ACM Summer Challenge 2k20

Editorial for Hashing

Ancient Problems require modern solutions

Author: Mihir and Jignesh

Fun fact: The testcases for one of the version were made 5 minutes before releasing questions.

In this question we have to return the position of K^{th} person from society *D* in the range of *L* to *R.*

First thing that may cross someone's mind will be linear search but the time complexity of that solution will be *O(Q*N)* in worst case which is not good enough. So to make those search easy we use hashing.

Hashing is basically a way to store data to make it more easily accessible. In this case it will help us to turn this linear search solution to a binary search solution. We just need to hash the data in such a way that for every possible *D* we have sorted list of its indices where *D* appears such that we can binary search *L* and *R* in that list and get our answers in *O(Q*logN).*

WHEN YOU SEE TWO SAME PROBLEMS WITH DIFFERENT CONSTRAINTS
AND GOT THE SOLUTION FOR HARD VERSION
imgflip.com

Ender's game

Author: Jitendra

Remember the thing I said about storing the list of indices where a particular number appears in a hashmap for previous question? We have to do the same for this one too.

First of all we will hash the number that appears on the boxes with the respective position it appears on.

Then we need to traverse the expected sequence of boxes(*B*) we want and select the index from where we want to pick that box greedily such that the number of times *Ender* needs to go through the tunnel is minimum.