



Coding Essentials and Interview Preperation

Vrishin Vigneshwar

Contents

[CP Topics](#)

[Linkedin Profile](#)

[Github Profile](#)

[Projects](#)

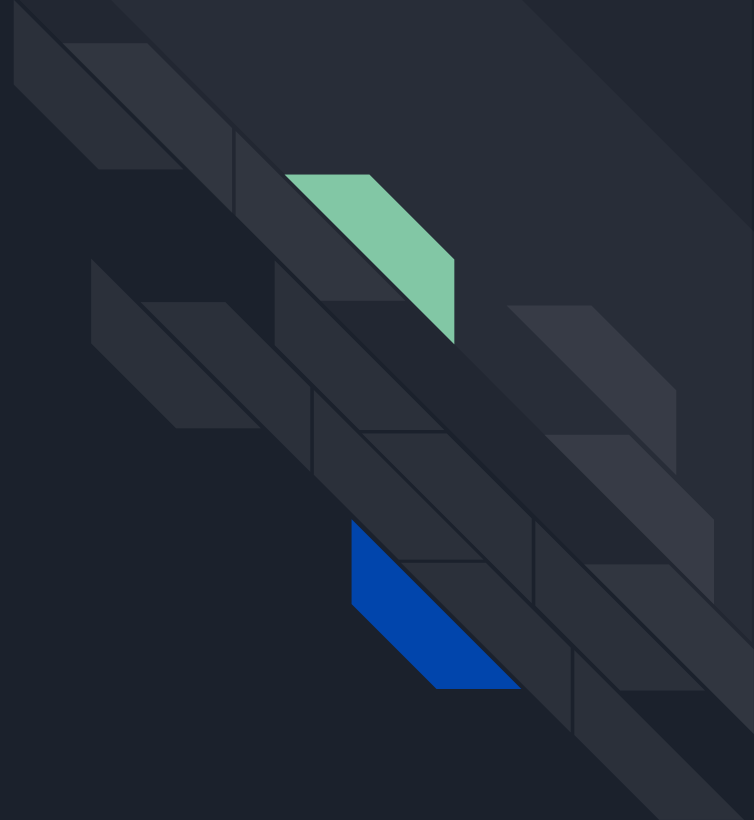
[Resume Building](#)

[Interview Psyche](#)

[RoadMap](#)


[Resources](#)

[FAQs](#)





CP Topics

- Dynamic Programming
 - Prefix Sums
 - Two Pointers
 - Binary Search
 - Trees/Graphs
 - Stacks/Queues
 - String Hashing
 - Miscellaneous/ADHOC
- 



Dynamic Programming

Material (Links/Problems)

- <https://www.quora.com/Are-there-any-good-resources-or-tutorials-for-dynamic-programming-DP-besides-the-TopCoder-tutorial> - answer by Michael Danilak - for beginners<->intermediate
- <https://drive.google.com/file/d/13AHqKC7wkBeCn29kxsWly1jY-bT4vfzU/view>

Notes


- BIT MASK DP - vvv important
- CLASSICAL DP - vvv important

Marquee - advanced DP

Super dream - very basic / classical DP

Dream - don't think it's necessary

Block - don't think it's necessary





Prefix Sums


Material (Links/Problems)

- <https://drive.google.com/file/d/1qwWDD16OJGj74WU5S0Px129xDaT1DK0C/view?usp=sharing>

Notes

- Helps in finding frequency!
- <https://codeforces.com/contest/816/problem/B>
- <https://codeforces.com/problemset/problem/295/A/>

Marquee - 100%
Super dream - 100%
Dream - 80%
Block - 50%





Two Pointers


Material (Links/Problems)

- <https://codeforces.com/edu/course/2/lesson/9>
- <https://drive.google.com/drive/folders/1yEvEaQmAx7cAwNxnMALdc9ENeKTEWY9?usp=sharing>

Notes

- Sliding Window
- Slow Pointers

Marquee - 100%
Super dream - 100%
Dream - 80%
Block - 50%






Binary Search

Material (Links/Problems)

- <https://codeforces.com/edu/course/2>
- <https://drive.google.com/drive/folders/1-OPNVYJhIOOcRdd3Q8n5A3uHujL3fuNn?usp=sharing>

Marquee - 100%
Super dream - 100%
Dream - 80%
Block - 20%





Trees/Graphs(DFS,BFS)

Material (Links/Problems)

