Q) What is the sheet **requirements**? Should I study algorithms and Data structures?

- ONLY programming skills (e.g. Programming 1 level). It is highly advised to implement 2-3 projects
- NO for OOP
- NO for datastructures, but learn STL (or Collections in Java/C#). It helps alot
- NO for algorithms, the sheet will teach you that in a smooth way
- For C++ guys (and others as guide) - first 18 videos here: https://www.youtube.com/playlist?list=PLPt2dINI2MIZPFq6HyUB1Uhxdh1UDnZMS

Q) How much time do I need to finish the sheet?

- Answer varies from one to another.
- Some trainees are fast through whole sheet
- Some trainees are slow through whole sheet
- Some trainees are fast in early pages, but slow at the end
- Some trainees are slow in early pages, but fast at the end

- The sheet has ~900 problems. Around 60 videos. Problems targets average guys. However, you are ecnouraged to skip problems whenever you could. I expect many guys could skip 20%-30% of the mandatory problems. Make use of the Topics page.

215 problems of level <= 2.5 (avg 20 min per problem)
93 problems of level <= 3.5 (avg 30 min per problem)
270 problems of level <= 4.5 (avg 40 min per problem)
178 problems of level <= 5.25 (avg 60 min per problem)
127 problems of level <= 5.75 (avg 75 min per problem)
53 problems of level > 5.75 (avg 90 min per problem)

215*20+93*30+270*40+178*60+127*75+53*90 = ~700 hours (say max 900 hours)

If you trained in the summer vacation seriously for 2 month (e.g. 10 hours * 30 days * 2 month = 600 hours) + the reamining of the year effort, you could solve the whole sheet smoothly and move to Div2-E level goal

- https://ask.fm/mostafasaad87/answers/144907000290 [adjust to whatever fits with you]

Q) When should I give up and **check the editorials** and solutions?

https://ask.fm/mostafasaad87/answers/144907000290

Q) Got **WA**, should I check directly the test cases?

- No, remember in a real contest you only know your problem status (WA, TLE, ...etc)
- Struggle to find the wrong case by yourself. At least 15-30 minutes.
- Don't keep trying longer, just check the test cases
- If you can write a brute force solution for your problem, write a stress test: Generate random cases and compare the optimal algorithm with the brute force case

Q) What is the **debug time**?

- Once you finish coding and start testing, you verify if the program is working as expected or not
- If not, there are bugs that you need to find to make the program behave as expected. From this moment till getting the program AC = debugging time
- People could debug using 'print statements'. A better way using a debugger
- Check out these 4 videos: https://www.youtube.com/watch?v=DlbQwQEiDW0&t=0s&index=35&list=PLPt2dINI2MIZPFq6HyUB1Uhxdh1UDnZMS

Q) Should I solve **every problem**?

- Generally, preferred, but If you think certain level is easy (e.g. solve it within 15 minutes), then jump a block and so on
- This jumping might be for codeforces problems only

Q) About UVA

- Just started in Div2-A, could I finish its codeforces problems first, then solve the UVA/Colored problems?
- Many juniors find UVA problems in Div2-A hard, I understand
- Yes, almost same for DIv2-B. But don't do that in next sheets as order might matter, because all of such knowledge are mainly preparations for hard Div2-B or Div2-C
- However, following the order is a much better idea
- Similarly, one could finish All Div2-A/Div2-B codeforces problems, then solve their colored problems. Again, this is not the best way.

- Why UVA/Other judge problems?
- Because they have important-to-solve problems that don't exist in other judges

- I can't know my bug in a UVA problem, what to do?
- 1) Google solutions and compare. To find cpp soliutions e.g. write: UVA 10110 filetype:cpp
- 2) Use **udebug** to find test cases of some problems. E.g. https://www.udebug.com/UVa/10110
- 3) UVA used to have very nice forum. Seems not active
- 4) Get AC solution and use it to generate test cases, if easy

Q) Is using **C#** ok?

