Algorithms

errantProgrammer

25 de octubre de 2024

Índice general

Capítulo 1

Standard Template Library

Es un conjunto de clases de plantillas que nos proporciona la implementación de estructuras de datos y algoritmo. También, proporciona los iteradores y functores que facilitan algoritmos y contenedores.

1.1. Contenedores

Los contenedores son un objeto que nos va a permitir almacenar otros tipos de objetos, están implementados por medio de plantillas, por que admiten diferentes tipos de objetos.

Capítulo 2

Bitwise Operation

2.1. AND

P	Q	P & Q
0	0	0
0	1	0
1	0	0
1	1	1

Código de ejemplo:

```
int a = 7, b = 4;
int p = a & b;
cout <<"a & b = " << p << endl;
```

Ejemplo de operacion:

$$\begin{array}{rr} & 111_2 \\ \& & 100_2 \\ \hline & 100_2 = 4 \end{array}$$

2.2. OR

P	Q	P | Q
0	0	0
0	1	1
1	0	1
1	1	1

Código de ejemplo:

```
int c = 7, d = 4;
int q = c | d;
cout << "c | d = " << q << endl;
```

Ejemplo de operacion:

$$\begin{array}{rr} & 111_2 \\ | & 100_2 \\ \hline & 111_2 = 7 \end{array}$$

2.3. XOR

P	Q	P ^Q
0	0	0
0	1	1
1	0	1
1	1	0

Código de ejemplo:

```
int e = 7, f = 4;
int r = e ^ f;
cout << "e ^ f = " << r << endl;
```

Ejemplo de operacion:

$$\begin{array}{rr} & 111_2 \\ \hat{} & 100_2 \\ \hline & 011_2 = 3 \end{array}$$

2.4. NOT

P	$\sim$ P
0	1
1	0

Código de ejemplo:

```
int g = 9;
int s = ~g , n = !g;
cout << "~g = " << s << endl;
cout << "!g = " << n << endl;
```

Ejemplo de operacion:

$$\begin{array}{r} \sim \quad 1001_2 \\ \hline 0110_2 = 6 \end{array}$$

El not lo que hace nos brinda el complemento a 2, del número, no confundir con el negador lógico ¡".

2.5. Left Shift

P	Q	P $\ll$ Q
1010	1	10100
1010	2	101000
1010	3	1010000
1010	4	10100000

Código de ejemplo:

```
int h = 0b101, i = 3;
int t = h << i;
cout << "h << i = " << t << endl;
```

Ejemplo de operacion:

$$\begin{array}{r} 111_2 \\ \ll \quad 2 \\ \hline 011_2 = 3 \end{array}$$

Ver que si hacemos ($\ll n$) es similar a multiplicar por 2^n.

2.6. Right Shift

P	Q	P $\gg$ Q
1010000	1	1010000
1010000	2	10100
1010000	3	1010
1010000	4	101

Código de ejemplo:

```
int j = 0b1010000,k = 3;
int u = j >> k;
cout << "i >> k = " << u << endl;
```

Ejemplo de operacion:

$$\begin{array}{r} 111_2 \\ \gg \quad 2 \\ \hline 011_2 = 3 \end{array}$$

Ver que si hacemos ($\gg n$) es similar a dividir por 2^n.

2.7. Complemento a 2

El **complemento a 2** de un número es la forma de como representamos números negativos de un número, para calcular su valor existen multiples formas:

- $C_2(N) = 2^n - N$
- $C_2(N) = C_1(N) + 1$

Pero la forma más sencilla que tenemos de calcularlo, es representando el número a binario, y desde el *bit menos significativo*, avanzamos hacia la izquierda hasta encontrar el primer 1, y a partir de este, invertimos ceros y unos.

