Please open this document in your PCs and NOT your mobile phones

INDEX

Few steps you could follow

★ **Step 0:** Get Your **C basics** right.

a. Conditionals and Loops(If else, while, for, switch)
b. Basic input output,
c. Mathematical operators.
d. Data types and structs
e. Pointers and pointer arithmetic
f. Bubble Sort, Insertion Sort,Selection Sort.
g. Dynamic Memory allocation
h. Basic Recursion problems
i. Join the whatsapp group of ACM, VNITs programming club.

★ **Step 1**: Switch to a **better language** (*C++ or Java only*)

a. Basic input output,
b. Data types and structs
c. Conditionals and Loops
d. Pointers and pointer arithmetic
e. Dynamic Memory allocation
f. You don't have to repeat everything again, just get the hang of the language you have chosen. You'll get fluent with it as you practice.

Why not python? Historically, many companies have been providing only C,C++ and Java as options to test the skills of a programmer.

Who should go for Java?: Those of you who have studied Java in their 12th grade, or are more comfortable with Java than C++.

Why not stick to C?: C++ comes with **STL** and 'algorithm.h' libraries that have inbuilt sort,swap,etc functions which will shorten the number of lines of code you have to write significantly. Similarly Java also helps you write better code.

Pro Tip: C++ is much less time consuming to learn as most of the syntax is the same as in C.

★ **Step 3:**Sign up at **Hackerrank** (or geeksforgeeks/practice) and try **Warmup**

★ **Step 4:** Try out problems on the topics that are given on the pages that follow. **Note that** the order of the topics given below is irrelevant, you may follow any order as per your prerogative.

★ **Step 5:** Read interview experiences from GeeksforGeeks and S.P. VNIT

★ **Step 6:** Build a Resume

a. An ideal CV would kind of look like - this
b. 3 coding projects are a must
c. Its preferred NOT to put 'hardware only' projects in it
d. Make it simple and aesthetic

The following is a gist of topics which should be covered to clear the rounds of the interview. You'll have to google about them and get an overview.

STL for C++ users✅

1. Intro to STL
2. STL cheat sheets
3. Vectors
4. Double ended queue
5. Sets and Unordered Sets
6. Pairs
7. Sort and Search
8. Maps
    - Unordered Maps
    - Ordered maps

Pointers✅

1. Pass by ref Pass by value
2. New and Delete
3. Pointers of arrays
4. Malloc vs Calloc Realloc
5. Void pointers in c only
6. Printf scanf format and Implementation

Complexity Analysis✅

1. What is time and space complexity?
2. Worst best and average cases
3. Analyse time complexity of basic codes

Implementation✅

Link

Sorting✅

1. Bubble,Selection,Insertion Sort
2. Counting Sort
    a. Sort an array 0s,1s and 2s
3. MergeSort
4. QuickSort
    a. Quick vs Merge
5. Radix Sort
6. Analyse Complexities of all sorting algos
7. Sort an Array of duplicates and remove the duplicates as well(skip for later)
    a. Use Binary Search Tree
    b. Use Heapsort

Strings ✅

1. Characters and the ASCII table
2. Inbuilt functions
    a. push_back,pop_back,size,resize,clear
    b. Find
        i. std::size_t found = str.find(str2);
        ii. if (found!=std::string::npos)

iii. std::cout << "first 'needle' found at: " << found << '\n';

c. Insert and Erase
d. Substring

3. KMP algorithm
    a. Video
4. Trie (skippable)
    a. https://www.geeksforgeeks.org/check-if-given-words-are-present-in-a-string/

OOPs ✔

1. Data Abstraction Definition
2. Encapsulation Definition
    a. To encapsulate = to protect / cover
3. Constructors and Destructors
    a. Copy Constructors
4. Access Modifiers
5. Inheritance and its types
6. Operator overloading
7. Friend Classes
    a. Friend Classes and Friend Functions
8. **Constant** objects, variables and functions
    a. Member Initalizers for them
9. Polymorphism
    a. Runtime PolyMorphism
        i. Operator Overloading
        ii. Function Overloading
    b. **Compile type Polymorphism** Function Overriding
10. Diamond Problem in OOPS - Virtual Keyword
11. Data Hiding vs Encapsulation vs Abstraction
12. Static keyword
13. Commonly-asked-oops-interview-questions

