diuacm

Daffodil International University

Problem Analysis

Take-Off Programming Contest Fall-23

[Preliminary - B Slot]

Problem Setters

Md. Taufik Ferdous
Imrul Hasan Sifat
Sudipto Kumar Mitra
Md. Uodoy Hossan Rafi
Saimur Rahman Robin
S.M. Ashibur Rahman
Arfan Uddin

Reviewers

Saiful Islam Ramim
Nesar Ahmed
Nabid Anzum Akash
Sabbir Ahamed Mridha

Judges

Fahim Istiak Nabil
Arfan Uddin

Problem A: King of the Programming World!

Category: Give away
Setter: Md. Taufik Ferdous
Solution Idea:
You just need to print the line “I'm Going To Become King Of The Programming World!” (without quotes).

Problem B: The Foolish Hunter

Category: Basic math
Setter: Imrul Hasan Sifat
Solution Idea:
Number of places *X*, number of animals in each places *Y*
Hence, total number of animals = *X* * *Y*
As a weapon can kill at most *Z* numbers of animals, then the number of required weapons would be *(X * Y)* divide by *Z* and if there is any remainder then add *1* to the answer.
Another simple formula would be, *((X * Y) + Z - 1) / Z*

Problem C: A Wonderful Match

Category: If-else
Setter: Sudipto Kumar Mitra
Special Thanks: Arfan Uddin
Solution Idea:
Calculate the run rate by dividing the runs by the overs. Then check if the run rate is greater or equal to 6 or not. Then the output will be "Yes" or "No" (without quotes).

Another approach, calculate the total balls bowl by (Overs * 6). Then check if the runs are greater or equal to the total balls or not.

Problem D: Amusement Park

Category: Nested If-else
Setter: Md. Uodoy Hossan Rafi
Solution Idea:
Strategy: Declare a variable ticket_price to store the calculated ticket price.
Use a series of if-else statements to check the conditions defined by the park's pricing guidelines:

- If the age is less than 3 or greater than or equal to 70, set ticket_price to 0 (free ticket).
- If the age is between 3 and 12 (inclusive), set ticket_price to $10.
- If the age is between 65 and 69 (inclusive), set ticket_price to $15.

- If the age is between 30 and 40 (inclusive) and the weight is greater than 75 kg, set ticket_price to $18.
- For all other cases, set ticket_price to the regular price of $20.

Output: Finally, check the value of ticket_price and output the result accordingly:
If ticket_price is 0, print "Free."
Otherwise, print the ticket_price.

Problem E: Kakashi's Game

Category: Loop + If-else
Setter: Saimur Rahman Robin
Observation:
There are N Buildings in a row which have heights from (H_1 to H_N). Robot P & Q wants to land on top of a maximum number of buildings starting from building K.
But there are two conditions for moving from one building to another:

- Robot P can only move to the right side from position K and it can never land on a lower-height building.
- Robot Q can only move to the left side from position K and it can never land on a higher-height building.

By following the above rules, you need to determine the steps for Robot P & Q and print the building heights on which they will land sequentially.

Solution Idea:

- Firstly we need to take the input N, K - the number of the buildings and the starting position.
- Then we take input of the heights for all N buildings.
- Now according to the rules, Robot P will first land on the K^{th} building, then it will go to the $(K+1)^{th}$ to N^{th} building one by one and check if it can land on that building or not.
- In the same way, Robot Q will first land on the K^{th} building, then it will go to the $(K-1)^{th}$ to 1^{st} building one by one and check if it can land on that building or not.
- While checking, if the i^{th} building height follows the rules for the specific robots, then print the building height sequentially for Robot P and Q respectively.

Problem F: The Great Sherlock Holmes

Category: String, Implementation
Setter: S.M. Ashibur Rahman
Solution Idea:

Determining the existence of a substring is a straightforward task. A simple approach involves implementing a nested loop to check if the substring can be formed.
What is a substring?
A substring is a contiguous sequence of characters within a string. In other words, it's a subset of characters taken from a larger string without changing their order. For example, consider the string "hello." Some of its substrings include "h", "he", "hell", "hello", "e", "el" and so on.
Therefore we need to check if the string S(fractional part) is a substring of the Sni(Name of the person). To do that, We will first run a loop which will start from 0 to the difference of the two strings. Why's that? Because otherwise it will go out of bounds, giving us runtime errors. Then, we'll run another loop from 0 to fractional part length.
After that we will check if the fractional part character and the people name character matches or not. We can use an if condition for that. If it matches we increase the counter, else we break it. If all characters match, then the substring exists.

The time complexity of this approach is O(n * m), where n is the length of the main string and m is the length of the substring. This is because the nested loop iterates over all possible substring positions in the main string.

Problem G: Realm of Numericon

Category: Number Theory
Setter: Arfan Uddin
Special Thanks: Fahim Istiak Nabil
Solution Idea:

This problem can be solved using Legendre's Formula and Prime Factorization.

Legendre's Formula: Let m be a prime number p, then the exponent of p in the prime factorization of $n!$ is:

$$f(n, p) = \lfloor n/p \rfloor + \lfloor n/p^2 \rfloor + \lfloor n/p^3 \rfloor + \ldots \text{ until } p^x \text{ is less than or equal to } n.$$

When m is not a prime, then we need to express the composite number m as a product of its prime factors. Let's say, $m = p_1^{c1} * p_2^{c2} * p_3^{c3} * \ldots * p_k^{ck}$ where $p_1, p_2, p_3, \ldots, p_k$ are distinct prime factors of m and $c_1, c_2, c_3, \ldots, c_k$ are the corresponding exponents.

For each prime factor p_i (where $1 \le i \le k$) we can use Legendre's formula to find the exponent of p_i in the prime factorization of $n!$. Then the largest exponent of m that divides $n!$ would be the minimum of $f(n, p_i)$ divided by c_i.

$$min_{i=1}^{k} \lfloor f(n, p_i)/c_i \rfloor$$

Now, what is the largest exponent of m that divides ^nC_r?

For each prime factor p_i (where $1 \leq i \leq k$) we can apply Legendre's formula to find the exponent of p_i in the prime factorization of $n!$, $r!$ and $(n-r)!$ separately. Then subtract the exponents we get for $r!$ and $(n-r)!$ from the exponent for $n!$ to find the exponent of p_i in the prime factorization of ^nC_r.

So, the largest exponent of m that divides ^nC_r would be:-

$$min_{i=1}^{k} \lfloor (f(n, p_i) - f(r, p_i) - f(n-r, p_i))/c_i \rfloor$$

Time and Space complexity: To avoid worst-case time complexity, use Sieve of Eratosthenes algorithm to store all primes up to 10^6, facilitating efficient prime factorization of m.

Let, $N = 10^6$, then to store all prime up to N using Sieve of Eratosthenes:
Time complexity, $O(N * log(log(N)))$
Space complexity, $O(N)$

Let, S be the number of primes which are less than or equal to $\sqrt{m}$. Then in each test cases, prime factorization of m using stored primes:
Time complexity, $O(T * S)$
In the worst case, S can be at most $78,498$ (Number of primes less than or equal to 10^6). However, it's still enough to pass within the given time limit.

So, the overall complexity of our solution:
Time complexity, $O(N * log(log(N)) + T * S)$
Space complexity, $O(N)$
All other complexities are small enough to be neglected.