- Vast topic - suggest googling and finding your best fit

Notes

- MBFS is a must know - vvv important
- CP algorithm style recursive DFS - simple/ must know

Trees/Graphs Algorithms:

- DFS/BFS
- DSU
- Floyd Warshall
- MST(Kruskal's)
- Dijkstra's
- LCA

Notes

- Finding no of components in a graph
- Find islands in a graph
- Detecting cycle in a graph

Marquee-100%

Super dream - dfs/bfs/dsu - 100 %, MST/djik/fw/lca - 45%

Dream -dfs/bfs/dsu - 70 %, MST/djikstras/fw/lca - 5%

Block - DFS/BFS 70%



Stacks/Queues


Material (Links/Problems)

- <https://www.hackerearth.com/practice/data-structures/stacks/basics-of-stacks/tutorial/> - very basic for beginners
- <https://www.hackerearth.com/practice/data-structures/queues/basics-of-queues/tutorial/> - very basic for beginners

Notes

- Finding nearest to the left whose value is less than mine - variations
- Height based questions - usually Stack
- If you want to left-shift , right-shift an array - deque

Marquee - 100%
Super dream - 100%
Dream - 100%
Block - 100%





String Hashing


Material (Links/Problems)

- <https://codeforces.com/blog/entry/60445>
- <https://drive.google.com/file/d/1murmNz72ZtN5FdZVsoC0kobZ3-L01wDO/view?usp=sharing>
- <https://drive.google.com/file/d/19ZPEy2N3FSEAHtHtiWkhTAxz-6S7UF4u/view?usp=sharing>

Notes

- Finding nearest to the left whose value is less than mine - variations
- Height based questions - usually Stack
- If you want to left-shift , right-shift an array - deque


Marquee - 100%
Super dream - 60%
Dream - 20%
Block - ?






Miscellaneous/ADHOC

Notes (brackets denote marquee.superdream.dream.block)

- Scheduling Algorithms (80, 100,100,?)
 - Two Heaps problem (80, 100,100,?)
 - Fenwick trees/KMP/Segment Trees/Tries/Suffix Arrays (100, 50,0,?)
 - Hair and Tortoise (finding cycle in a linked list) (100, 100,100,100)
 - Design Rounds - YouTube is good for this (40, 30,20,10)
 - Invert Count (100, 100,100,?)
 - Merge Sort/ Count Sort (100, 100,100,?)
 - Linked Lists/ GfGs style - Binary Tree (10, 100000,10000000000,?)
- 




LinkedIn Profile

- Try not to comment "interested" unless it's a verified website
 - Marketing is everything
 - Add links, projects, references
 - Put everything about you into LinkedIn
 - REFERENCES GALLORE
- 




Github Profile

- Push all your code GIT
 - Git is very important - it's literally proof of your work
 - Git is considered a skill by many companies
 - Marketing here also very important
- 




Projects

- Very important from interview perspective
 - Good to have 1-2 really good projects on resume - adds lot of value to resume
 - Ideal project
 - Visual appeal 10/10
 - Likely something the user can play with
 - Shows depth of research and work put in (doesn't mean it has to be a research project)
 - One idea is to convert every need to learn a new language/ framework/ concept ---> project
- 




Resume Building

- Keywords
 - Scholarship
 - Hackerrank
 - Project Euler
 - "Master"
 - Difference between beauty and necessity - when to use what
 - Strictly ONE page
- 




Interview Psyche

1. KEEP TALKING - DO NOT BE QUIET"
 2. Catch out on deliberate question gaps - ask all details until you are satisfied
 3. Look out for cues from the interviewer - they will always try to help you
 4. "do you have any questions?" - have a good question to ask - do not ask for performance review
 5. Get to know about the company
 6. Solutions should build up one by one, don't give the best solution even if you get it in the start itself
 7. Know your projects well
 8. Its always great to take a mock interview with a friend before an actual one
- 



RoadMap - 1st Years


1. <https://a2oj.com/Ladders.html> - ID 11-16, 22-26
2. <https://codeforces.com/> - all contests
 - a. Div3/Div2 ---> Focus
 - b. Up-solve ---> Solve 1 unsolved question after every contest
3. <https://atcoder.jp/> - all contest
 - a. ABCs ---> Focus
4. Projects can be focused later down the line
5. Learn touch typing - consistent effort -> TOO HIGH reward
6. Participate in symposiums (jan - march)

