Intra-University Programming Contest 2025
Juniors
Editorial
Contest Link
Github Link

Rahul Kumar Ghosh and Sourav Mondal
Northern University of Business and Technology Khulna
Department of Computer Science and Engineering

Contents

1 A. The Thousand-Mile Journey

Author: Sourav Mondal
Legend/Story: Sourav Mondal
Tester: Rahul Kumar Ghosh, Sourav Mondal
Tag: Implementation

1.1 Editorial:

Just Print **Hello, World!**

1.2 Implementation in C

```c
#include <stdio.h>
int main()
{
    printf("Hello, World!\n");
    return 0;
}
```

1.3 Implementation in C++

```cpp
#include<bits/stdc++.h>
using namespace std;
#define int long long
#define endl '\n'
int32_t main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    cout << "Hello, World!" << endl;
    return 0;
}
```

1.4 Implementation in Python

```python
def main():
    print("Hello, World!")
if __name__ == "__main__":
    main()
```

2 B. Prime Number Obsession Turns Into Madness

Author: Rahul Kumar Ghosh
Legend/Story: Rahul Kumar Ghosh
Tester: Rahul Kumar Ghosh, Sourav Mondal
Tag: Mathematics, Number Theory

2.1 Editorial

We need to check whether a number n can be prime and whether it can be represented in the form

$$n = 2^{2^m} + 1.$$

Since n can be as large as $2^{63} - 1$, we must determine valid values of m.

2.2 Approach

The value of m can range only up to 5, because for $m \geq 6$ the value of n exceeds the limit $2^{63} - 1$. Let us compute the values step by step:

$$\begin{aligned} m &= 0 \quad 2^{2^0} + 1 = 3, \\ m &= 1 \quad 2^{2^1} + 1 = 5, \\ m &= 2 \quad 2^{2^2} + 1 = 17, \\ m &= 3 \quad 2^{2^3} + 1 = 257, \\ m &= 4 \quad 2^{2^4} + 1 = 65537, \\ m &= 5 \quad 2^{2^5} + 1 = 4294967297. \end{aligned} \tag{1}$$

Thus, only these six numbers are possible candidates. We then check if the number is prime. If it is prime, we print "YES"; otherwise, "NO".

$$m = 5,\ 2^{2^5} + 1 = 4294967297$$

This number is not a prime, so in this case we have to print "NO".

2.3 Implementation in C

```c
#include<stdio.h>
int main()
{
    int test;
    scanf("%d", &test);
    while(test--){
        int n;
        scanf("%d", &n);
        if (n == 3 || n == 5 || n == 17 || n == 257 || n ==
           65537){
            printf("Yes\n");
        }else{
            printf("No\n");
        }
    }
}
```

2.4 Implementation in C++

2.4.1 Implementation 1

```cpp
#include<bits/stdc++.h>
using namespace std;
#define int long long
#define endl '\n'
auto QKnot = []()->void
{
    int n;
    cin >> n;
    if (n == 3 || n == 5 || n == 17 || n == 257 || n == 65537) {
        cout << "YES" << endl;
    } else{
        cout << "NO" << endl;
    }
};
int32_t main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    bool bl = true;
    bl?[]()->void
    {
        int test;
        cin >> test;
        while(test--) QKnot();
    }():QKnot();
    return 0;
}
```

2.4.2 Implementation 2

```cpp
#include<bits/stdc++.h>
using namespace std;
#define int long long
#define endl '\n'
int32_t main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    vector<int> arr;
    for(int i = 0; i < 6; i++){
        int x = pow(2, i);
        arr.push_back(pow(2, x) + 1);
    }
    vector<int> prime;
    auto isprime = [](int n)->bool
    {
        if(n == 2) return true;
        else if (n == 1 || n % 2 == 0) return false;
        else{
            for(int i = 3; i * i <= n; i += 2){
                if(n % i == 0) return false;
            }
        }
        return true;
    };
    for(int i = 0; i < arr.size(); i++){
        if(isprime(arr[i])){
            prime.push_back(arr[i]);
        }
    }
    int test;
    cin >> test;
    while(test--){
       int n;
       cin >> n;
       bool tag = false;
       for(int i = 0; i < prime.size(); i++){
            if(n == prime[i]){
                tag = true;
            }
       }
       cout << (tag ? "yes" : "no") << endl;
    }
    return 0;
}
```

