Editorial of DIU Take Off Programming Contest, Fall-25 [Preliminary B]

Hosted By

Department of Computer Science and Engineering, Daffodil International University

December 05, 2025

Problem A: Three answers, One question

Problem Setter: Sharif Md. Husain
Problem Tester: A.K.M Shohan
Category: Giveaway
Solve Count: 359

Solution:

Print *Hamzaa.* Don't forget to add a newline!

Problem B: Ticket Treasury

Problem Setter: Md Minhajul Islam
Problem Tester: Juhair Akhter
Category: Basic Math
Solve Count: 266

Solution:

This problem is about calculating how much money the stadium earns from selling football match tickets. You are given three numbers:

- V → number of VIP tickets sold
- R → number of Regular tickets sold
- X → price of one Regular ticket

But there is one special rule:

- VIP ticket price = 3 × X

So VIP tickets cost three times more than Regular tickets.

Step 1: Calculate VIP revenue

vip_money = 3 × X × V

Step 2: Calculate Regular revenue

regular_money = X × R

Step 3: Add them

total = vip_money + regular_money

That's the final answer.

Problem C: The Great Wall of Putul

Problem Setter: Anup Barman
Problem Tester: Sajib Hasan
Category: If-Else
Solve Count: 130

Solution:

The opponent takes **S** shots and Putul can save **K**; if **S ≤ K**, the opponent scores 0, otherwise they score **S − K** goals.
Bangladesh scores exactly **A** goals because each assist guarantees one goal.
Compare Bangladesh’s goals with the opponent's:

- If A > opponent_goals, print *Shabash Bangladesh*
- If A == opponent_goals, print *Well Played*
- Otherwise, print *Heartbreak*

Problem D: Not football

Problem Setter: Abid Hasan
Problem Tester: Md Abdul Quym Shanto
Category: Loop + AdHoc
Solve Count: 108

Solution:

The solution of this problem is pretty straight forward. Take an array of length **N+1** starting from 1 based indexing. Then just take two count variables for each team, let’s say **Acnt** and **Bcnt**. After that, check if the index and value on that index is both odd, if yes, increase Acnt. Or, check if the index and value on the index is both even, if yes, then increase Bcnt. Lastly check whichever of Acnt or Bcnt is greater and print “Anda” if Acnt is greater or print “Dim” if Bcnt is greater. If both count the same, print “Draw”.

Problem E: BFL Star Player Festival

Problem Setter: Md. Sakibur Rahman (Sijan)
Problem Tester: Mohammad Tasnim Ahmed
Category: Implementation + Array + Nested Loop
Solve Count: 2

Solution:

To maximize the product of two distinct numbers, we must consider that the product of two negative numbers is positive. Thus, the optimal solution is formed by either the two largest values in the list (producing a large positive from positives) or the two smallest values (producing a large positive from negatives).

For each club, perform a linear scan **O(M)** to identify the two maximum values **(max_1, max_2)** and the two minimum values **(min_1, min_2)**. The answer is simply **max($max_1 \times max_2$, $min_1 \times min_2$)**. Since the input values range up to **10^9**, the product can reach **10^{18}**, so you must use 64-bit integers (long long in C/C++) to avoid overflow.

Problem F: The Decisive Penalty

Problem Setter: Siyam Bhuiyan
Problem Tester: Md. Sifat
Category: Pre-Stopper / String
Solve Count: 1

Solution:
When a minus appears before a parenthesis: -(a+b-c)
all signs inside must be inverted (+ ↔ -). If parentheses are nested, inversions can stack. So for every position we must know whether we are inside a “flipped” zone. We use an integer array as a stack in C. Each entry stores:

- 0 → signs are normal
- 1 → signs must be flipped

While reading the string:

- If we see '(', push the current flip state inside the stack.
- If we see ')', pop.
- If we see '+' or '-', then:
  - if current flip state = 0, make operator -
  - if current flip state = 1, make operator +
- Digits do not affect the state; they just mark that we entered a number.

We update the current flip state as **v[top]** after each step.This fully simulates how parentheses affect signs.The stack correctly tracks nested minus-brackets.Every operator is flipped exactly when its surrounding minus requires it.Thus the final expression matches the one with parentheses removed.

Problem G: Will Hamza Goal?

Problem Setter: Md. Whahidul Islam Payel
Problem Tester: Md. Uodoy Hossan Rafi
Category: Stopper / Number Theory
Solve Count: 0

Solution:

Idea: Factorize $X = \prod p_i^{k_i}$. A subarray $[L, R]$ has a subset whose product is divisible by X iff the total exponent of each p_i in $A_L, ..., A_R$ is at least k_i. Order of elements doesn't matter.

Preprocess:

- Build SPF sieve (up to 10^6) for fast factorization.
- Factorize X to get primes p_i and required exponents k_i.
- For each p_i build prefix sums pref[i][j] = total exponent of p_j in $A_1 .. A_i$ (count multiplicities by dividing each A_i by p_j).

Query: For $[L, R]$ check for all j: pref[R][j] - pref[L-1][j] >= k_j. If true for every prime, print **Goal**, else print **Missed**.

Correctness: Multiplicities add under multiplication, so total exponent in any chosen subset equals sum of exponents from selected elements; existence reduces to whether the subarray provides the required exponents.