- Generally yes, but you won't be able to submit in UVA judge, as C# is not supported
- For such problems, write your code, but heavily test it. You may download an internet code and evalaute the test case on both
- On the other hand, learning Basic C++ + STL is not hard for C#/Java guys
- C++/Java/Python are official in UVA
- Codeforces allows more such as Javascript

Q) When I watch a **video**, should I solve the **problems** in its info section?

- No, sheet has subset of these problems already in specific order
- Sheet is self-contained

Q) I watched the **video**, but it is **hard**, any tips?

- Algorithms are hard, learn to struggle
- Watch the video 2-3 times, try to rewrite its code by yourself
- Still can't get it? Google for more materials from the web (ppt/pdf/videos) and try to learn
- In worst case, leave it for now and return to it later

Q) How does your sheet prepare for **ECPC/ACPC**?

- The sheet prepares you to reach level 5-5.5/10 in several categories
- If a team of 3 members solved the whole sheet, they may rank in the top 15 in the contest

But let's go in details. Individual sucess in contests depends on several factors. Let me state some of them.
- Solving many problems of good quality
- Improving your different skills (reading, thinking, coding and accuracy).
-- 2 persons could solve in training the same problem. One got it in 20 min first submission, and the 2nd needed 90 minutes due to 60 minutes debugging.
- Healthy training: Regular / good times for training (e.g. morning) / weekly contests / reading other codes / collaboration with others / etc
- Stress management during contests
- Emotions management when fail in solving or feel performance is not improving enough
- Avoiding Psychological issues: Comparing to others, Negative feelings, Your image, Regretting trainging time

Moving toward a team contest, you need more concerns:
- Serious team members. If only one active member, they may end up in bad performance. So EACH team member need to finish the sheet individually + weekly contests
- Tolerating team mistakes during the contest
- PC management
- Suitable strategy + several team contests to tune it

As you may notice, there are MANY factors for sucess.
- This sheet provides you with high quality problems and good topics distribution + way to record your stats to know your weak points
- However, there are many concerns that YOU have to tackle by yourself and your team members

- Finish up to CF-C2 sheet, then study from the "Cracking the Coding Interview: 150 Programming Questions and Solutions" book
- Also watch: https://www.youtube.com/watch?v=39vqarATPyM

Q) How different is your sheet versus **Ahmed Aly Ladders**?

- Ladder problems are selected automatically, no personal investigation for the actual benefit/need from the problem
- Mine is mixed between automated and manual.
- At the current moments, many of my trainees and students feedback, I am aware of the problem level and its category.
- I updated the sheet many times because of the received feedback
- My sheet involves the algorithms videos to learn, in order, while you grow up.
- I selected videos to prepare you as soon as possible for Div2-C/Div2-D where many algorithms starts to appear
- It is a sheet..ready for you to record your times, notes...etc...this help to improve yourself
- It is not blocking style. If you can't solve problem, just leave it and move to other one. In ladders, you see next problem when solve current one (or do workarounds)

Q) How did you **select problems** for the sheet?

- Long story, many versions were there, from a version to another improvements were applied
- Codeforces problems where rated based on this CF tool: http://codeforces.com/blog/entry/46304?mobile=true#_=_
- Any rating is just an estimation. I found this one a pretty reasonable measure
- The videos are selected such that when comes to Div2-C/, you are ready
- Manual selections and investigations for non-CF problems to be used in the sheet
- Lots of manual efforts and investigations and feedback processing

Q) what is the next step **after finishing** your sheet ?

- Joining directly my ICPC semi-seniors supervision, **BUT**
- Email me with your online sheet copy link and it **must** have
- Each row should have: code link, time details, problem level, category and comment per a problem
- I will review and decide
- Side note: If you started in Div2-C1 and solved first 15 problems, you can share the sheet with me to follow your updates

Q) can't **access** the sheet in **edit** mode?

