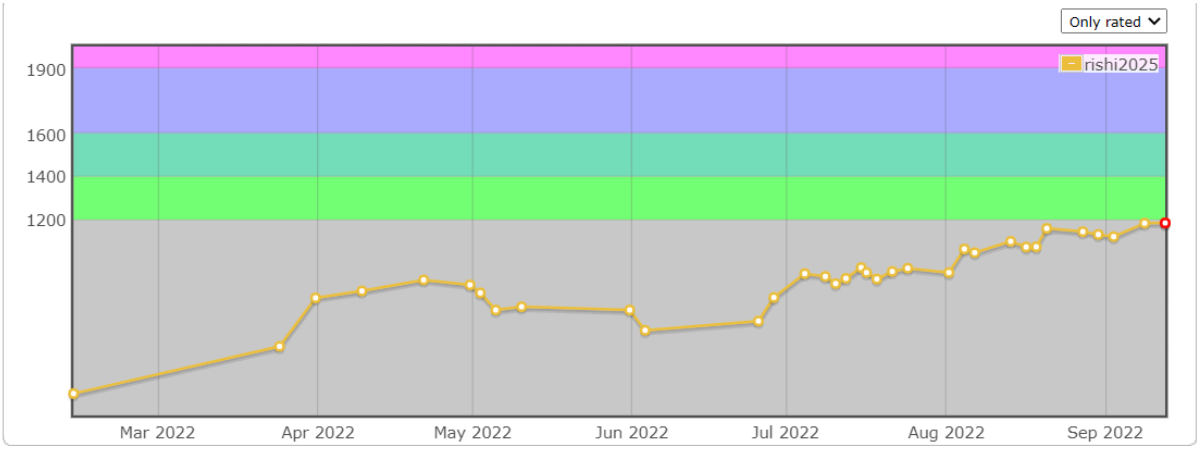
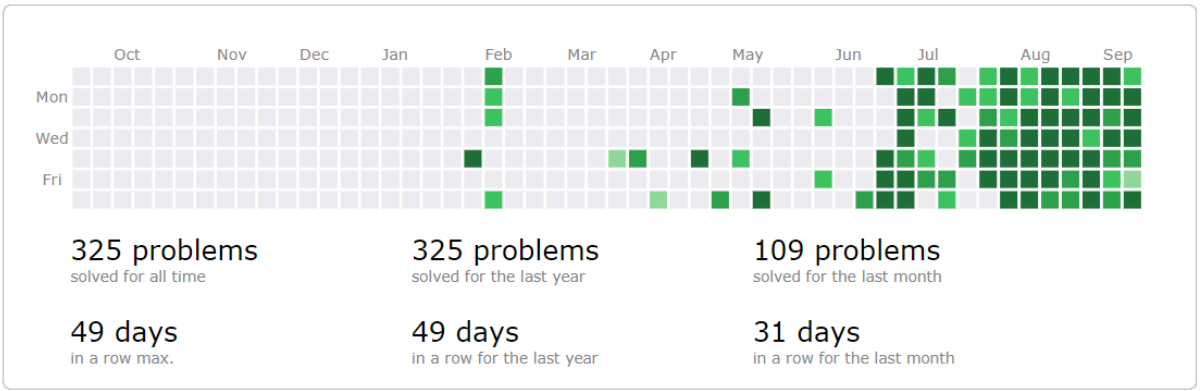
I come across folks who seek guidance for progressing in cp and most of them are pupils / newbies, because when you reach an expert you yourself know what has to be done to progress further . So for those here is some advice.

So let's start with

People with rating 800 - 1200





If your graph looks something like this , you are at level 1 , your first goal should be reaching specialist -> 1400 rating .

Topics needed to cover **level 2 (>=1400)** in order of their importance.

0) stl in c++ / equivalent .

1) **Basic maths , basic logic**
2) **Searching - Sorting , greedy , two pointers ,sliding window**
3) **Implementation , stl based**
4) **Binary search (very important and a saviour)**
5) **Number theory (sieve , gcd , factorisation , prime numbers etc)**
6) **Basic bit manipulation and basic combinatorics. (very basic)**

Practice strategy

1) Two types of practice are needed
2) 1) to build critical thinking / problem solving / application of concepts known.
   2) to learn new concepts, tricks and standard problems.
3) For the first kind of practice you have to solve 40 each in the rating range 800-1400. You can skip a level if you are way too comfortable with it , but **don't do more than 50 of any level** and waste your time , **most beginners do 100-150 800's which are completely wasted** as in the above profile, the person has done 100 800's .
4) So do 40 each 800->900->1000->1100->1200->1300->1400.
5) Now at this point you would be able to do A,B and C sometimes.
6) Now do 10 more of each if you want .
7) Now if you do this you will reach near 1400 .

