Part - 1

This is week-1 content. We will be checking out the following things in this week-1

- Configuring VsCode in Windows
  - This is **not** required to be done if you can run C++ code in your laptop.
  - **Note that we won't be using simplecpp that some of you would have used during cs101**.
- Compiling and executing C++ codes in Linux and Mac
  - This is **not** required to be done if you can run C++ code in your laptop.
- Basic Number theory :
- Bitwise Operators
- Time complexity
- Now, to the real stuff, Data structures in C++ STL
  - Stacks
  - Queues
  - Priority Queues
  - Double ended queues
  - Vectors
  - Bitset
  - Set
  - Unordered set
  - Maps
  - Multimaps
  - Pair

Few Common Resources

https://cses.fi/book/book.pdf: Competitive Programmer's handbook (must be followed)

Whenever in future weeks: **book refers to the above book.**

https://duoblogger.github.io/assets/pdf/memonvyftw/guide-t-cp.pdf : other version of the same book.

https://usaco.guide/PAPS.pdf#page=15 : principles of algorithmic Problem solving

https://cses.fi/problemset/ : Important problem set for learning CP

https://usaco.guide/problems/ : Problem set for USACO (This is advanced, requires a lot of knowledge)

https://cp-algorithms.com/ : a good website which has most of the important theorems and algorithms.

Configuring VsCode in Windows

There are mainly two ways to configure it. Do it only if you haven't been able to run C++ code in windows.

One is to configure it directly. You can follow the link given to configure VsCode for C++, so that you can conveniently code for competitive programming questions. It is not necessary that you have to use the resource that we have given. It's fine to use any certified resource.

link : https://code.visualstudio.com/docs/cpp/config-mingw

Second one is to use WSL.

Link for installing WSL : https://learn.microsoft.com/en-us/windows/wsl/install

(contact Siddharth Bhuva in case of any problems for installation.)

Compiling and executing C++ codes in Linux and Mac

People using Linux and Mac can actually use terminal to compile. This is hard for people in windows.

1. open the terminal in the folder where you code is present. This can be done by a right click, followed by selecting the option, open in terminal.

2. use ls command to check if the file that you want to compile and run is in the directory of your terminal. then use g++ to compile the filename. and use ./a.out to run the code.

```
g++ <filename>
./a.out
```

where <filename> has to be replaced by the name of .cpp file. OR

```
g++ <filename> -o <binaryname>
./<binaryname>
```

here <binaryname> has to be replaced by the name of the binary file that you wanted. Note that this will be a.out by default.

Basic Number theory

> Basic-Numbertheory.pdf

This same pdf has been posted in the WhatsApp group too. Number theory is a huge topic but only a part of it is important now. The topics that are important now are:

- GCD and Euler's theorem: https://crypto.stanford.edu/pbc/notes/numbertheory/euclid.html
- Modular arithmetic: https://crypto.stanford.edu/pbc/notes/numbertheory/arith.html
- Euler's totient function: https://crypto.stanford.edu/pbc/notes/numbertheory/units.html

A book which contains all of these and more : https://crypto.stanford.edu/pbc/notes/numbertheory/book.pdf

Note that this project is not about number theory, so we expect you to go through these three topics and work on other parts. do check out the chapter 21 of the book. Although,

it’s fine if you don’t.

Bitwise operators

https://www.geeksforgeeks.org/bitwise-operators-in-c-cpp/

use the above link to know more about these operators and do check out what these operators do. These operators sometimes makes the work faster. for example, dividing by 2^16 will take more time than right shifting 16 times. There is a wide array of problems completely based on bitwise operations on binary numbers.

Now to the Good Part,

Time complexity

This is the way by which we measure how efficient your program is. Learn about Big-O notation from https://www.geeksforgeeks.org/analysis-algorithms-big-o-analysis/ (This website). then check out the **chapter-2 in the book.**

It is important for all of you to be **confident about time complexity**.

Other resources : https://usaco.guide/bronze/time-comp?lang=cpp

This is a good resource that i had found lately.

Data Structures that you will be using frequently

So what’s the use of data structures? , they make the work better and faster. They can be used for a lot of problems in CP. Learn the basic idea behind the data structures and the syntax for those data Structures in C++ STL from the book **Chapter:4** in **Competitive programming handbook.**

For the methods in the various data structures other than given in the book. Refer Geeks for Geeks websites. Links are pasted here.

Stacks: https://www.geeksforgeeks.org/stack-in-cpp-stl/

Queue: https://www.geeksforgeeks.org/queue-cpp-stl/

Priority Queue: https://www.geeksforgeeks.org/priority-queue-in-cpp-stl/

Heap : https://www.geeksforgeeks.org/cpp-stl-heap/

Double ended Queue: https://www.geeksforgeeks.org/deque-cpp-stl/

Vector: https://www.geeksforgeeks.org/vector-in-cpp-stl/

Set: https://www.geeksforgeeks.org/set-in-cpp-stl/

Map: https://www.geeksforgeeks.org/map-associative-containers-the-c-standard-template-library-stl/

Pair: https://www.geeksforgeeks.org/pair-in-cpp-stl/

List : https://www.geeksforgeeks.org/list-cpp-stl/

There are few data structures which are not in the book. For those not in the book, try to learn them from the website. The important part that you will have to learn is **not the syntax, but the specialty of each data structure. You should get the idea of what each data structure means.** Correctly deciding the data structure to solve the problem might reduce the problem size to half.

Few Tips while solving questions:

1. Practice makes man perfect. To master CP, practice is must. One should practice regularly and learn regularly.
2. You can use the internet to search for syntax for data structures till you can do it by yourself.
3. If you find any C++ syntax weird in the book, try to google it up or message in WhatsApp group or dm any one of the three mentors.
4. Try to incorporate data structures while solving a problem only if it increases the efficiency of the code, that is only if the code takes less time.