BUBT Intra University Programming Contest Summer - 21
Editorial (Senior + Junior)

Problem Name: BUBT Programming Contest
Problem Setter: Maruf Ahmed Rahad
Problem Type: Basic

Solution: Just print “BUBT Intra University Programming Contest Summer 2021” with newline“\n”’.

Problem Name: Are you vaccinated?
Problem Setter: Maruf Ahmed Rahad
Problem Type: Basic

Approach: Use if-else or switch-case conditional statements to check the value of given integer. Then print the string according to the question.

Solution: https://pastebin.com/1nwUPvGs

Problem Name: The End of Online Classes
Problem Setter: Rifatul Islam
Problem Type: Basic

Solution: Difference between the two dates mentioned is **638**. Since it is an even number, you have to print **638/2**, which is **319**.

Problem Name: PRIME NUMBER
Problem setter: ALL MOON TASIR
Problem Type: Basic

Approach: 193829184982643982045 7 is a non-prime number. So we have to print NO but it is a special contest where you have to print **N0(Nzero)** without a new line.

Problem Name: MINIMUM DIFFERENCE
Problem Setter: ALL MOON TASIR
Tester : Jahin, Rifatul Islam & Khairul Anam Mubin
Problem Type: Combinatorics + STL

Hints: Learn stl function next_permutation.
Approach: Let's try all possible ways to rearrange digits in the numbers and check the difference between maximum and minimum number.

Authors solution:https://pastebin.com/6GMfgJ5r

Problem Name: You Can't Repeat it
Problem Setter: Habibullah
Tester : Jahin, Rifatul Islam & ALL MOON TASIR
Problem Type: Two Pointer + STL

Approach: This problem can be solved with an STL function called (SET) + two pointers technique.
The first pointer will "mark" the beginning. For each such beginning, iterate the second pointer until the actual element is not in the set (and also always add the element).Then add second-first, erase the first element and move the pointer. This will lead to **O(NIog(N))**.

Author Solution : https://pastebin.com/bDmiss1b

Problem Name: Lost Array
Problem Setter: Khairul Anam Mubin
Tester : Jahin & Rifatul Islam
Problem Type: Number Theory + STL

Approach: If we look closely at the given input divisors we will find that the maximum number will always be present in the lost array. So we will keep it in answer and all of its divisors will be erased from the input array. Finding the divisors should be taken not more than sqrt(N) times. After erasing the divisors we will get the maximum of this modified array again and store it in our answer and also erase their divisors.This process will continue until the input divisors array is empty. As array erases and finding maximum numbers in an array takes some bad time complexity so we can use multiset / map to store the given input divisors. Multiset / map keeps the numbers in sorted manner to find maximum and erase operation takes **log(N)** time which is a good time complexity for this problem.

Author Solution : https://ideone.com/OSLTYa

Problem Name: Divisible
Problem Setter: Khairul Anam Mubin
Tester : Jahin & Rifatul Islam

Problem Type: Number Theory + Data Structure(segment tree)

Approach: If all the numbers in an array are divisible by x then the gcd of all the numbers is also divisible by x and we can't find any x bigger than gcd of all the numbers that can divide all the numbers. Because gcd contains all common multiples of all the numbers. So we can query the result with range gcd. If the range gcd is divisible by x then the whole range is divisible by x. As there are updates in queries and we've range queries also so we have to handle it with some efficient data structure such as segment tree.
Overall time complexity: **O(NLog^2(N)))**

Author Solution: https://ideone.com/vsasX2

Problem Name: Ali Baba and N Thieves
Problem Setter: Khairul Anam Mubin
Tester: Jahin
Alternate Solution: Maruf Ahmed Rahad

Problem Type: Math

Approach: Ali Baba And N thieves that means total number of thieves is **N + 1**. They have to steal **M** liters of oil from the tank and each of them has a bottle capacity of **W** liter. So, the number of thieves needed to steal is **Needed = M / W**. If **M** is not divisible by **W** then we need **an extra thief** Needed + 1.
So, lastly if needed thieves is less than or equal N + 1 then no need for extra thieves else we need **Needed - (N + 1)** thieves.

Author Solution: https://ideone.com/TCClPi

Problem Name: Modified string
Problem setter: Maruf Billah
Tester : Jahin, Tasir & Khairul Anam Mubin
Problem Type: String + implementation

Approach: Here the main tricks are that vowels are changed.Here **a,e,k,o,u** are vowels and other letters are consonants.Then implement the code according to the given three conditions.
Author Solution: https://ideone.com/gkqmet

Problem Name: Hashmat the Brave Warrior (remake)
Setter: Jahin Hossain
Tester: Khairul Anam Mubin
Problem Type: Implementation

Approach: Just do as the problem says. Check each element of the array serially,

- Increase the strength of Hashmat if it is greater than or equal to the current soldier.
- Use the special weapon if the soldier's strength is greater than Hashmat.