Searching ✔

1. Try Problems from Practice Mode in GeeksforGeeks
2. Divide and Conquer
    - Recursive DnC
    - While looped DnC
3. Triplet Sum = 0
4. Two pointer Search
5. Closest Pair , Closest Prod
6. Sorted and Rotated array
7. Matrix Search
8. Histogram question
9. Find in Cpp

Greedy Algorithms ✔

1. Standard Problems
2. Fractional KnapSack Problem

3. Activity Selection Problem
4. Job Sequencing Problem
    a. Basic method
    b. Using disjoint set
5. Egyptian Fraction Problem
6. Huffman Coding
7. Sort function applications in C++
8. Sorting a vector of structs/pairs in C++

Linked List ✔

1. **BASIC OPERATIONS**
    a. Insertion, Iteration, Deletion of Nodes
    b. Reverse a LL
    c. Reverse a DLL
    d. Remove Duplicates from a Sorted Linked List
    e. Find the middle Node of a LL
2. Detect a loop
    a. Use Hashing
    b. Floyd algorithm
3. Remove a loop
    a. One by one method
    b. Hashing
    c. Using a flag in the structure
    d. Counting the no of nodes
4. Delete Without Head
5. Check if a singly linked list is a palindrome

Stacks and Queues ✔

1. Implement a Stack
2. Sort a stack
    a. Using helper stack
    b. Using Recursion
3. Queue using stack
4. Stack using two queues
    a. Dequeue until last approach for achieving pop as costilier
    b. Reverse the enqueue to make push cheap
5. Applications of Stacks
    a. Histogram Question (Largest Rectangle) Using Stack
    b. Number following a pattern
    c. Next greater Element
    d. Rain water problem

Dynamic Programming

पक्का सवाल आएगा !

1. Standard Approach
2. Overlapping Subproblems
    - Recursive Solution
    - Memoized Solution(Top Down)
    - Tabulized Solution (Bottom Up)

Standard Problems in Increasing Difficulty Order :-

- ❏ Fibonacci Series Calculation
- ❏ Ways to cover a distance
- ❏ Kadane algorithm (Maximum Contiguous Subarray)
- ❏ LCS
- ❏ Longest Increasing Subsequence
- ❏ Variations of the LIS problem
- ❏ Coin Change Problem
- ❏ 0-1 Knapsack Problem
- ❏ Rod Cutting
  - ❏ Analyse the similarities and differences between the Knapsack and the Rod Cutting Problem
- ❏ Egg dropping Problem Code

"Those who cannot remember the past are condemned to repeat it."

- Dynamic Programming

Binary Trees

1. Binary Tree
   a. InsertNode in a BST
   b. Insert a Node in binary tree at level order
   c. BFS - Level Order traversal
      i. Using queue
      ii. Using recursion
   d. DFS of Binary Tree
      i. Preorder PostOrder Inorder Traversal
   e. **Delete** a Node
2. Diameter of tree
3. Tree to DLL
4. Sorted linked list to a balanced bst
5. LCA Problem in BST
6. Using Insertion in BST to remove repetitions in an array
7. Right View of Tree
8. Tree in a spiral

Graph theory ✔

1. Adjacency Matrix/List. (Representations of graph)
2. BFS,DFS of graphs (Traversals of Graphs)
3. Connected components (Total number of islands)
4. Minimum Spanning Tree
   a. Prim's Algorithm---video
   b. Kruskal's algorithm - Sort Select Eliminate
5. Detect a cycle in Directed Graph -- Using Dfs with stack
6. Detect a cycle in Undirected Graph
   a. Using DFS that's Recursive and not stack based in particular
   b. Using Disjoint Set aka Union Find
      i. Disjoint Set implemented using array like geeks for geeks
      ii. Disjoint Set implemented using a tree data
7. Flood Fill algorithm
8. Topological Sort

Heaps ✔

1. Implementation
2. Heap class in STL (C++)
3. Heap class in Java
4. Heap Sort
    a. Array Representation of any Binary tree and A complete Heap
    b. Max Min Heap
        i. Push Pop then Sort code
    c. Iterative Heap Sort
5. Kth largest element
6. Almost sorted array
7. Merge-k-sorted-arrays
8. Min cost of ropes

