3-2. Practice Skills – From Rating 1400 to 1900

One of the most major rating range in Codeforces is [1400, 1500]. They are wanting to improve rating, but from rating 1500 is difficult and many people gave up here. But there are also many people who practiced without giving up and gained rating.

To be rating 1900, skills as follows are needed:

- You know and can use major algorithms like these:

Brute force	DP	DFS	BFS	Dijkstra
Binary Indexed Tree	nCr, nPr	Mod inverse	Bitmasks	Binary Search

Note: I think that segment tree is not needed to be rating 1800. I knew segment tree when I was a purple-rated coder.

- You can code faster (For example, 5 minutes for R1100 problems, 10 minutes for R1400 problems) Fast coding is very important for Codeforces because generally, if there is a wide gap of difficulty in a contest, fast coding affects rating very much.

[[How to Practice]]

If you are not good at fast-coding and fast-debugging, you should solve AtCoder problems. Actually, and statistically, many Japanese are good at fast-coding relatively while not so good at solving difficult problems. I think that's because of AtCoder.

I recommend to solve problem C and D in AtCoder Beginner Contest. **On average**, if you can solve problem C of AtCoder Beginner Contest within 10 minutes and problem D within 20 minutes, you are Div1 in FastCodingForces :)

If you are not good at solving more than R1400 in Codeforces, you should learn some algorithms above and solve typical problems in Codeforces. For example, if you felt that you are not good at DP, solve DP-tagged problem in Codeforces which is R1200-R1400. Surprisingly, there are only ~50 problems which tagged DP and less than or equal to R1400.

Interestingly, typical problems are concentrated in Div2-only round problems. If you are not good at Div2-only round, it is likely that you are not good at using typical algorithms, especially 10 algorithms that are written above.

11

If you can use some typical problem but not good at solving more than R1500 in Codeforces, you should begin TopCoder. This type of practice is effective for people who are good at Div.2 only round but not good at Div.1+Div.2 combined or Div.1+Div.2 separated round.

Sometimes, especially in Div1+Div2 round, some problems need mathematical concepts or thinking. Since there are a lot of problems which uses them (and also light-implementation!) in TopCoder, you should solve TopCoder problems.

I recommend to solve Div1Easy of recent 100 SRMs. But some problems are really difficult, (e.g. even red-ranked coder could not solve) so before you solve, you should check how many percent of people did solve this problem. You can use https://competitiveprogramming.info/ to know some informations.

Unfortunately, I don't know the good website which you can know what problem did you solve in TopCoder SRM, like AtCoder Problems. So, if you want to record what problem did you solve, you should make spreadsheets or tables. This is my brother's spreadsheets which he used two years ago. (Sorry for Japanese, but there are no Japanese in Div1Medium page, so you can refer to it)

https://drive.google.com/file/d/1mSy9PM4Km8EVv8Lp4nhitorOe2HbAS1e/view?usp=sharing

When I was a blue, I was also very bad at mathematical thinking. When I solve 50 Div1Easys, I became yellow in TopCoder and purple in Codeforces.

If you are good at solving problems but did not perform well in real contests, you should do virtual contests more. Do you know Virtual Contest system in Codeforces? You can do virtual participation!