Learning strategy

1) You can learn on the fly by just reading editorials and solutions , but it would be more helpful if you spare some time for learning new things.
2) For the topics listed above just search on youtube or google about what are those algos now for doing some standard problems. **Resource is Cses Problemset - those are educative and standard problems.**
3) **Competitive programming handbook - very soothing book .**
4) **Luv's youtube channel has a competitive programming playlist** Which is also a nice resource to learn things from beginners' pov .
5) You can actually follow any thing you wish just search on google / yt try reading/ watching if you don't get ,-> switch , after two three tries , you will find something which works for you .

Bottomline - practice consistently and 40/50 of each rating till 1400 and learn some standard algos , tricks .

From level 2 (1400) to level 3 (1500-Expert) .

Topics :

1) **Strong implementation , speed , accuracy and brute force.**
2) **Binary Search(very important),**
3) **Number Theory ,**
4) **Bit manipulation and Combinatorics (basic - intermediate)**
5) **Trees / graph -> basic (questions <=1600/1700 rated)**
6) **Dynamic Programming (1600-1700) , standard problems.**

Practice strategy

1) For first kind of practice
    1) now you need to practise 40 each in 1500 - 1600
    2) Then 10 each of both .
    3) Now if you have 50 of each till 1600 begin with 1700 , 1800
    4) Strategy remains the same 40 each then after 10 more for each .
2) **For second kind - Learning strategy**
    1) Learn binary search on answer and everything related to binary search
    2) Do some good problems on bit manipulation and combinatorics as you encounter them.
    3) Start learning dp and trees side by side or one after from youtube or wherever you wish (I started trees from codencode and dp from Aditya Verma) .
    4) Can refer Cses for standard problems.

Bottomline:- If you do this you will reach or have potential to reach expert(>1600) .

3) For people being specialist and want to reach Expert

- You have to do 40-50 each till 1800 .
- When you complete 1800 then you might reach Expert soon ,
- I take it like this - you should solve <=x+200 in problems set to reach a rating x in contests .
  Topics to be covered - Trees ,Graphs, Dp , Segment trees/Range query data structures.
  Learn various useful graphs algos like mst, toposort etc , you should be doing plenty of problems on dp , you should learn segment trees and solve questions on it.
  So all in all you have to master these topics and come to intermediate or above intermediate level .

Resources- Cp algorithms , Competitive programming handbook, And tbh any good tutorial , web or video based which you like , doesn't matter much .

Necessary points to keep in mind :

1) **Participate in every contest** , don't care about ratings till you reach an expert. **Learnings in one contest = learnings for one week in practice** .
2) **Always upsolve the problem you couldn't do** (<=C in cf div2 rounds) and watch others' solutions who are better rated / your idols -how , why they did that problem in that way ,(**lol just stalk their solutions**).
3) **In a month if you do 100 problems it's great** , you should tap on your shoulder. That costs like 3 problems a day, very much doable if consistent.
4) **If you feel uncomfortable with the new rating don’t back out, it's fine .** When I moved to a new rating I did the first 20 mostly by reading editorials, taking hints etc . You can do the same but just understand the intuition behind it .Also you will come across some problem where you won’t understand anything even after reading editorial and trying your way out . In such a situation , leave that problem for the future or ignore it for sometime but don’t waste your day on that one problem .
5) **Make a habit of learning some algo** or doing some standard tasks like cses too.

6) **Be consistent , be confident , and perform in contests** and if couldnt analyse your mistakes.
7) One more golden advice in each question is to try to extract some conclusions/ some facts / some standard knowledge which you can use in other problems too. For example median is the point which has the least sum of difference from n points . Anything, the bottomline is that from each problem you should try to extract some takeaways which you can use in other problems too.
8) **Uplift yourself no matter what** , even if you are on bottom of rank list just know that , **you are still the best** , don't be too harsh on yourself because-there is whole world against you and at the end of the day It has to be you on your side , only you are there to pamper yourself so be easy on yourself , just learn from your mistakes and remember everyday is a new day .
9) Save the algorithms or implementation in files or snippets so that you can revisit , for example I have many files named as , graph_algorithms, dp , utility functions , string algos, range based data structures etc. So when I forget something I revisit that also I can use codes in contests.
10) Who am I ? https://codeforces.com/profile/xinyster .