2.8. Problemas

Resolución de algunos problemas:

Petr and A Combination Lock

```
/**
 * Date: 05/10/2024
 * URL: https://codeforces.com/problemset/problem/1097/B
 */

#include <bits/stdc++.h>

int main(){
    int n;
    std::cin >> n; //cantidad de angulos
    std::vector<int> angles(n);
    for (int i = 0; i < n; i++) std::cin >> angles[i];
    for (int mask = 0;mask < (1 << n);mask++){ // recorremos la mascara
        int total = 0;// suma total de los angulos
        for (int i = 0; i < n; i++){ // solo vamos a utilizar los n primeros bits
            if(mask & (1 << i)) // verificamos si el i-esimo bit esta prendido
                total += angles[i];
            else
                total -= angles[i];
        }
        if(total % 360 == 0){
            std::cout << "YES";
            return 0;//se acaba el programa
        }
    }
    std::cout << "NO";
    return 0;
}
```

Rock and Level

```
/**
 * Date: 06/10/2024
 * URL: https://codeforces.com/problemset/problem/1420/B
 */

#include <bits/stdc++.h>
#define Long long long int
```

```
void solve(void){
    int n;
    std::cin >> n;
    std::vector<Long> values(n);
    for (int i = 0; i < n; i++) std::cin >> values[i];// guardamos los valores
    //recorremos todos los bits, como nuestro numero maximo es 1e9 se requiere 30 bits
    // para su representacion
    Long ans = 0;
    for (int bit = 29; bit >= 0; bit--){
        Long cnt = 0;
        for (int i = 0; i < n; i++){
            // verificamos que el primer i-esimo bit este prendido
            // y que el i-esimo bit + 1 este apagado
            if(values[i] >= (1<<bit) && values[i] < (1 << (bit + 1)) ){
                cnt++;
            }
        }
        ans += cnt * (cnt - 1)/2;
    }
    std::cout << ans << std::endl;
}


int main(){
    std::ios_base::sync_with_stdio(false);
    std::cin.tie(0);
    std::cout.tie(0);
    int t;
    std::cin >> t;
    while(t--) solve();
    return 0;
}
```

A Mocha and Math

```
/**
 * Date: 06/10/2024
 * URL: https://codeforces.com/problemset/problem/1559/A
 */
#include <bits/stdc++.h>

int main(){
    std::ios_base::sync_with_stdio(false);std::cin.tie(0);std::cout.tie(0);
    int cnt;//cantidad de casos
    std::cin >> cnt;
```

```
    while(cnt--){
        int n;
        std::cin>>n; //cantidad de datos de entrada
        int result,m;
        std::cin >> result; // guardamos el primer elemento
        for (int i = 1; i < n; i++){
            std::cin >> m;
            result &= m;
        }
        std::cout << result <<  std::endl;
    }

    return 0;
}
```

Preparing Olympiad

```
/**
 * Date: 06/10/2024
 * URL: https://codeforces.com/problemset/problem/550/B
 */

#include <bits/stdc++.h>

int GetBit(int n, int i){
    return ((n>>i) & 1);
}

int main(){
    std::ios_base::sync_with_stdio(false);
    std::cin.tie(0);
    std::cout.tie(0);
    /**
     * n = cantidad de problemas
     * l = dificultad minima
     * r = dificultad maxima
     * x = diferencia maxima entre el problema más dificil y el más facil.
     */
    int n, l, r, x;
    std::cin >> n >> l >> r >> x;
    std::vector<int> problems(n); // dificultad de los problemas
    for (int i = 0; i < n; i++) std::cin >> problems[i];
    int result=0;
    for (int mask = 0; mask < (1 << n); mask++){
        if(__builtin_popcount(mask) == 1) continue; // no nos sirve esta combinacion
```

```
        int min = INT_MAX, max =INT_MIN,sum = 0;
        for (int bit = 0; bit < n; bit++){//evaluamos bit a bit
            if( GetBit(mask,bit) ){ // el bit esta prendido
                max = problems[bit] > max ? problems[bit] : max;
                min = problems[bit] < min ? problems[bit] : min;
                sum += problems[bit];
            }
        }
        if(l <= sum and sum <= r and max - min >= x) result++;
    }
    std::cout << result << std::endl;
    return 0;
}
```