- Target **1500+ questions** towards the end of your journey(1 ~ 1.5 years from now)
 - **10-15 problems/day** <-----> **100-200 problems/month**
- 



RoadMap - 2nd Years

1. <https://a2oj.com/Ladders.html> - ID 11-16, 22-26
2. <https://codeforces.com/> - all contests
 - a. Div3/Div2 ---> Focus
 - b. Up-solve ---> Solve 1 unsolved question after every contest
3. <https://atcoder.jp/> - all contest
 - a. ABCs ---> Focus
4. Projects - 2 good projects on your plate will be handy (already covered more on this)
5. Create Resume, LinkedIn, Github
6. Participate in symposiums (jan - march)

- Target **800+ questions** towards the end of your journey(6 ~ 12 months from now)
 - **10 problems/day** <----> **100 problems/month**
- 




RoadMap - 3rd Years

Beginning

1. <https://leetcode.com/contest/> ---> FOCUS
2. Aptitude questions ---> HIGH FOCUS
3. <https://www.interviewbit.com/>
4. DBMS/OS brush up

Later


5. <https://a2oj.com/Ladders.html> - ID 11-16, 22-26
6. <https://codeforces.com/> - all contests
 - a. Div3/Div2 ---> Focus
 - b. Up-solve ---> Solve 1 unsolved question after every contest
7. <https://atcoder.jp/> - all contest
 - a. ABCs ---> Focus
8. Projects - 2 good projects on your plate will be handy (already covered more on this)

- Target **400+ questions** towards the end of your journey(3 ~ 4 months)
 - **5 problems/day** <----> **75 problems/month**
- 



RoadMap - 4th Years

Beginning

1. <https://leetcode.com/contest/>
 2. Aptitude questions ---> HIGH FOCUS
 3. <https://www.interviewbit.com/>
 4. Design rounds- Youtube videos
 5. Target low-effort high-reward concepts
- 



Resources

For Research/Understanding/Learning

1. <https://cp-algorithms.com/>
2. <https://www.hackerearth.com/practice/data-structures/arrays/1-d/tutorial/>
3. <https://codeforces.com/edu/course/2> (and literally any good Codeforces blog)
4. https://www.youtube.com/channel/UCZLJf_R2sWyUtXSKiKlyvAw
5. <https://www.youtube.com/channel/UCRPMaQdtSgd0lpeef7iFsKw>

For interview prep style CP

1. <https://leetcode.com/> - recommend Hard Problems
2. <https://www.interviewbit.com/> - haven't tried but heard great reviews
3. <https://www.hackerrank.com/>

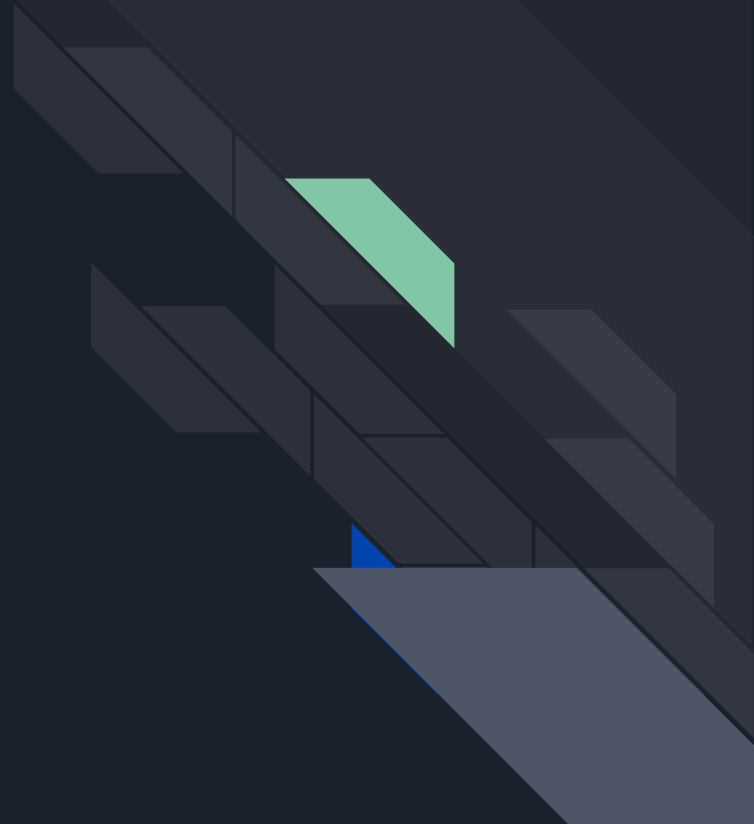
For Hardcore style CP

1. <https://codeforces.com>
2. <https://atcoder.jp/>
3. <https://a2oj.com/Ladders.html>

