Editorial
of
Intra LU Programming Contest Summer 2017

A. #Hashtag

Just run a loop in the String and print those characters only which are not space.

***Soln** : https://pastebin.com/r0bPjmww

B. Big Bang Theory

This problem can be solved using 2 simple dfs.

From each of the position of the bomb, we will run the first dfs and mark the traversed cells as covered. The time complexity of each dfs is and the total complexity for all the dfs is O(p * n * m) . After marking the covered cells, we will run our second dfs to check the number of connected components. The complexity of the second dfs is O(n * m).

***Soln** : https://pastebin.com/zdc51k06

C. Crisis in Antartica

In this problem, we are to find the maximum amount of equal acid and base we can manage. Then multiply the amount by 57.3 we get our desired heat output.
To find the maximum amount of acid and base, we can use a dynamic programming approach. Let's consider a 2-dimensional DP array, where dp[i][j] stands for maximum volume achievable with i containers and difference between acid and base volume if j. Meaning our final answer will be dp[n][0]*57.3. Now to calculate the DP array we initially any value is zero. At each step, we can either add a container to acid or base or discard it. So, dp[i][j] = max(dp[i-1][j],dp[i-1][j+a[i]]+a[i],dp[i-1][abs(j-a[i])]+a[i]).

Say, acid or base which has less volume we add a[i] to it then if j is the new difference then it should come from j+a[i]. Otherwise it's abs(j-a[i]). If i'th container is discarded then j remains unchanged.
Now to optimize the solution to manage in the given memory limit you need to use row swap technique.

***Soln: https://pastebin.com/e4Yn9wkN**

D. Easy Prime!

Use Sieve to pre-generate prime numbers. Now solve the problem using Segment Tree.

***Soln :** https://pastebin.com/d8108yPk

E. Is It A Square?

This problem can be solved using Hashing Technique. We will create a bit string for each of the element of the array. Each of the index of the bit string will represent whether some prime occurs in that number even or odd number of times.

For example if we consider first 3 primes, the bit-string of 10 will be . It means, in the prime factorization of , there are odd number of “2”s and “5”s. And “3” occurs even times. Let us consider another example, **.** The bit-string will be . Now from the bit-string how can we say that it is a square number or not ? If the bit-string contains only zeroes (which means each of the prime occurs even number of times), then we can call the number a square number. So for a given range we have to the find the sum of all bit-strings, each index-wise value modulo 2. If the result contains complete zeroes, then the product of numbers in that range is square. Assume we have been given two positions and . If the sum of bit-strings from to are complete zeroes, it means the sum bit-strings from position to and position to are same. Here, we use Hashing technique to compare the cumulative sum of bit-strings efficiently. If they match, then the product is a square. **Note that** ,before checking the bit-strings, check whether some element between position to contains or not. And also there must be even number of negative numbers.
***Soln:** https://pastebin.com/iE4ThMGs

F. Easy Peasy Subset Sum

Just solve the problem for some small sets like little oishee and see the pattern .
And always remember any calculation with subsets is some how linked with tow's powers !

***Soln :** https://pastebin.com/Y4zQVkJd

G. Pizza Delivery

This is a classical problem about **n** machine consistently handling **m** jobs.
Let, packing the **i-th** pizza takes **ai** unit time and the delivery time is equal to **bi**.
Now, Look for the smallest among the numbers **ai** and **bi** (i=1,n). If the lowest value is **ai** then put the **i-th** pizza for processing first, if **bi** the latter. Then remove the i-th pizza from the list, repeat this process for the remaining **(n-1)th** pizza.
Note that the chef will not wait if the delivery time is longer for the **i-th** pizza.

***Soln :** http://paste.ubuntu.com/24900522

H. Left Prime

Use Sieve to pre-generate prime numbers. Now apply binary search or cumulative sum technique to find the answer.
Cumulative sum technique -> Take an array **A** and save all the prime numbers from index **0** to **i** in **Ar[i]**.

***Soln :** https://pastebin.com/ija3EyEZ