Level-1 Term-1

Tentative Class Schedule:

Week	Topic
1	Variables, Data Types, Scanf/Printf, Format Specifier
2	If-else, Nested If-else
3	Loop Basics: While, Do-while, For-loop
4	Nested Loop, Break, Continue
5	Loop Advanced: Loop+If-else
Online Contest-1	
6	Array: 1D, 2D
7	Character Array, String
8	Function, Recursion
9	Binary Search + Others
10	STL Basic
Online Contest-2	

Guidelines:

- Get complete idea about C language
- Try to implement what you think
- Must participate in Long-Contests (Upto 5 days) that will be held after each class
- Take help from your Batch-mates, Seniors and Teachers if needed

Expectations:

- Able to implement your idea
- Familiarize with all online judges
- Read Tamim Shahriar Subeen's Book completely
- Solve all the problems of LightOJ beginner's category
- Total Number of problems solved: 200+
- CF rating:
  - -At the start: 1000+
  - -At the end: 1200+

Tips:

- Learn about different good resources as well. Geeksforgeeks, emaxx, competitive programming 3, Mahbubul Hasan Shanto vai's book on programming contest, Shafaet Ashraf vai's blog. These are good, but not just these-there are others. Google is your friend.
- Spend around an hour or two at each problem before searching for solutions online. Not more than that, because what is important for you at this stage is to solve more and more problems, and if you spend all your day at one problem, your overall solve count will be low and you will not learn much. But make sure that you spend this time wisely-when we say we have worked for an hour, what happens in reality is that we have worked for just 20 minutes in total, and the other 40 minutes have been wasted away via other 'useful' procrastination. Perhaps learn to adopt the Pomodoro method.
- After seeing the solution to a problem, make sure to understand the solution fully and absolutely. There should be no doubt regarding anything with the solution, and if there is, that means you have not understood it fully/internalized it. Problem solving is more about pattern matching, and being able to match an unseen problem with problems that you have seen before-but if you haven't internalized the solution to the problems that you couldn't solve by yourself, you have then actually just memorized it, and memorized stuff doesn't actually stay in your mind for long-and you won't be able to recognize patterns later as well.

Tutorials & Problems List for Level-1 Term-1:

Week 1 (Variables, Data types, Scanf/Printf, Format Specifier):

Tutorials:

1. http://cpbook.subeen.com/2011/08/data-type-input-output.html
2. https://codeforwin.org/2015/05/list-of-all-format-specifiers-in-c-programming.html
3. https://www.geeksforgeeks.org/data-types-in-c/
4. https://www.youtube.com/watch?v=e9Eds2Rc_x

Problems:

Codeforces: 1A, 99999100

Atcoder: ABC153A, ABC148A

Hackerrank: Playing with Characters

URI: 1001,1002,1006,1015,1019

Week 2 (If else, Nested If else):

Tutorials:

1. https://www.geeksforgeeks.org/decision-making-c-c-else-nested-else/
2. http://cpbook.subeen.com/2011/08/conditional-logic.html

Problems:

Codeforces: 122A, 263A

Codechef: LEAP, EO, ASET002

Atcoder: ABC152A, ABC065A

URI: 2582

UVA: 11172

Hackerrank: Conditional Statements in C

Week 3 (Loop Basics: While, Do-While, For-loop):

Tutorials:

1. http://cpbook.subeen.com/2011/08/loop.html
2. https://www.geeksforgeeks.org/loops-in-c-and-cpp/

Problems:

DimikOJ: Problem 6, Problem 3, Problem 5

Timus: 1149

LightOJ: 1000, 1022, 1216

Codeforces: 99999123, 791A, 977A

Week 4 (Nested Loop, Break, Continue):

Tutorials:

1. https://www.geeksforgeeks.org/nested-loops-in-c-with-examples/
2. https://www.geeksforgeeks.org/difference-between-continue-and-break-statements-in-c/

Problems:

CodeChef: BICBPAT1, PPATTERN

URI: 1189, 1190, 1183, 1435, 1478, 1186, 1188

Hackerrank: Printing Pattern using Loops

Week 5 (Loop Advanced: Loop + If-else):