Mathematical Topics ✔

➔ Permutations and Combination

a. Link

➔ Bit Manipulation

a. BitMasking to reduce space complexity
b. Using xor for searching

➔ Modular Arithmetic

a. Maximum Palindrome
b. Modulo 1097

Hashing ✔

1. Overview
2. C++ Hash tools to solve problems
    a. Unorder_maps
    b. Ordered Maps
3. Ice Cream Parlour
4. Implement LRU Cache
5. Ishaan's Dilemma

BackTracking

1. GFG
2. The Knight's tour
3. Rat In A Maze
4. NQueens
5. Print all possible paths
6. All Subsets
7. Tug Of War Problem
8. Knight on a chess board
9. Standard PseudoCode

Matrix Exponentiation

for recurrence relations

- Fibonacci using Matrices

- Divide and conquer power

DBMS

Video - Learn SQL in about 4hrs, (Practical Implementation is subjective)

1. RDBMS vs Non relational DB
2. How do you design a schema
3. Normal Forms
4. Normalization in a snippet
5. Reading Files

Operating Systems

Tutorial

1. Introduction
2. Process and Management
    a. Introduction
    b. Process scheduling
    c. Process synchronization
    d. Deadlocks and threads
3. Storage management
    a. Page replacement algos - LRU cache
4. Basic Linux Commands
5. File system interface,IO interfacing,Protection and Security

Computer networks

Puzzles

INTERN Diaries.... 🖋️

1. **Shubham Patil @ Morgan Stanley, Mumbai**
2. **Kalyani Takalkar @ Amazon, Hyderabad**
3. **Konark Jain @JPMC, Mumbai**
4. **Goldman Sachs Bangalore** Office Snaps

Websites we slog hours on 💪

1. GeeksforGeeks and Hackerrank obviously
2. Abdul Bari Youtube Videos on Algorithms
3. DSA Tushar Roy Youtube Videos
4. CodeChef,Hackerearth for competitions
5. Student Portal VNIT
6. Ide.geeksforgeek.com for testing codes out
7. https://www.interviewbit.com/courses/programming/
8. VNIT leaderboard on Hackerrank
9. https://www.hackerearth.com/practice/data-structures/arrays/1-d/tutorial/
10. https://www.hackerearth.com/practice/algorithms/searching/linear-search/tutorial/
11. https://www.geeksforgeeks.org/company-interview-corner/?ref=leftbar

Some Pre written C++ Codes that help save time

1. Copy paste this code every time you start a new code

```
/*
#include<bits/stdc++.h>
#define fr(n)   for(int i=0;i<n;i++)
typedef long long int ll;
#define vint vector<ll>
#define p(x)       cout<<x<<
#define ps(x)     cout<<x<<" "
#define pl(x)      cout<<x<<endl
#define p2(x,y)  cout<<x<<" "<<y<<endl
#define p3(x,y,z)           cout<<x<<" "<<y<<" "<<z<<" "<<endl
#define pb(x) push_back(x)
#define f first
#define s second
#define ret return
using namespace std;

int main()
{
        ll sum,ans,t,n,x,y,z;
        cin>>t;
        while(t--)
        {

        }
        return 0;
}
*/
```

1. MAPS are an useful STL tool

```
for(int i=1;i<=n;i++)
{
        cin>>a>>b;
        m.insert(make_pair(a+b,i));
}
```

2. Frequency mapping of something using unordered maps

```
unordered_map<int,int> freq;
int maxf=0;
for(int i=0;i<arr.size();i++)
{
        Freq[arr[i]]++;  //No Pushback
        maxf=max(maxf,freq[arr[i]]);
        cout<<maxf<<endl;
}
```

3. Freq mapping print

```
for(auto itr=freq.begin();itr!=freq.end();itr++)
```

```
{
        cout<<itr->first<<" "<<itr->second<<endl;
}
```

4. Useful STL functions
    - to_string
    - find
    - reverse(arr.begin()+i+1.arr.end())
    - lower_bound (Iterator first, Iterator last, const val)
    - upper_bound (Iterator first, Iterator last, const val)