

## Competitive Programming resources

### **Learning resources:**

<https://cp-algorithms.com/>

Geekforgeeks

Use YouTube where required, but first try blogs.

### **Online Judges:**

Codeforces, CodeChef, Atcoder

### **Practice Sets:**

CSES: <https://cses.fi/problemset/>

A2OJ ladder: <https://github.com/rishabhdeepsingh/A2OJ-Ladder>

Codeforces archives

Atcoder archives: <https://kenkoooo.com/atcoder/>

### **Intern test prep and interview prep:**

Interview Bit, Leetcode (start this in summer break)

### **Where to start?**

1. Learn basic C++, even if you are already familiar with other languages, get proficiency in C++.
2. Try solving A ladder in A2OJ Ladders. Don't complete the whole ladder, do it till you have confidence in basic IO, control flow and basic DS such as arrays. Initially work on building your intuition rather than theory, then at the end time, focus on predefined algorithms and patterns.
3. Learn following topic from STL: Set, multiset, vector, map, multimap, stack, queue, list, double list, dequeue, sorting algorithms etc. Have clear idea of time complexities of insertion and deletion, space complexity and its basic methods.
4. Learn basic methods such as binary search, sliding window etc. and start B ladder.
5. Solve B Ladder till you feel confident enough to move to C, there is no need to complete it.
6. Before solving C ladder, learn DP and solve basic problems of the CSES DP set, learn graph from YouTube playlists and solve problems from the practice set of GFG, then try CSES set.
7. In graphs don't learn advanced algos, instead learn BFS, DFS, Dijkstra, connected components, cycle detection etc.
8. C ladder must be completed entirely. But don't stick on a problem for too long, mark it as unsolved and move to next. Come back to it later.

Give all regular codeforces contest, div 4, div 3 and div 2. Give all atcoder's ABC and ARC. Give all codechef starters, cookoff and lunchtime.



### **While practicing problems hide problem tags in cf.**

DP is going to be tricky to learn, try the following playlist(It contains codes as well!) or you can also try looking at GFG DP problems. Do not use cp-algorithms for learning or practicing DP. Try doing most of the CSES DP problem set.

[https://youtube.com/playlist?list=PLgUwDviBlf0qUlt5H\\_kiKYaNSqJ81PMMY](https://youtube.com/playlist?list=PLgUwDviBlf0qUlt5H_kiKYaNSqJ81PMMY)

An alternative playlist for Dp is :

[https://www.youtube.com/playlist?list=PL\\_z\\_8CaSLPWekqhdCPmFohncHwz8TY2Go](https://www.youtube.com/playlist?list=PL_z_8CaSLPWekqhdCPmFohncHwz8TY2Go)

However , note that the videos in this playlist do not contain Codes of the problems discussed. They are more focussed on how to build the intuition for problem solving and writing a pseudo code for the same.

Try to upsolve problems from contest.

Look at editorials if a problem doesn't click you at all or have given your best effort. Also look at editorials of problems you have solved, look if the editorial approach is faster, or some niche trick has been used. Don't be afraid if you can't solve multiple problems in a row, keep trying.

To practice more contests on Codeforces(when there is a long gap between contests or you want to do time-based practice) , you can try creating your own Contests in Codeforces Gym as well or you can solve virtual contests.

Once you are confident you have built a pretty good grasp on the basics, you can try learning the following topics :

TRIES:(3 videos in a playlist)

<https://www.youtube.com/playlist?list=PLDzeHZWIZsToGppbCLGKiYI-gTVASNEVb>

Segment Trees: (2 videos)

<https://www.youtube.com/watch?v=-dUiRtJ8ot0>

<https://www.youtube.com/watch?v=rwXVCELcrqU>

Fenwick Trees: (2 videos)

<https://www.youtube.com/watch?v=DPiY9wFxFIw>

[https://www.youtube.com/watch?v=NQykDuH1\\_OY](https://www.youtube.com/watch?v=NQykDuH1_OY)

DP with bitmasks:

[https://www.youtube.com/playlist?list=PLb3g\\_Z8nEv1icFNrtZqByO1CrWVHLL05g](https://www.youtube.com/playlist?list=PLb3g_Z8nEv1icFNrtZqByO1CrWVHLL05g)



Note that these topics are relatively uncommon in tests, so you might consider skipping them if you are short on time.