Tutorials:

1. http://cpbook.subeen.com/2011/08/loop.html

Problems:

Codechef: LEAPY

Atcoder: 079A

Timus: 1083, 1209

LightOJ: 1001, 1015, 1053, 1008

URI: 1557, 1187

Week 6 (Array: 1D, 2D):

Tutorials:

1. https://hellohasan.com/category/data-structure/array
2. http://www.fredosaurus.com/notes-cpp/index.html (See the Arrays section)
3. http://www.shafaetsplanet.com/planetcoding/?p=1388
4. https://www.hackerearth.com/practice/data-structures/arrays/1-d/tutorial/ (try to solve some problem from this problem section)
5. https://www.quora.com/How-do-I-visualize-multidimensional-arrays
6. https://www.geeksforgeeks.org/array-data-structure/
7. https://www.cs.cmu.edu/~rjsimmon/15122-f14/lec/04-arrays.pdf

Problems:

UVA: 10038 (Jolly Jumpers), 414 (Machined Surfaces), 665 (False coin), 467 (Synching Signals), 10703 (Free spots), 10855 (Rotated square), 10920 (Spiral Tap),11349 (Symmetric Matrix), 11581 (Grid Successors), 12187 (Brothers)

Codeforces: 1213B, 1265B, 1279C

Codechef: SALARY

Week 7 (Character Array, String):

Tutorials:

1. https://www.geeksforgeeks.org/strings-in-c-2/
2. https://www.programiz.com/c-programming/c-strings
3. https://www.w3schools.com/cpp/cpp_strings.asp
4. http://www.cplusplus.com/reference/string/string/
5. https://www.tutorialspoint.com/cplusplus/cpp_strings.htm

Problems:

Codeforces: 71A, 1146A, 118A, 1139A, 1223B, 1243B1, 448B, 1151A, 165C, 1243B2, 1268A, 1153C

Week 8 (Function, Recursion):

Tutorials:

1. https://anindyaspaul.com/blog/2015/12/25/recursion/
2. https://hellohasan.com/tag/recursion/
3. https://sites.google.com/site/smilitude/recursion_and_dp

4. https://www.hackerearth.com/practice/basic-programming/recursion/recursion-and-backtracking/tutorial/

Problems:

UVA: 624, 167, 539, 729, 750, 628, 10276, 11085, 574, 524, 10063, 10503, 301, 331, 193, 129, 208, 416, 861, 1262

Week 9 (Binary Search + Others):

Tutorials:

1. https://codeforces.com/blog/entry/9901 (Implementation variations)
2. https://www.topcoder.com/community/competitive-programming/tutorials/binary-search/
3. https://www.hackerearth.com/practice/algorithms/searching/binary-search/practice-problems/
4. https://hellohasan.com/2016/10/20/%E0%A6%AC%E0%A6%BE%E0%A6%87%E0%A6%A8%E0%A6%BE%E0%A6%B0%E0%A6%BF-%E0%A6%B8%E0%A6%BE%E0%A6%B0%E0%A7%8D%E0%A6%9A-binary-search-algorithm/
5. https://www.hackerearth.com/practice/algorithms/searching/ternary-search/tutorial/
6. https://cp-algorithms.com/num_methods/ternary_search.html
7. https://codeforces.com/blog/entry/11497

Problems:

Codeforces: 492B, 474B, 195B, 1221C, 1183C, 371C, 812C, 706B, 580B, 340D, 75C, 448D, 1250J, 616D, 1077D

LightOJ: 1043, 1072, 1307, 1138, 1088,1048, 1137, 1235,1383, 1146, 1240

Spoj: AGGRCOW, ACTIV

Codechef: MCO16503, KGP16G

Week 10 (STL Basic):

Tutorials:

1. https://www.topcoder.com/community/competitive-programming/tutorials/power-up-c-with-the-standard-template-library-part-1/

2. https://www.topcoder.com/community/competitive-programming/tutorials/power-up-c-with-the-standard-template-library-part-2/
3. https://sites.google.com/site/smilitude/stl
4. http://www.progkriya.org/gyan/stl.html
5. https://www.youtube.com/playlist?list=PLgLCjVh3O6Sgux985GYG22xkFt9z9Sq0_
6. https://drive.google.com/drive/folders/1htftK7kjtfnbc0fKZedXLCVrdMP9cVv-