- Don't download the sheet, Work over it online "better"
- Can't edit it? Because it is read-only. Read below notes.
- Just make a copy to your google driver
- Then work over it online. Following are the details

- Login to ur google Gmail
- Go to my sheet
- In the sheet click on 'file' menu
- select Make copy
- it will create copy for u
- RENAME it to Junior Training Sheet
- Now the copied sheet is opened for you (or go inside ur Google drive and you will find it)

NOTE: If u did so and still read-only format, then you are again opening my sheet (e.g. with old name), NOT your copy

Q) What to write in the **category column**?

The algorithm used to solve a problem. In Div2-A, this might be:

- This column is for the algorithm you used during solving. Usually, new guys in CF-A are confused. If so, leave it CF-A and start to write in CF-B
- The more you go in the sheet you will learn algorithms (e.g. Binary search, DP, DFS, etc). Then this what you write in level column
- The problem that has no algorithm but a specific idea called ad-hoc, This is the case for most of CF-A and less later
- Implementation: Means the problem request is almost direct, just code it
- Brute Force: Means instead of finding elegant solution, try all possible solutions (e.g. 3 nested loops) and select the solution
- Ad-hoc: Just per-problem thinking in a special way/analysis on how to solve the problem
- Please watch from this minute: https://youtu.be/DZ6YTtILCE8?t=839

Q) Are problems really sorted based on **easiness**? I don't feel so.

They are sorted by easiness already. But, whatever order, anyone will find some are easy and some are hard in some order.

That is, no one can give you a list that every problem for YOU is easier than the next problem.

In other words, If we gave 100 problems to 10 students of same level to solve and told them rank from easy to hard, they will rank them differently.

So, questions ordered by people average. The promise is, the problems will be within your range to solve.

Q) What are these problems **colors**?

See "Problems Colors" notes in info page

Q) Are the problems **sorted**?

Yes, but this is tricky as sorting is subjective.
That is imagine 10 prblems given for 100 people to order based on its level, people will arrange in different ways based on theie experience

So if you felt they are not sorted, just keep going

Q) Why problem-solving is that **important**?

See the first 2 videos here: https://www.youtube.com/playlist?list=PLPt2dINI2MIaNcU070HIAO8JWYBcafuyG

Q) I **feel bored** when solving problems compare to doing projects?

https://ask.fm/mostafasaad87/answers/145333554402

Q) I would like to **freeze** my study for 1-2 years to be good in problem-solving?

I never liked that. Graduate on time. In your free times and vacations do more problem solving
Relevant: https://ask.fm/mostafasaad87/answers/145151822818

Q) Topics based-training vs Blind Order

In topics training, we study a topic, then solve a lot of problems over it.

Advantages:
- Mastering the algorithm till solving some hard problems in short time

Disadvantages:
- Discovering the algorithm behind the problem is an important skill. Given that you know the topic, you lose a good space to improve this skill
- Being in the mode of specific algorithm lets you solve many of it easier. However, when solving in real contests, your mind is not so active on specific topic

In my sheets - Blind style:
- You solve 3-5 per topic. Then you have to discover the other problems by yourself. So you train to avoid the missing 2 points

Claim:
- Although topics training let guys be so good early, they level stuck early and they don't improve. Seems to me, topics training is an important factor in doing so. Meanwhile, if you just target to be good in Div2D level in shorter time and no interest in further competitions achievments, you may go topics based.

Q) Who FInished my sheet? Their levels?

https://ask.fm/mostafasaad87/answers/150802497762

Q) How to share my sheet progress with you?

https://ask.fm/mostafasaad87/answers/148552940002

Q) What is after the sheet?

- There are 2 other levels, each has around 1000 problems. Semi-senior level and seniors level
- Generally speaking, the region stars will solve a lot of problems, e.g. 2000-3000 problems with many of them of hard level
- Whoever finish the sheet, I join him in my supervision for the next levels