2.4.3 Implementation 3

```cpp
#include<bits/stdc++.h>
using namespace std;
#define int long long
#define endl '\n'
const int mod = 1e9 + 7;
auto QKnot = []()->void
{
    int n;
    cin >> n;
    double x = log2(log2(n - 1));
    int p = round(x);
    if (p <= 4 && fabs(x - p) < 1e-9){
        cout << "YES" << endl;
    }else {
        cout << "NO" << endl;
    }
};
int32_t main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    bool bl = true;
    bl?[]()->void
    {
        int test;
        cin >> test;
        while(test--) QKnot();
    }():QKnot();
    return 0;
}
```

2.5 Implementation in Python

```python
def main():
    test = int(input())
    for _ in range(test):
        n = int(input())
        if n == 3 or n == 5 or n == 17 or n == 257 or n == 65537:
            print("YES")
        else:
            print("NO")
if __name__ == "__main__":
    main()
```

3 C. Unique Roll Number

Author: Rahul Kumar Ghosh
Legend/Story: Rahul Kumar Ghosh
Tester: Rahul Kumar Ghosh, Sourav Mondal
Tag: Hashing, Implementation

3.1 Editorial

You are given a list of roll numbers.

- If every roll number appears no more than once, print any case-insensitive form of **"YES"**.
- Otherwise, print any case-insensitive form of **"NO"**.

In short, check whether all roll numbers are unique.

3.2 Key Idea

- If no roll number is repeated → print **YES**.
- If at least one roll number appears more than once → print **NO**.

This is simply a duplicate detection problem.

3.3 Approaches

3.3.1 Approach 1: Using Set

- Insert each roll number into a set.
- A set stores only unique elements.
- If the size of the set is equal to the number of roll numbers, all are unique → print YES.
- Otherwise, duplicates exist → print NO.

3.3.2 Complexity

Time Complexity: $O(n)$
Space Complexity: $O(n)$

3.3.3 Implementation in C++

```cpp
#include<bits/stdc++.h>
using namespace std;
#define int long long
#define endl '\n'
int32_t main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    set<int> set_;
    int n;
    cin >> n;
    for(int i = 0; i < n; i++){
        int a;
        cin >> a;
        set_.insert(a);
    }
    cout << (set_.size() == n ? "YES" : "NO") << endl;
    return 0;
}
```

3.3.4 Approach 2: Sorting

- Sort the roll numbers.
- Check if any consecutive elements are equal.
- If yes → duplicate found → print NO.
- Otherwise → print YES.

3.3.5 Complexity

Time Complexity: $O(n \log n)$
Space Complexity: $O(1)$

3.3.6 Implementation in C++

```cpp
#include<bits/stdc++.h>
using namespace std;
#define int long long
#define endl '\n'
const int mod = 1e9 + 7;
auto solve = []()->void
{
    int n;
    cin >> n;
    vector<int> arr(n);
    for(auto &x : arr) cin >> x;
    sort(arr.begin(), arr.end());
    for(int i = 0; i < n - 1; i++){
        if(arr[i] == arr[i + 1]){
            cout << "NO" << endl;
            return;
        }
    }
    cout << "YES" << endl;
};
int32_t main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    solve();
    return 0;
}
```

4 D. High Valyrian

Author: Rahul Kumar Ghosh
Legend/Story: Rahul Kumar Ghosh
Tester: Rahul Kumar Ghosh, Sourav Mondal
Tag : Implementation, Maping

4.1 Editorial

Adam di Hul needs to command his dragon Urraxion in High Valyrian. We are given a command in English:

- If the command exists in the dictionary, output its High Valyrian translation.
- Otherwise, output `Skoros?` which means "What?".