You can handle the weapon count in many ways. One of them is given below,
You can use a counter to simply increase it when a special weapon has to be used. Then after traversing the whole array, you just have to check if the counter is less than or equal to the value of K given.
One thing to keep in mind here is that the strength of each soldier can be maximum 10^6 and there can be a maximum total 10^5 number of soldiers. Then in worst case, Hashmat's strength can be maximum 10^6 * 10^5 = 10^11 which does not lie in the range of 'int' you need to use 'long long int' for this problem.

Author's Solution: https://pastebin.com/1c9CkkBe

Problem Name: Angry Birds with laser beam (easy version)
Setter: Jahin Hossain
Problem Type: Geometry

Approach: You may try to solve it using multiple conditions, handling different cases manually. But the most efficient way for solving these problems is to use the concept of cross multiplication of vectors.
The ultimate goal of this problem is to determine if the two points are on the same side of the wall or different side. This can be done by cross multiplying each point with the coordinates of the wall separately. This will give two values,
C1 = cross_product(s,e,p) and C2 = cross_product(s,e,b), where s and e are two coordinates on the wall, and p,b are coordinates of pig's house and bird respectively.
But we are only concerned with the signs of C1 and C2. If both are negative or both are positive, the bird and pig are on the same side of the wall. But if one is negative and the other is positive, They are on the different side, thus no matter how hard the birds try, they can never shoot pig's house.
This concept is used for checking if a 'line segment'(finite line) and a 'line'(infinite line) intersect each other. In this problem, the wall can be assumed as an infinite line, and the laser beam as a line segment. If they intersect, the birds cannot win, otherwise the birds can win.

Author's Solution: https://pastebin.com/wDqZkezw

Problem Name: Angry Birds with laser beam (extended)
Problem Setter: Jahin Hossain
Problem Type: Geometry

Approach: As the problem says, here the number of walls has increased, and this time the wall is not infinite. If you have understood the concept in the easy version, then it should be easy for you to visualize this one.
You have to treat the walls as separate line segments that do not intersect each other. Also, the laser beam as another line segment.
To get to the answer you have to check if the line segment of the laser beam intersects with any wall, which are also line segments. In the easy one, you had to check the intersection between a line and a line segment. But here, you have to check the intersection between two line segments.
To check the intersection of two segments you have to make double checks.

- First check if the bird and the pig are on different sides of the wall segment.
- Second check if the starting point and the ending point of a wall is on different sides of the laser beam.

If both of these are true, then the wall and the laser beam intersect each other. Thus, the birds cannot win.
If the laser beam does not intersect with any of the given walls, then it's a sure shot, and the birds win.

There are more corner cases to handle in this problem.

- Check if the laser beam segment and a wall segment are collinear
- Check if any end point of the walls lies on the laser beam segment.

Time complexity: O(N), where N is the number of walls

Author's Solution: https://pastebin.com/18a1E3wx

Problem Name: Lexicographically Smallest Sequence
Problem Setter: Rifatul Islam
Tester: Khairul Anam Mubin
Problem Type: dp+greedy/implementation

Approach: using Prefix and Suffix Sums:

Maintain two arrays prefix[i] and suffix[i] where prefix[i] denotes sum of elements from index [0,i] and suffix[i] denotes sum of elements from index [i,N-1].

Now iterate from left and one by one pick elements from left for example: if you pick ‘a’ elements from left and remaining ‘k-a’ elements from right.

So the sum in this case will be prefix[a-1] + suffix[n-(k-a)]

Maintain the maximum among all.

We found the maximum sum. Now we have to print the solution Lexicographically Smallest order. Now iterate from left and one by one pick elements from left for example: if you pick ‘a’ elements from left and remaining ‘k-a’ elements from right.This time whenever we found the Maximum summation have to consider it a possible solution. If there are multiple such solutions we have to just find the Lexicographically Smallest among them.Using greedy approach we can find it easily in O(n^3).For more clear Understanding see the code given below.

Time Complexity: O(N^3)

Space Complexity: O(N)

Code : https://pastebin.com/a1nvD3QK

Problem Name: Modular Congruence

Problem Setter: Rifatul Islam
Tester: Jahin, Khairul Anam Mubin
Alternate Solution: Jahin
Problem Type: Number theory

Approach :

Given,

A % N = M

B % N = M

We can write as,
A = k1*N + M or A-M = k1*N

B = k2*N + M or B-M = K2*N

In these two equations, we can clearly see that N must be the divisor of (A-M) and (B-M).We have to find the Greatest that’s why the answer will be the GCD(A-M,B-M).

Time Complexity: O(Log(max(A-M,B-M))

Code ： https://pastebin.com/E4c71tJd