Examen de Algorítmica y Complejidad (Plan 2014) **5 de diciembre de 2020**

Nº matrícula:________________**Nombre:** ____

Apellidos: ____

Problema. Se dice que un array, *v*, es de tipo colina si existe un índice *n* tal que: para todo i<*n* se cumple que v[i]<v[i+1]; y para todo i>*n* se cumple que v[i-1]>v[i]. Dado un *array* de tipo colina, queremos encontrar su máximo. Ejemplo:

0	1	2	3	4	5	6	7	8
5	**7**	**8**	**9**	**3**	**2**	**1**	**0**	**-7**

El máximo del array es 9.

a) Diseñar un algoritmo basado en *Divide y Vencerás* con complejidad O(log *N*) en el caso peor[1] (donde *N* es el tamaño del vector) que devuelva el máximo de un array de tipo colina.

```
int maxArrayColina(int[] vector)
```

```
int maxArrayColina(int[] vector){
     return maxArrayColinaAux(vector,0,vector.length-1);
}

int maxArrayColinaAux(int[] vector, int i0, int iN){
     if (i0==iN)
         return vector[i0];
     else if (iN==i0+1)
         return Math.max(vector[i0],vector[iN]);
     else{
         int k=(i0+iN)/2;
         if (vector[k]<vector[k+1])
             return maxArrayColinaAux(vector,k+1,iN);
         else if (vector[k - 1] < vector[k])
             return vector[k];
         else
             return maxArrayColinaAux(vector,i0,k);
     }
 }
```

[1] Desarrollar un algoritmo que tenga una complejidad diferente a O(log *N*) en el caso peor conllevará una puntuación de 0 en la pregunta.

b) Calcula la complejidad del algoritmo desarrollado en el apartado anterior para el caso mejor y peor indicando qué condiciones tiene que cumplir el vector para estar en esos casos mejor y peor.

* El caso mejor sucede cuando el máximo del vector se encuentra justo en i=N/2. En ese caso el algoritmo no realiza ninguna llamada recursiva, y la complejidad es O(1).

* El caso peor sucede cuando el máximo del vector se encuentra en uno de los extremos del vector (por ejemplo en la posición 0 del vector). En ese caso, el algoritmo implementado obedece a la siguiente ecuación de recurrencia en el tiempo:

$$T(N) = T(N/2) + O(1) \text{ para } N>2$$

Esta ecuación es del tipo $T(N) = p \cdot T(N/q) + f(N)$, donde $f(N) \in O(N^a)$, con p=1, q=2 y a=0, por lo que podemos aplicar el Teorema Maestro. Dado que $\log_q(p) = \log_2(1)=0$ y a=0 nos encontramos en el caso 2º del Teorema maestro ($a=\log_q(p)$), por lo que la complejidad del algoritmo es: $T(N) \in O\left(N^{\log_q(p)} \cdot \log N\right) = O(N^0 \log N) = O(\log N)$