**Choosing a language for Competitive**

Well, this has been a long debate always as to what to choose as one's language for Competitive Programming. While most companies generally offer a huge pool of choices in which you can attempt the questions, the maximum clutter is seen around C++ vs Java. I myself am a huge fan of Java (have learnt the same for 6 years now) but had to go for C++ for all Competitive and Coding purposes.

My major reasons for the same were the following. The speed of C++ is comparatively faster than JAVA. This is ensured by the fact that C++ is closer to the machine than JAVA in terms of abstraction (compiler gyaan heh). C++ does offer a wider control over your data than Java. Java has a lot of features like autoboxing/unboxing and garbage collection which you have to do yourself. This ensures that you keep your code optimised to discount on the time you are spending on trivial issues. C++ has also got a richer STL than Java in my opinion. Finally, the most important reason for the same is the fact that 75% of the literature available for Competitive is in C++. You will get stuck while coding and C++ will have solutions to offer atleast. Very few good resources or approaches are available in native Java and that does make life tough, especially in the higher levels of problems.

**RESOURCES**

So coming down to the resources for CPP. There are a hell lot of resources available for the same and I feel most of you must have scraped them by date. For people who have prior experience in CPP, you can revise your concepts on IB's site itself (<https://www.interviewbit.com/courses/fast-track-cpp/> ).

For people starting new, the major topics can again be seen on the link above to know the required topics you need to search. Some good resources are given below:

**STL** : <https://www.geeksforgeeks.org/the-c-standard-template-library-stl/>

Refer the STL algos definitely. Sort will be used by us heavily. pair<> is again something that is a programmer's biggest friend. STL data structures like stack, queue, dequeue, priority lists and set are unquestionably important.

**Pointers:**

Am adding resources from CP itself. So all courtesy to Jennifer Maam and Pratik sir for the same

<https://docs.google.com/presentation/d/19BghBSUfWqQAHUeWaW_vNnPufptP0cT5/edit?usp=sharing&ouid=114926424821290218963&rtpof=true&sd=true>

<https://docs.google.com/presentation/d/1yV12Dp_Am9g1v229sw2uRix__DjPblKp/edit?usp=sharing&ouid=114926424821290218963&rtpof=true&sd=true>

Although not something that is used very very frequently, still a very important concept. Best resource for them is our own CP course where we dealt in detail with pointer arithmetic and referencing/dereferencing.