Problems:

UVA: 146, 10107, 514, 10935, 484, 10815, 10954, 400, 10194, 127, 540, 1203, 12058, 855, 11321, 10264, 732, 1062, 10172, 978

Level-1 Term-2

Tentative Class Schedule:

Week	Topic
1	Basic Math, Modular arithmetic, Big-mod, Modular inverse
2	Number Theory: Sieve, Prime Factorization, Number of divisors
3	Recursion, Backtracking, Tail-call recursion
4	Greedy Techniques, Task Scheduling, Kadani's Algorithm
5	Basic DP (Recursive/Iterative dp, knapsack, LIS)
Online Contest-1	
6	Advanced STL
7	Graph Basics: BFS/DFS
8	Shortest Path: Dijkstra, Floyd-Warshal, Bellman-Ford
9	Topological Sort, Tree-Diameter
10	Basic Geometry
Online Contest-2	

Guidelines:

- Participate in online contest
- Upsolve problems after attending online contest
- Try to attend and do well in onsite contest

Expectations:

- Total Number of problems solved: 300+
- CF rating:
  - -At the start: 1200+
  - -At the end: 1400+

Tips:

- Learning how to think is important at this stage. You should have gotten a feel for determining how much time you should give to each problem in order to learn properly-however, we would say do not make it less than 30 minutes, and definitely not more than 90 minutes. But make sure that you have utilized that time of thinking fully-many of us do not utilize this time fully and do not realize it. Learn to think in a IDA* approach, that is, realizing all problems have an easy solution, and can be solved in a few steps. If you find your train of thought getting

too complex, or taking too many steps/difficult to visualize, then it is likely not correct, and if it is, it is most definitely not the intended solution. Learn to abandon current train of thought as soon as you realize that it may be getting too complex for you-these problems are made for humans by humans, not for Einsteins by Einsteins.

- You should have also realized that this **competitive programming business is no joke**. The future world finalists are currently working very hard, and while you are watching movies/browsing the net, they are currently solving problems at their ACM Room. You should spend most of times on thinking about problems. You can also think about problems when eating, bathing or even before sleeping
- Also, do not pick too hard problems, problems in the difficulty range where you find yourself seeking the solution almost every time. Even if you feel that you have understood the solution fully, you may not have fully internalized it-and possibly even indirectly memorize it. Ideally you must pick problems, that, if they were to appear in a contest, you can almost get the idea, but couldn't get a clear picture of the full solution within contest time.
- And at the minimum, **do every CF contests. Even if there is a CT the next day**. Solve the problems that you couldn't solve the next day. This is very, very, very important. This is called upsolving, and upsolve the next one or two problems that you couldn't solve in contest time. Perhaps leave the others for now.
- Please do look at the official solution (editorials) of problems, even if you have managed to solve one by yourself. You may find new ideas/trick there. Make a habit of learning from other people's code too.
- Look at other people's code to learn new tips/tricks after solving a problem.
- And do hide CF tags. 50% of solving a problem is about identifying the correct category of the problem, and you'll be stunted in this respect.
- Expected Number of Problems solved (including previous semester)- 500 at the very least, more like 1000 if you hope for qualifying for WF in the future.

Level-2 Term-1

Tentative Class Schedule:

Week	Topic
1	Number theory: Extended Euclidean, Euler phi, inverse phi, factorizing n!
2	Graph: SCC, MST
3	Graph: Bicoloring, Bipartite Matching
4	Data Structure: Segment Tree, Lazy
5	Data Structure: BIT, LCA
6	Data Structure: Trie
Online Contest-1	
7	Pattern Matching: KMP, Z-algo
8	DP: LCS, LIS, Matrix-Chain Multiplication
9	DP: Bitmask DP, Digit-DP
10	Geometry: Convex Hull
Online Contest-2	

Guidelines:

- Eat, Sleep, Code, Repeat…

Expectations:

- Total Number of problems solved: 300+
- CF rating:
  - -At the start: 1400+
  - -At the end: 1600+

Tips:

- By now you must have formed stable teams. You must take take part in team contests regularly-at least one every week. Select 2 star contests from CF gym at the beginning, and slowly move on to 3 star contests. Or you may select contests from Timus as well. Discussing unsolved problems after the contest is important too, make sure to solve them. Maintain spreadsheets containing daily log of what one has solved.
- Expected number of problems solved (including previous semester) -800 at the very least. Highly motivated people can go for 1500. You can do it.