4.2 Approaches

This is a simple dictionary mapping problem. Store all English-to-High Valyrian pairs in a map:

English	High Valyrian
Fire	Dracarys
Serve	Dohaeras
Calm	Lykiri
Wait	Umbas
Focus	Rybas
Come	Mazis
Forward	Naejot

4.2.1 Complexity

Time Complexity: $O(1)$
Space Complexity: $O(1)$

4.2.2 Implementation in C++ using map

```cpp
#include<bits/stdc++.h>
using namespace std;
#define int long long
static map<string, string> highValyrian = {
    {"Fire", "Dracarys"},
    {"Serve", "Dohaeras"},
    {"Calm", "Lykiri"},
    {"Wait", "Umbas"},
    {"Focus", "Rybas"},
    {"Come", "Mazis"},
    {"Forward", "Naejot"}
};
int32_t main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    int test;
    cin >> test;
    while(test--){
        string s;
        cin >> s;
        if (highValyrian.contains(s)){
            cout << highValyrian[s] << endl;
        }else {
            cout << "Skoros?" << endl;
        }
    }
    return 0;
}
```

4.2.3 Implementation in python using dictionary

```
highValyrian = {
    "Fire": "Dracarys",
    "Serve": "Dohaeras",
    "Calm": "Lykiri",
    "Wait": "Umbas",
    "Focus": "Rybas",
    "Come": "Mazis",
    "Forward": "Naejot"
}

def QKnot():
    s = input().strip()
    print(highValyrian.get(s, "Skoros?"))

def main():
    try:
        test = int(input())
        for _ in range(test):
            QKnot()
    except:
        QKnot()

if __name__ == "__main__":
    main()
```

5 E. The most beautiful equation in mathematics

Author: Rahul Kumar Ghosh
Legend/Story: Rahul Kumar Ghosh
Tester: Rahul Kumar Ghosh, Sourav Mondal
Tag : Mathematics, ad hoc, Number theory

5.1 Editorial

We want to evaluate:

$$e^{ni\pi} + 1 \tag{1}$$

From Euler's formula:

$$e^{ni\pi} = \cos(n\pi) + i\sin(n\pi)$$

Since $\sin(n\pi) = 0$, we have:

$$e^{ni\pi} = (-1)^n$$

Therefore,

$$e^{ni\pi} + 1 = (-1)^n + 1$$

5.2 Approach

Let us consider cases for n:

- If n is even, then $(-1)^n = 1$:

$$e^{ni\pi} + 1 = 1 + 1 = 2$$

- If n is odd, then $(-1)^n = -1$:

$$e^{ni\pi} + 1 = -1 + 1 = 0$$

5.2.1 Complexity

1. **Time Complexity:** $O(1)$
2. **Space Complexity:** $O(1)$

5.2.2 Implementation in C

```c
#include<stdio.h>
int main()
{
    int test;
    scanf("%d", &test);
    while(test--){
        int n;
        scanf("%d", &n);
        if(n % 2 == 0){
            printf("%d\n", 2);
        }else{
            printf("%d\n", 0);
        }
    }
    return 0;
}
```

5.2.3 Implementation in C++

```cpp
#include<bits/stdc++.h>
using namespace std;
#define int long long
#define endl '\n'
const int mod = 1e9 + 7;
auto solve = []()->void
{
    int n;
    cin >> n;
    cout << ((n & 1) ? 0 : 2) << endl;
};
int32_t main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    int test;
    cin >> test;
    while(test--){
        solve();
    }
    return 0;
}
```

5.2.4 Implementation in Python

```python
def main():
    test = int(input())
    for i in range(test):
        n = int(input())
        print(2 if n % 2 == 0 else 0)
if __name__ == "__main__":
    main()
```

6 F. Is it a Perfect Square?

Author: Rahul Kumar Ghosh
Legend/Story: Rahul Kumar Ghosh
Tester: Rahul Kumar Ghosh, Sourav Mondal
Tag : Greedy, Binary Search

6.1 Editorial

In this problem, we have to check whether a number is a perfect square or not. A number is a perfect square if

$$x = \sqrt{n}$$

where x is an integer. There are many ways to determine whether a number is a perfect square.

6.2 Approach 1

First, we find the square root of the number:

$$x = \sqrt{n}$$

This will give us a floating-point value. Then, we typecast the value into an integer and check whether

$$x^2 = n$$

If it is equal to n, then the number is a perfect square; otherwise, it is not.

