get only partial points or no points at all.

Thus comes into the picture the concept of **time complexity**. By calculating the time complexity, we can find out whether the algorithm is fast enough without implementing it.

Here are some resources which will help you understand time complexity: -

1. **CPH** (Competitive Programmers' Handbook) Chapter 2.
   - Do read this chapter thoroughly without skipping to get a complete understanding of Big O notation etc.
2. **Big-O notation in 5 minutes**
3. This **blog** might help you in avoiding Time Limit Exceeded error

Some more aspects of a language become handy while doing CP problems. One such aspect, which we usually ignore when learning the language is that of range of numbers.

In Python3, the int type stores large enough numbers, so you don't have to worry about the value which will be stored in the variable.

In C++, **int** is a data type that takes up 32 bit of space and can store integers ranging from **-2,147,483,648 to 2,147,483,647**. However, the answers in the problems very often go over that range and give wrong output. This is called **integer overflow**. Hence, we can use **long long** data type over **int** in such cases. You can read more about it **here**.

Also Read : **Difference between float and double**

Following are some challenging problems based mainly on implementation:-

- **Increasing Array (Solution)**
- **Permutations (Solution)**
- **Number Spiral (Solution)**
- **Chloe and the sequence (Solution)**
- **Viki and Squares (Solution)**
- **Rudolph and Christmas Tree (Solution)**
- **Masha and two friends (Solution)**

*Try to attempt a question for a minimum of 30 minutes before checking the solution.

*We have attached the codes with solutions. If you still have any doubts feel free to ask us in the discord channel.

*In general, the solutions to the problems can be found in the Editorial/ Tutorial section on websites like Codeforces, Atcoder etc. For CSES, most of the solutions can be found on google.

MATHS

(Expected Time: 0-1 days)

There are several questions in CP, which can be solved only by observations and logical reasoning skills. No algorithm is needed as such to solve these kinds of problems. Such questions test the intuitive/ mathematical skills of the programmer, and not his/her coding skills.

Try to solve these given problems without using any known algorithms:-

- **Hard Calculation**
- **Who's Opposite**
- **Comma**
- **Fraction Floor Sum**

Additional Questions for practice :

- **Histogram Ugliness**
- **Cat Cycle**
- **Recent Actions**
- **Fibonaccharsis**

AD-HOC PROBLEMS *(Expected Time: 0-1 days)*

Ad-hoc problems are those whose algorithms do not fall into standard categories with well-studied solutions. There are no general techniques to solve them. They are intuition-based problems, and you may understand better by solving these:

- **Teleportation**
- **Funny Permutation**
- **Three Doors**
- **Sleepy Cow Herding**

Additional Questions for Practice:

- **Exchange**
- **MIN-MEX Cut**
- **Sleepy Cow Sorting**
- **Difference of GCDs**

General Advice

- Try to **write readable codes**. Readability makes the thinking process clearer and faster.
- Try to **visualize a pseudo code** to grasp the flow of code before directly jumping into implementation. No point of an early submission if it does not get accepted.
- Always try to solve that one problem you could not do in contest, after the contest ends. **Consistent upsolving** contributes significantly in improving CP skills.
- If you turn to reading editorial, read hints/partial solution and give it a try again. **Implementing solutions on your own** guarantees that you properly understood the problem and the solution.
- Maintain **healthy competition**, Discuss your solutions with your friends after contest ends. Doing CP with friends not only makes it more exciting and fun but also helps in discovering different ways to approach a problem.
- You will definitely have contests where you underperform and your rating drops. Brush that negative feeling off as quick as possible. **You always have the next contest waiting for you :)**
- Submission of every participant is visible. Follow people who write neat codes, sometimes if the editorial solution is tough to understand, you can try to make sense out of their code. Some of them are :
  tourist **jiangly** **ecnerwala** **errichto** **yangster67**

Sneak Peek

Excited for the next week? So are we! Expect a lot of new thrilling topics and handful of challenges waiting to be conquered.
You will be introduced to STL, Sorting and many more topics which are present in one form or other in almost all questions.
Solve plenty of problems in the mean time and cheers for coming this far!
See you next week!