ALGORITMOS: AYUDANTÍA 4

Ayudante: Yerko Ortiz

Objetivo de la ayudantía: Reforzar análisis de complejidad algorítmica, entendimiento de stacks y ver un ejemplo de uso de union-find disjoint-set.

Compilador mental

Compile y ejecute los siguientes algoritmos a mano para las entradas correspondientes, posteriormente usando la notación **O(f(n))** describa la complejidad del algoritmo en cuestión.

Algoritmo 1

```
public static int algoritmo1(int arr[], int n)
{
    int cont = 0;
    for (int i = 0; i < n; i ++)
        for (int j = i, tmp = 0; j < n; j ++) {
            tmp += arr[j]
            cont += tmp ;
        }
    return cont;
}
```

- **Input:**
  arr = {1, 2, 3}
  n = 3
- **Complejidad:**
- **Output:**

Algoritmo 2

```
public static int algoritmo2( int arr[] , int n )
{
    int cont = 0;
    for (int i=0; i < n; i++)
        cont += (arr[i] * (i+1) * (n-i));
    return cont;
}
```

- **Input:**
  arr = {1, 2, 3, 4}
  n = 4
- **Complejidad:**
- **Output:**

"I think that I shall never see A poem lovely as a tree."

Joyce Kilmer

Algoritmo 3

```java
static boolean algoritmo3(int arr[], int n, int key)
{
    int k = 0;
    for(int b = n/2; b >= 1; b /= 2)
        while(k + b < n &&  arr[k + b] <= key) k += b;
    if(arr[k] == key) return true;
    return false;
}
```

- **Input:**
  arr = {1, 2, 7, 10, 24}
  n = 5
  key = 10
- **Complejidad:**
- **Output:**

Diseño de algoritmos

Diseñe e implemente algoritmos en Java para los siguientes enunciados, describa la complejidad de su solución en términos de O(f(n)).

UFDS

Sea un conjunto de N personas enumeradas del 0 hasta el N - 1; del conjunto de personas algunas son amigas entre sí. Un grupo de amigos se define como todas las personas que son *amigas o que tienen amigos en común*, por ejemplo: la persona 0 es amiga de la persona 1, la persona 1 es amiga de la persona 3, pero la persona 2 no es amiga ni de 0, 1, 3, por lo que se dice que en ese conjunto hay dos grupos de amigos: grupo 1 {0, 1, 3} y grupo 2 {2}. Diseñe e implemente un programa en Java que permita saber si dos personas pertenecen al mismo grupo de amigos o no.

Entrada

- La primera linea consiste en un entero N que representa la cantidad de personas enumeradas del 0 hasta N - 1.
- La segunda linea consiste en un entero K que representa la cantidad de personas que son amigas en el grupo.
- Las subsecuentes K lineas contienen dos enteros u, v, que representan que la persona u es amiga con la persona v.
- La ultima linea contiene dos enteros x e y, que representan la consulta(saber si la persona x y la persona y pertenecen al mismo grupo de amigos).

Dominio

- $1 \leq N \leq 100$
- $0 \leq K \leq \frac{(N-1)^2+(N-1)}{2}$
- $0 \leq u, v, x, y \leq N - 1$

Salida

- Un booleano B que representa si la persona x y la persona y pertenecen al mismo grupo de amigos

> *"Time-travel is possible, but no person will ever manage to kill his past self. Godel laughed his laugh then, and concluded, The a priori is greatly neglected. Logic is very powerful."*
>
> Kurt Gödel

Caso de prueba

- **Input:**
  5
  2
  0 1
  1 2
  1 3
- **Output:**
  0

STACK

Sea un arreglo de enteros, imprima el máximo elemento a la derecha para todo elemento en el arreglo. El máximo elemento a la derecha(Next Greater Element) para un entero x es el **primer** entero mayor que a la derecha en el arreglo. En caso de no encontrar NGE(x) retorne -1.

Entrada

- La primera linea contiene un entero N que representa el tamaño del arreglo.
- La segunda linea contiene N enteros x_i separados por un espacio, que representan a el arreglo

Dominio

- $1 \leq N \leq 10^6$
- $0 \leq x_i \leq 10^6$

Salida

- N enteros que corresponden al NGE(x_i), para todo elemento x_i en el arreglo, de izquierda a derecha

Caso de prueba

- **Input:**
  4
  4 5 2 25
- **Output:**
  5 25 25 -1

Gracias por su atención!