Tutorials & Problems List for Level-2 Term-1:

Week 1 (Number theory: Extended Euclidean, Euler phi, inverse phi, factorizing n!):

Tutorials:

1. http://www.shafaetsplanet.com/ (Number theory/ Mathematics part)
2. http://lightoj.com/article_show.php?article=1001
3. http://lightoj.com/article_show.php?article=1002
4. http://lightoj.com/article_show.php?article=1003
5. https://artofproblemsolving.com/community/c90633h1291397
6. https://www.hackerearth.com/practice/math/number-theory/basic-number-theory-1/tutorial/
7. https://forthright48.com/p-cpps-101/ (Number Theory Section)
8. https://crypto.stanford.edu/pbc/notes/numbertheory/
9. https://www.codechef.com/wiki/tutorial-number-theory/
10. https://www.youtube.com/watch?v=gk2MUZc8jTM (BACS - Beginner)
11. https://www.youtube.com/watch?v=5OBZYhJnne0&list=PL2_aWCzGMAwLL-mEB4ef20f3iqWMGWa25
12. https://www.youtube.com/watch?v=tKOjYDdeci4

Problems:

LightOJ: 1007, 1028, 1054, 1077, 1098, 1163, 1024, 1197, 1214, 1333, 1340, 1038, 1151, 1298, 1161, 1234, 1318

ICPCLive: 2158 - Factorial, 879 - Boring Card Game, 3196 - Gaussian Prime Factors, 5381 - TEX Quotes, 3017 - Permutation Primes

Week 2 (Graph: SCC, MST):

Tutorials (SCC):

1. http://www.shafaetsplanet.com/planetcoding/?p=2531
2. https://cp-algorithms.com/graph/strongly-connected-components.html
3. https://www.hackerearth.com/practice/algorithms/graphs/strongly-connected-components/tutorial/
4. https://www.geeksforgeeks.org/tarjan-algorithm-find-strongly-connected-components/

Problems (SCC):

Codechef: CHEFRUN

LightOJ: 1034, 1003, 1168, 1210, 1406, 1429, 1390, 1417

UVA: 247, 11504, 11709, 11770, 11838, 10731, 1229

SPOJ: TFRIENDS, CAPCITY, ADAPANEL, GOODA

DevSkill: DCP-79

Codeforces: 22E

Tutorials (MST):

1. http://www.shafaetsplanet.com/planetcoding/?p=825
2. https://cp-algorithms.com/graph/mst_kruskal.html
3. https://cp-algorithms.com/graph/mst_kruskal_with_dsu.html
4. http://www.shafaetsplanet.com/planetcoding/?p=692
5. https://cp-algorithms.com/graph/mst_prim.html
6. https://cp-algorithms.com/graph/second_best_mst.html (Second Best MST)
7. https://cp-algorithms.com/graph/kirchhoff-theorem.html (Kirchhoff's Theorem (To Find number of spanning Tree))

Problems (MST):

SPOJ: MST, ULM09, IITKWPCG

UVA: 544, 10034, 11228, 908, 11733, 10369, 10462

LightOJ: 1002, 1029, 1041, 1040, 1059, 1123

Codeforces: 1095F, 160D, 125E

Week 3 (Graph: Bicoloring, Bipartite Matching):

Tutorials:

1. https://cp-algorithms.com/graph/bipartite-check.html (Bipartite Checking)
2. https://brilliant.org/wiki/matching-algorithms/
3. http://shakilcompetitiveprogramming.blogspot.com/2013/12/maximum-bipartite-matching-by-kuhns.html (Maximum Bipartite Matching by Kuhn's Algorithm)
4. http://zobayer.blogspot.com/2010/05/maximum-matching.html (Hopcroft–Karp)

