**TAREA 4**

**Tarea IV: parte I** Para los siguientes problemas diseñe el algoritmo (GCL con pre y postcondición), seleccione estructuras de datos a usar y especifique su complejidad en tiempo (exacta y asintótica).

1. Dado un conjunto de números entregados en forma de arreglo de tamaño 𝑛, determine el elemento que es la moda (elemento que ocurre el mayor número de veces).

fun modaHallar ( A: **array** of int(0,N) ) : int

var M : **HashMap** of int, int

var S : **Array** of int

var i, j, k, mayor: int

{Q:true}

i,j,k,moda;=0,0,0; 🡪 **c1**

do i< N → 🡪 **c2(n+1)**

if M.containsKey(A[i]) = false → M.put(A[i],1); 🡪 **n(c4 + c5 + c2 + c6 + c4)**

[] M.containsKey(A[i]) = true → j := M.get(i) + 1;🡪 **n(c5 +c2 + c1 + c2 + c7)**

M.put(A[i], j); **n(c6 + c4)**

fi

i := i+1; 🡪**n(c1+c3)**

od

S:=M.keySet() 🡪 **c8**

do k< N → 🡪 **c2(n+1)**

if M.get(S(k)) > M.get(moda) → moda := S(k); **n(c7+c4+c7+c1+c4)**

fi

k := k+1; **n(c1+c3)**

od

{R:””}

ret moda

* Complejidad exacta

c4 = acceder a un elemento de un array

c5 = containsKey hashmap

c6 = put hash map

c7 = get hash map

c8 = keyset hashmap

c1 + c2 + c2 + **n(c4 + c5 + c2 + c2 + c6 + c4) + n(c5 +c2 + c1 + c2 + c7) + n(c6 + c4) + n(c1+c3) + c8 + c2(n+1) + n(c7+c4+c7+c1+c4) + n(c1+c3)**

**= c1 + 2c2 + c8 + n(5c2+4c4+2c5+2c6+2c3+3c1)**

**=k1+nk2**

**=O(n)**

* Complejidad Asintótica

Es posible ver que el algoritmo puede ser , solo basta con encontrar un c>k2 y un c que podría ser c = 1/k2 para que pueda ser ambas.

Con este tenemos que es .

2. Dado un conjunto de números entregados en forma de arreglo de tamaño 𝑛, determine el elemento que es la mediana (si hay un número par de datos, la mediana es el promedio de los dos números medios).

fun medianaHallar ( A: array of int(0,N) ) : float

var M : array of int

var length, indexMediana: int

var mediana: float

{Q:true}

length,indexMediana;=0,0;

M := sorted(A);

length := A.length

indexMediana := (length - 1) intdiv 2

if length mod 2 = 0 → mediana:=M[indexMediana]

[] length mod 2 0 → mediana:=(M[indexMediana] + M[indexMediana+1]) div 2

fi

{R:””}

ret mediana

3. Diseñe (3) algoritmos para determinar si dos conjuntos (de tamaño 𝑚 y 𝑛) son disjuntos. Analizando la complejidad de los algoritmos en términos de 𝑚 y 𝑛, deberían tener las siguientes complejidades:

▪ 𝑂 (𝑛log(𝑛)+ 𝑚𝑙𝑜𝑔(𝑛))

▪ 𝑂 𝑛 log(𝑚) + 𝑚𝑙𝑜𝑔(𝑚)

▪ 𝑂(𝑛) en el caso esperado.

Pista: Use tabla hash y analice una posible relación de orden entre n y m.

**Tarea IV: parte II** ▪ Resuelva las siguientes ecuaciones de recurrencia usando el método maestro:

▪

Caso 1, ∈ = 1

▪

Caso 2

▪

Caso 3, ∈ = ½