CP Series
Session 1

Web Club NITK
2017-18

Topics

1. Introduction to CP
2. C++ basics with STL
3. Time complexity
4. Divide and conquer
5. Searching

Introduction to Competitive Programming

Why Competitive Programming?

- Fun way to learn algorithms and data structures.
- Closeness to real world applications.
- Know where you stand in the world.
- Steep learning curve due to diversity of topics and contests.
- Helps you crack interviews. Both internships and placements. P.S: It is not going to get you the job. It will just be easier to crack coding rounds and interviews.
- Helps you design and implement algorithms, data structures and many things you will learn soon :).
- And you get goodies, job offers, swag. A lot of them :)

Introduction to C++

- Header File : <bits/stdc++.h>
- cout and cin in C++
- printf and scanf work faster

```
std::ios::sync_with_stdio(false);
```

- Precision for double values

STL

Standard Template Library

Some Common Containers

1. Vectors
2. Set
3. Map
4. Priority Queue
5. Multiset
6. Stack
7. Pair
8. Queue
9. Deque

Time Complexity

Divide and Conquer

What is divide and conquer?

- Recursively breakdown the problem into smaller subproblems of the same or related type until they these become simple enough to be solved directly or trivial.
- Combine the results of the subproblems to get the solution of the original problems.
- Example: Binary Search, Merge sort, Quicksort.

Merge Sort

Searching Algorithms

Problem statement

Find a target value within a list.
Target value?

- Particular element (Search for 5 or “Mohit”)
- Maximum/Minimum element.

Many such searching problems.

Linear Search

Algorithm:

- Traverse the list.
- Match every element with the target value.
- If it satisfies the condition, return or perform required operation.
- Else move onto the next operation.

Time Complexity:

- In the worst case, we match/compare every element in the list with the target. => We do at least ‘N’ comparisons.
- $T(n) = O(n)$.

Linear Search

C++ function.

```
// Return the index of first occurence of target_value in list. Return -1 (flag) if not present.

int linear_search(vector<int> list, int target_value) {

    for(int index = 0; index < list.size(); index++) {
        if (target_value == list[index]) {
            return index;
        }
    }

    return -1;
}
```

What if the list contains 10^7 elements and multiple queries?

Can we do anything better?

- No! :(
  - If the array is not ordered.
- YES!! :D
  - If the array is ordered (sorted in ascending or descending order).

HOW??

Binary Search

Algorithm:

1. Compare the target value with the middle element of the list.
2. If target value is equal to the middle element, done!
3. Else if the target value is greater than the middle element, consider only the right half of the list.
4. Else consider only the left half of the list.
5. Goto 1, till we reach a conclusion.

Intuition:

- If the list is ordered (assume in increasing order) and the target value is greater than the middle element of the list, we know that all the elements before the middle element i.e left half of the list are smaller than target value and hence can be ignored.
- Similarly for the case when the target value is smaller than middle element of the list.

Time complexity:

- T(n) = O(logn)

Requirement:

- The list needs to be ordered/sorted.

Let’s solve few questions?

Problem link: Queues

Q: Little girl Susie went shopping with her mom and she wondered how to improve service quality.

There are *n* people in the queue. For each person we know time *ti* needed to serve him. A person will be disappointed if the time he waits is more than the time needed to serve him. The time a person waits is the total time when all the people who stand in the queue in front of him are served. Susie thought that if we swap some people in the queue, then we can decrease the number of people who are disappointed.

Help Susie find out what is the maximum number of not disappointed people can be achieved by swapping people in the queue.

1 <= n <= 10^5

1 <= ti <= 10^9

Problem Link: Building Permutation.

Q: ***Permutation*** *p is an ordered set of integers p1, p2, ..., pn, consisting of n distinct positive integers, each of them doesn't exceed n. We'll denote the i-th element of permutation p as pi. We'll call number n the size or the length of permutation p1, p2, ..., pn.*

You have a sequence of integers *a*1, *a*2, ..., *an*. In one move, you are allowed to decrease or increase any number by one. Count the minimum number of moves, needed to build a permutation from this sequence.

1 <= n <= 3.10^5

-10^9 <= ai <= 10^9

Example:

2

3 0 (Ans: 2)

Problem link: Build the fence

Q: You have N wooden boards, having the same width, but different heights. You can take any wooden board and cut in two, as long as the resulting two wooden boards have **integer** heights. The cut can only be performed horizontally, so the two resulting boards will have the same width as the initial one.

Your goal is to build a fence consisting of K boards, all having the same height. What's the maximum height of the boards in the fence?

1 <= N <= 10^5

1 <= K <= 10^14

Example:

3 4

15 10 8

Ans: 7.

- Useful starter resources: https://wncc-iitb.org/wiki/index.php/Programming_101 & https://wncc-iitb.org/wiki/index.php/Competitive_Programming
- Competitive programming handbook: https://cses.fi/book.html (Very good handbook).
- This presentation along with STL ppt is present in: https://github.com/WebClub-NITK/Competitive-Programming-League
- Easy section of problems (searching): https://www.hackerrank.com/domains/algorithms/search
- More problems in searching: https://a2oj.com/category?ID=40 (preferably do codeforces, spoj, codechef).
- Easy section of problems (sorting): https://www.hackerrank.com/domains/algorithms/arrays-and-sorting
- More problems in sorting: https://a2oj.com/category?ID=95 (preferably do codeforces, spoj, codechef).
- Sorting and Searching section in https://www.hackerearth.com/practice/codemonk/
- STL: https://www.hackerearth.com/practice/notes/standard-template-library/ & https://www.topcoder.com/community/data-science/data-science-tutorials/power-up-c-with-the-standard-template-library-part-1/ (Highly recommended. Read part 2 also).
- Use this https://a2oj.com/categories to find and solve more problems based on category. (For now try to do searching, sorting.