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

Nº matrícula:_____ **Nombre:** _____

Apellidos: _____

1) **Problema 1** (**5 puntos**). 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) (**3 puntos**) 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);
    }
}
```

b) (**2 puntos**) 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 Θ(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:

[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.

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

Aplicando el teorema maestro para el caso 2 (ya que $\log_2 1=0$), la complejidad del algoritmo es:

$$\Theta(\log N)$$