# Contents

# 1 Contest Setup

## 1.1 vimrc

```
 1  set number        " Show line numbers
 2  set mouse=a       " Enable inaction via mouse
 3  set showmatch        " Highlight matching brace
 4  set cursorline       " Show underline
 5  set cursorcolumn     " highlight vertical column
 6
 7  filetype on "enable file detection
 8  syntax on   "syntax highlight
 9
10  set autoindent       " Auto-indent new lines
11  set shiftwidth=4     " Number of auto-indent spaces
12  set smartindent      " Enable smart-indent
13  set smarttab         " Enable smart-tabs
14  set softtabstop=4    " Number of spaces per Tab
15
16  " ---------Optional----------
17
18  set undolevels=10000     " Number of undo levels
19  set scrolloff=5      " Auto scroll
20
21  set hlsearch      " Highlight all search results
22  set smartcase     " Enable smart-case search
23  set ignorecase    " Always case-insensitive
24  set incsearch     " Searches for strings incrementally
25
26  highlight Comment ctermfg=cyan
27  set showmode
28
29  set encoding=utf-8
30  set fileencoding=utf-8
31  scriptencoding=utf-8
```

## 1.2 bashrc

```
 1  alias g++="g++ -Wall -Wextra -std=c++11 -O2"
```

## 1.3 C++ template

```
 1  #include <bits/stdc++.h>
 2
 3  using namespace std;
 4
 5  #define x first
 6  #define y second
 7
 8  typedef long long int ll;
 9  typedef pair<int, int> ii;
10
11  int main()
12  {
13      return 0;
14  }
```

## 1.4 Java template

```java
import java.io.*;
import java.util.*;

public class Main
{
    public static void main(String[] args)
    {
        MyScanner sc = new MyScanner();
        out = new PrintWriter(new BufferedOutputStream(System.out));
        // Start writing your solution here.

        // Stop writing your solution here.
        out.close();
    }

    public static PrintWriter out;

    public static class MyScanner
    {
        BufferedReader br;
        StringTokenizer st;

        public MyScanner()
        {
            br = new BufferedReader(new InputStreamReader(System.in));
        }

        boolean hasNext()
        {
            while (st == null || !st.hasMoreElements()) {
                try {
                    st = new StringTokenizer(br.readLine());
                } catch (Exception e) {
                    return false;
                }
            }
            return true;
        }

        String next()
        {
            if (hasNext())
                return st.nextToken();
            return null;
        }

        int nextInt()
        {
            return Integer.parseInt(next());
        }

        long nextLong()
        {
            return Long.parseLong(next());
        }
```

```java
        }

        double nextDouble()
        {
            return Double.parseDouble(next());
        }

        String nextLine()
        {
            String str = "";
            try {
                str = br.readLine();
            } catch (IOException e) {
                e.printStackTrace();
            }
            return str;
        }
    }
}
```

## 2  Reminder

1. Read the problem statements carefully. Input and output specifications are crucial!
2. Estimate the **time complexity** and **memory complexity** carefully.
3. Time penalty is 20 minutes per WA, **don't rush**!
4. Sample test cases must all be tested and passed before every submission!
5. Test the corner cases, such as 0, 1, -1. Test all edge cases of the input specification.

## 3  Useful code

### 3.1  Fast Exponentiation $O(log(exp))$

```cpp
ll fast_pow(ll base, ll exp, ll mod)
{
    if (exp == 0)
        return 1LL;
    ll res = 1;
    while (exp > 0) {
        if (exp & 1) {
            res = ((res % mod) * (base % mod)) % mod;
        }
        exp >>= 1;
        base = (base * base) % mod;
    }
    return res;
}
```

### 3.2  GCD

注意負數的 case!

```cpp
ll gcd(ll a, ll b)
{
    return b == 0 ? a : gcd(b, a % b);
}
```

## 3.3 Extended Euclidean Algorithm

$ax + by = gcd(a, b)$

```cpp
ll ext_gcd(ll a, ll b, ll *x, ll *y)
{
    if (a == 0) {
        *x = 0;
        *y = 1;
        return b;
    }

    ll x1, y1;
    ll gcd = ext_gcd(b % a, a, &x1, &y1);

    *x = y1 - b / a * x1;
    *y = x1;

    return gcd;
}
```

## 3.4 Leap year

```cpp
year % 400 == 0 || (year % 4 == 0 && year % 100 != 0)
```

## 3.5 Prime Generator

```cpp
bool is_prime[N];
vector<ll> primes;
void init()
{
    fill(is_prime, is_prime + N, true);
    for (int i = 2; i < N; i++) {
        if (is_prime[i] == true) {
            primes.push_back(i);
            for (int j = i * i; j < N; j += i)
                is_prime[j] = false;
        }
    }
}
```

## 3.6 STL quick reference

### 3.6.1 Map

```cpp
map<T1, T2> m; // iterable
void clear();
void erase(T1 key);
it find(T1 key); // <key, val>
void insert(pair<T1, T2> P);
T2& [](T1 key); // if key not in map, new key will be inserted with
    default val
it lower_bound(T1 key); // = m.end() if not found, *it = <key, val>
it upper_bound(T1 key); // = m.end() if not found, *it = <key, val>
```

### 3.6.2 Set

```cpp
set<T> s; // iterable
void clear();
size_t count(T val); // number of val in set
void erase(T val);
it find(T val); // = s.end() if not found
void insert(T val);
it lower_bound(T val); // = s.end() if not found, *it = <key, val>
it upper_bound(T val); // = s.end() if not found, *it = <key, val>
```

### 3.6.3 Algorithm

```cpp
// return if i is smaller than j
comp = [&](const T& i, const T& j) -> bool;
vector<T> v;
bool any_of(v.begin(), v.end(), [&](const T& i) -> bool);
bool all_of(v.begin(), v.end(), [&](const T& i) -> bool);
void copy(inp.begin(), in.end(), out.begin());
int count(v.begin(), v.end(), int val); // number of val in v
it unique(v.begin(), v.end()); // it - v.begin() = size
// after calling, v[nth] will be n-th smallest elem in v
void nth_element(v.begin(), nth_it, bin_comp);
void merge(in1.begin(), in1.end(), in2.begin(), in2.end(), out.begin(),
    comp);
// include union, intersection, difference, symmetric_difference(xor)
void set_union(in1.begin(), in1.end(), in2.begin(), in2.end(), out.
    begin(), comp);
bool next_permutation(v.begin(), v.end());
// v1, v2 need sorted already, whether v1 includes v2
bool includes(v1.begin(), v1.end(), v2.begin(), v2.end());
it find(v.begin(), v.end(), T val); // = v.end() if not found
it search(v1.begin(), v1.end(), v2.begin(), v2.end());
it lower_bound(v.begin(), v.end(), T val);
it upper_bound(v.begin(), v.end(), T val);
bool binary_search(v.begin(), v.end(), T val); // exist in v ?
void sort(v.begin(), v.end(), comp);
void stable_sort(v.begin(), v.end(), comp);
```