Problems:

UVA: 10004, 11080, 11396, 10505, 10080

SPOJ: BUGLIFE

LightOJ: 1009, 1149, 1184, 1403, 1201, 1209, 1152, 1304, 1218, 1373, 1206, 1150, 1429, 1242, 1171, 1356

Codeforces: 1144F, 120H, 387D, 468B

Week 4 (Data Structure: Segment Tree, Lazy):

Tutorials:

1. Shafayet Blog (Segment Tree): http://www.shafaetsplanet.com/?p=1557
2. Shafayet Blog (Lazy): http://www.shafaetsplanet.com/?p=1591
3. Codeforces (Dark knight Blog): https://codeforces.com/blog/entry/15890
4. CP-Algorithms: https://cp-algorithms.com/data_structures/segment_tree.html
5. Hacker-Earth: https://www.hackerearth.com/practice/data-structures/advanced-data-structures/segment-trees/tutorial/

Problems:

SPOJ: MULTQ3, SEGSQRSS, CNTPRIME, KGSS

Codeforces: 52C, 380C, 295A

LightOJ: 1080, 1183, 1207

Week 5 (Data Structure: BIT, LCA):

Tutorials:

1. Shafayet BIT: http://www.shafaetsplanet.com/?p=1961
2. Hackerearth: https://www.hackerearth.com/practice/notes/binary-indexed-tree-or-fenwick-tree/
3. CF gvikei's blog: https://codeforces.com/blog/entry/619
4. Algorithms Live: https://www.youtube.com/watch?v=kPaJfAUwViY
5. Hackerrank (RMQ method): https://www.hackerrank.com/topics/lowest-common-ancestor
6. Binary Lifting Method: https://cp-algorithms.com/graph/lca_binary_lifting.html

Problems:

ICPCLive: 6139 - Interval Product, 4806 - Bingo!

SPOJ: INVCNT, ADACABAA, SUMSUM, LCA, DISQUERY

LightOJ: 1101, 1112, 1348

Timus: 1752

Week 6 (Data Structure: Trie):

Tutorials:

1. http://www.shafaetsplanet.com/planetcoding/?p=1679
2. https://www.hackerearth.com/practice/data-structures/advanced-data-structures/trie-keyword-tree/tutorial/
3. https://www.quora.com/q/threadsiiithyderabad/Tutorial-on-Trie-and-example-problems
4. https://www.geeksforgeeks.org/trie-insert-and-search/
5. https://paste.ubuntu.com/p/qSJN4Ty96P/
6. https://paste.ubuntu.com/p/WCXNrkqXJJ/

Problems:

SPOJ: ADAINDEX, PHONELST, SUBXOR,

Codechef: BANKPASS, SUBXOR

POJ: 2001

UVALive: 4682

UVA: 11488

LightOJ: 1129, 1269

Codeforces: 282E

Week 7 (Pattern Matching: KMP, Z-algo):

Tutorials:

1. Shafayet (KMP): http://www.shafaetsplanet.com/?p=3209
2. CP-algorithms (KMP): https://cp-algorithms.com/string/prefix-function.html
3. Hackerearth (Z-algo): https://www.hackerearth.com/practice/algorithms/string-algorithm/z-algorithm/tutorial/
4. CP-algorithms(Z-algo): https://cp-algorithms.com/string/z-function.html
5. CF paladin8's blog: https://codeforces.com/blog/entry/3107

Problems:

Onlinejudge: 455 - Periodic Strings, 11022 - String Factoring

ICPCLive: 6439 - Pasti Pas!

