Examen de Algorítmica y Complejidad (Plan 2014) | **7 de noviembre de 2017**

Nº matrícula:_____ **Nombre:** _____

Apellidos: _____

Problema. Dado un *array*, **vector**, de números enteros **ordenados de menor a mayor** y **diferentes entre sí**. Encontrar un índice *i* tal que **vector[i]==i** (en caso de existir varios elementos que cumplan la condición bastará con devolver uno de ellos).

Ejemplo:

0	1	2	**3**	4	5	6	7	8	9	10
-10	-2	0	**3**	7	9	19	28	30	42	55

Para *i*=3, se tiene que **vector[3]==3**

a) Implementar un algoritmo en Java, basado en *Divide y Vencerás,* que solucione el problema expuesto, con **complejidad O(log *N*)** en el caso peor[1] (donde *N* es el tamaño del vector).

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

int elementoEspecialAux(int[] vector,  int i0, int iN){
     if (i0 == iN)
         if (vector[i0] == i0) return i0;
         else   return -1;
     else {
         int k = (i0 + iN)/ 2;
         if (vector[k]> k)
             return elementoEspecialAux(vector, i0, k);
         else if (vector[k] == k)
             return k;
         else
             return elementoEspecialAux(vector, k + 1, iN);
     }
 }
```

[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) Justifica que la complejidad del algoritmo desarrollado en el apartado anterior para el caso peor es O(log N).

El algoritmo implementado obedece a la siguiente ecuación de recurrencia en el tiempo para N>1:

$$T(N) = T(N/2) + O(1)$$

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: $T(N) \in O\left(N^{\log_q(p)} \cdot \log N\right) = O(N^0 \log N) = O(\log N)$