6.2.1 Complexity

1. **Time Complexity:** $O(1)$
2. **Space Complexity:** $O(1)$

6.2.2 Implementation in C++

```cpp
#include<bits/stdc++.h>
using namespace std;
#define int long long
#define endl '\n'
#define space " "
const int mod = 1e9 + 7;
auto solve = []()->void
{
    int n;
    cin >> n;
    int x = sqrt(n);
    cout << ((x * x == n) ? "YES" : "NO") << endl;
};
int32_t main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    int test;
    cin >> test;
    while(test--){
        solve();
    }
    return 0;
}
```

6.3 Approach 2

We apply binary search in the range $[1, n]$:

- Take the middle value $mid = low + \frac{high-low}{2}$.
- Compute mid^2 and compare it with n.
  - If $mid^2 = n$, then n is a perfect square.
  - If $mid^2 < n$, move the search space to the right: $low = mid + 1$.
  - If $mid^2 > n$, move the search space to the left: $high = mid - 1$.
- Continue until $low > high$. If no exact match is found, n is not a perfect square.

6.3.1 Complexity

1. **Time Complexity:** $O(\log n)$
2. **Space Complexity:** $O(1)$

6.3.2 Implementation in C++

```cpp
#include<bits/stdc++.h>
using namespace std;
#define int long long
#define endl '\n'
#define space " "
const int mod = 1e9 + 7;
auto solve = []()->void
{
    int n;
    cin >> n;
    int low = 1, high = n;
    bool isPerfectSquare = false;
    while(low <= high){
        int mid = low + (high - low) / 2;
        if(mid * mid == n){
            isPerfectSquare = true;
            break;
        }
        else if(mid * mid < n){
            low = mid + 1;
        }
        else{
            high = mid - 1;
        }
    }
    cout << (isPerfectSquare ? "YES" : "NO") << endl;
};
int32_t main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    int test;
    cin >> test;
    while(test--){
        solve();
    }
    return 0;
}
```

7 G. Johann Carl Friedrich Gauss

Author: Rahul Kumar Ghosh
Legend/Story: Rahul Kumar Ghosh
Tester: Rahul Kumar Ghosh, Sourav Mondal
Tag : Number theory, Mathematics

7.1 Editorial

In this problem, if we sum the numbers one by one, it will result in a time limit exceeded. Instead, we can use a formula instead of the brute force solution.

$$1, 2, 3, 4, \ldots, n-2, n-1, n$$

We can pair the numbers as follows:

$$1+n,\ 2+(n-1),\ 3+(n-2), \ldots$$

Each pair sums to $(n+1)$, and there are $\frac{n}{2}$ such pairs. So, the total sum will be:

$$\frac{n \times (n+1)}{2}$$

Since the value of n can be as large as 10^9, the result may exceed the range of a 32-bit integer. Therefore, we should use a 64-bit integer (`long long`) to avoid overflow.

7.2 Approach

7.2.1 Complexity

1. **Time Complexity:** $O(1)$
2. **Space Complexity:** $O(1)$

7.2.2 Implementation in C++

```cpp
#include<bits/stdc++.h>
using namespace std;
#define int long long
#define endl '\n'
int32_t main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    int test;
    cin >> test;
    while(test--){
       int n;
       cin >> n;
       cout << n * (n + 1) / 2 << endl;
    }
    return 0;
}
```

7.2.3 Implementation in Python

```python
def main():
    test = int(input())
    for i in range(test):
        n = int(input())
        print((n * (n + 1)) // 2)

if __name__ == "__main__":
    main()
```

8 H. Mother Of Dragon

Author: Rahul Kumar Ghosh
Legend/Story: Rahul Kumar Ghosh
Tester: Rahul Kumar Ghosh, Sourav Mondal
Tag : Implementation

8.1 Editorial

If the first number is the largest among them, then print `Drogon`. If the second number is the largest, then print `Rhaegal`. If the third number is the largest, then print `Viserion`.

8.2 Approach

First, find the maximum number. Then, check which number it corresponds to and print the respective name.

8.2.1 Complexity

1. **Time Complexity:** $O(1)$
2. **Space Complexity:** $O(1)$

8.2.2 Implementation in C++

```cpp
#include<bits/stdc++.h>
using namespace std;
#define int long long
#define endl '\n'
int32_t main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
    int test;
    cin >> test;
    while(test--){
       int a, b, c;
       cin >> a >> b >> c;
       int mx = max({a, b, c});
       if (a == mx){
            cout << "Drogon" << endl;
       }else if(b == mx){
            cout << "Rhaegal" << endl;
       }else{
            cout << "Viserion" << endl;
       }
    }
    return 0;
}
```