For Aptitude

1. <https://www.youtube.com/watch?v=lxm6ez2cx6Y&list=PLjLhUHPsqNYnM1DmZhlbtd9wNhPO1HGPT>
2. <https://www.indiabix.com/>

FAQs



1. How and on what basis to choose a programming language
 - a. time
2. Is it fine, on knowing only one programming language to attend any IT interviews?
 - a. Usually fine
3. What are the fields in CS that can be explored? Keeping the future in mind.
 - a. IOT
 - b. ML/DL/AI
 - c. Cloud computing
 - d. Big data and stream processing
 - e. Distributed Computing
 - f. Block Chain and crypto mining
 - g. Data science and data visualization
4. How to arrive at the solution of the problem at a very limited time period?
 - a. practice
5. Is medium level questions in leetcode enough for Marquee companies. Any idea.?
 - a. I would recommend hard
6. I am new to coding. Am I late / Can I ace the interview if I start now (3rd year)
 - a. If you meant, intern drive, then for marquee it might be a bit difficult but, you should find yourself comfortable with the rest
7. Do we need to have knowledge about about competitive coding stuffs for placements?
 - a. Marquee - definitely
 - b. Super dream - mostly not needed
8. How useful will it be? - lol
9. Being in EEE dept, I don't do any projects or interns but a good knowledge in data structures and algorithms. How is the possibility of me getting into a good software job?
 - a. Again im not very qualified to make a good guess here. But Ill try
 - b. If you have a sound understanding of aptitude -> you can make it to interviews -> after that it depends all on how the interview goes, and how you showcase your strengths and reason out your weaknesses
10. What are the basics of coding which a Mechanical Engineers must learn?
 - a. Basics of coding like, loops, conditionals, IO, functions -> basics of problem solving (leetcode easy)
11. How to begin practicing CP?
 - a. We have covered this in the coding club, too many times, we have a whole video for this, we will share the same with you
12. Is knowing any one coding language enough before placements?
 - a. I don't think that will be an issue for interviews and coding problems round. But it may have an effect on your resume, that you are aware of only one coding language. You should be prepared to answer an interview question that goes "why do you know only one language"
13. What are the skills companies expect from us especially in machine learning field in the coming future and how do u develop these skill sets in a effective way!!
 - a. Apologies, I haven't dabbled myself in ML at all to effectively answer this question
14. Could you give us an idea about the skills required to enter in a technical and non technical role in a firm ,how to build the same and have a relevant resume.
 - a. Technical - depends on the firm, too vague to answer
 - b. Non technical - community fit, team playing, ability to meet deadlines, ability to stretch if needed, ability to solve unknowns on your own
15. how to prepare for competitions like codejam,kickstart in a short span of time?
 - a. Sorry but I think continuous, consistent effort will be necessary
16. Best platform to learn c programming for beginners, how to code placement related questions, span of time to learn the language and about dbms
 - a. Id suggest c++ instead.
 - b. Id suggest any blog/ youtube playlist/ udemy course for learning a language
 - c. If you know how to IO/ functions/ conditionals - you are good to go