LightOJ: 1255,

SPOJ: NAJPF

Codeforces: 126B

CodeChef: CDVA1606

Week 8 (DP: LCS, LIS, Matrix-Chain Multiplication):

Tutorials:

1. http://www.shafaetsplanet.com/planetcoding/?p=1862
2. http://www.lightoj.com/article_show.php?article=1000
3. https://cp-algorithms.com/sequences/longest_increasing_subsequence.html
4. https://www.techiedelight.com/matrix-chain-multiplication/
5. http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/Dynamic/chainMatrixMult.htm

Problems:

SPOJ: ELIS, LMIS, SUFFER, LCS, NAJLG, ADFRUITS, ROCK, Mixtures, Lisa

LightOJ: 1277, 1110

UVa: 481, 231, 10534, 111, 497, 437, 348, 12045

Codechef: OSQUE, CIRMERGE

Week 9 (DP: Bitmask DP, Digit-DP):

Tutorials:

1. http://www.shafaetsplanet.com/planetcoding/?p=1357
2. https://codeforces.com/blog/entry/337
3. http://shakilcompetitiveprogramming.blogspot.com/2015/09/digit-dp.html
4. https://codeforces.com/blog/entry/53960
5. https://discuss.codechef.com/t/tutorial-for-digit-dp/12839
6. https://www.hackerrank.com/topics/digit-dp
7. https://codeforces.com/blog/entry/67679

Problems:

SPOJ: GONE, RAONE, CPCRC1C, LUCIFER

CF: 431D, 628D, 215E

LightOJ: 1068, 1122, 1125, 1205, 1011, 1057, 1119, 1228

Timus: 1152, 1817

Codechef: Tools, PPXOR, Chefshop

Week 10 (Geometry: Convex Hull):

Tutorials:

1. CP algorithms (Graham's Scan): https://cp-algorithms.com/geometry/grahams-scan-convex-hull.html
2. Implementation: http://zobayer.blogspot.com/2010/02/convex-hull.html
3. Hackerrank: https://www.hackerrank.com/topics/convex-hull

Problems:

ICPCLive: 3655 - Onion Layers, 4558 - Convex Hull of Lattice Points

SPOJ: BSHEEP, VMILI

LightOJ: 1203

UVA: 10065 - Useless Tile Packers, 681 - Convex Hull Finding

Codeforces: 166B

Other: Usaco 2014 January Contest, Gold - Cow Curling

Level-2 Term-2

Tentative Class Schedule:

Week	Topic
1	Probability, Expected value
2	Graph: Articulation point, Bridge
3	Graph: FLOW
4	Data Structure: RMQ, BST
5	Data Structure: Heap, SQRT Decomposition
6	Data Structure: HLD
Online Contest-1	
7	DP optimization: Convex Hull Trick
8	DP optimization: Divide and conquer trick
9	Game Theory: Nim, Grundy
10	Geometry: Line Sweep
Online Contest-2	

Guidelines:

- Eat, Sleep, Code, Repeat…

Expectations:

- Total Number of problems solved: 300+
- CF rating:
  -At the start: 1600+
  -At the end: 1900+

Tips:

- Expected number of problems solved-1100 at least, and if you want WF, 2000 minimum. You must have solved around 350 Problems in LightOJ by now, and more than 1000+ on Codeforces
- You must also start solving from CF frequently. Solve as much div2D and div2E as possible-(minimum 300 in total, around 500 if you are serious). In team contests, select 3-4 star contests in CF Gym.

Level-3 + Level-4

Guidelines:

- Eat, Sleep, Code, Repeat…
- Learn on your own
- Take care of your juniors (Take classes, monitor their performance, motivate and help them)

Expectation:

- Achieve top ranks in onsite contests
  - -Top in divisional
  - -Top-5 in national
- **Qualify for the World Finals**
- Assuming you have worked hard, you have become the best contestant CUET has ever seen. Your name and fame spreads in BD competitive programming landscape.

Tips:

- By now you must have learnt the basic and intermediate things of almost all topics. Start specializing, discussing among teammates.
- Try to overcome your weak points and learn the things that you missed before. Focus on more advanced topics.
- For example, you decide to specialize in DP. Learn complex DP techniques like Convex Hull trick, Knuth Optimization, etc.
- Or if you want to specialize in Math, learn FFT, NTT and more advanced number theory topics.
- For data structure, splay tree, treap, HLD(solve QTREEs from SPOJ) etc.
- For graph, more esoteric problems.
- I mean, by now you must have a good idea of what topics you want to do, and further specialize on them. The above are the topics that must be learnt at least by one member of the team, but there may be rarer topics as well.