- d. Max spend 3 days in learning a language for CP - other technicalities learn as you go
 - e. DBMS understanding will always help passively in many ways
17. How to crack Google
 - a. All the things we have discussed till now hopefully answers this question
 18. How long will it take to complete DS, if I start now with basic knowledge on it, and how should I fast forward the process so that I could complete and at least be ready with the concepts in 3-4 weeks
 - a. If you start now with basic knowledge, try to solve 10+ question a day, focus on some curated list of classical questions. Example -
 19. Which programming language is easy and effective to prepare placement training?
 - a. Python.
 20. How should we prepare for competitive coding?
 - a. We have a video on that. We will share.
 21. Do companies look for only people with domain specialisation or also people with just good fundamental programming knowledge
 - a. Depends on company interest
 22. How much cgpa we take to get dream companies?
 - a. 7.5+ im guessing - not sure
 23. What qualities are expected from a mechanical engineer when companies like Infosys, TCS, CTS (other software companies) approach them
 - a. Non technical - community fit, team playing, ability to meet deadlines, ability to stretch if needed, ability to solve unknowns on your own
 - b. Technical - basic programming skills
 24. Is it too late to start competitive programming now? (2020 - 2024 batch) - daai athukula too late aa xD
 - a. 92848209480979035725% NOT TOO LATE xD
 25. Is it enough to learn deeply one c language?
 - a. Doesn't matter if you are god level in a language
 - b. It matters if you are god level in applying that language
 - c. For example you can try to expertise yourself in applying C++ to CP, JS -> rest APIs etc
 26. What are the some important things to note and keep in mind before entering the interview?
 - a. Hopefully answered by now
 27. What are some of the proven sources or websites to learn and practice competitive programming?
 - a. Hopefully answered by now
 28. Which programming language is mandatory for attending placement?
 - a. Any language is good. Python/c++/JAVA are favourites
 29. Are aptitude questions asked, if yes, then how to prepare for the same?
 - a. HABN
 - b. Give some advice to attend interview
 - c. HABN
 30. What all do product companies look for?
 - a. Again it highly depends on what company we are talking about
 31. What programming languages are most important? Is there a new unpopular but rising language?
 - a. It again depends on what field. In ML/DL/AI -> python. For faster, underlying apps, cp-> C++, for rest APIs -> node JS, etc.
 - b. Google should surely answer that question better than me :p
 32. how to be consistent
 - a. Not sure. Im very inconsistent myself.
 33. What are the basic languages should I know?
 - a. Python should be enough
 34. Which are the most important computer languages to learn
 - a. Depends on what you want to do. As stated in previous questions
 35. About Critical thinking
 - a. Critical thinking is the intellectually disciplined process of actively and skillfully conceptualizing, applying, analyzing, synthesizing, and/or evaluating information ...

- b. From
<https://www.google.com/search?q=critical+thinking&rlz=1C1RXQR_enIN944IN944&oq=critical+thinking&aqs=chrome..69i57.2117j0j7&sourceid=chrome&ie=UTF-8>
 - c. I think this comes with time and experience.
 - d. Not sure how to answer this question :P
36. How much coding skills does a eee student expected to have?
 - a. Basic programming skills -> leetcode easy level for software companies
 37. How many hours on an average per day should anyone spend to prepare for any interview?
 - a. If your interview is in 1 month. Why not spend entire days?
 38. Can non-computer branches get placed in product based companies, if yes what has to be done
 - a. Short answer YES. Long answer, HABN
 39. Is it necessary that my program will have to satisfy every test case asked , in order for me to progress to the next round? Or will it be overlooked ,if I miss out on certain edge cases?
 - a. Depends on company and how they set problems. Usually partials is allowed
 40. What is the max rating one should aim for in codeforces to ace cp round in big companies
 - a. CM + is a good target
 41. People say that you have to learn C, another groups says python.. I don't know which one to learn..
 - a. I'd definitely suggest C++ over C. Python is good as well.
 - b. If your use case is hard core cp -> ONLY C++
 - c. If your use case is interview prep -> python should be enough
 42. Do they have any higher preference for a particular coding language in the interview process ?
 - a. Not to my knowledge
 43. Does preparation for competitive programming cover preparation for interviews as well?
 - a. 85% yes
 44. how to tackle the programming questions that are asked during placements
 - a. Have programming questions knowledge
 45. How to start the preparations to get placed in FAANG companies?
 - a. CP CP CP
 46. What to do if we have no idea for a particular question
 - a. If you can elaborate, if this is during practice or interview, I can answer this question
 47. Programming Languages to know thoroughly? - HABN
 48. How to come up with a solution for a question that we have kind of no clue? - if you have no clue, its gonna be difficult, better learn from solution and keep in mind about this new concept
 49. Which platform is best for practicing or for last day preparations?
 - a. leetcode
 50. Any courses for us to prepare
 - a. I'm not aware of a good course :(
 51. How useful is learning STL if you don't have enough time for preparation?
 - a. Very very very useful if you code in c++. Just finish it off in 2hrs max in hackerrank (STL Proficiency)
 52. Which is better? Java vs C++
 - a. Hehe for app dev, JAVA, for cp C++. Depends on use case
 53. How long does it take for core branches like Mech,Civil and etc. to become as competent in coding as a CS or an IT student ?
 - a. 100% not long. Same effort in my opinion. Any one at any point of time can become good at programming given the time and effort. Trust the process and go for it :)
 54. What do you do when you can't solve a problem?
 - a. Learn from edit