Examen de Algorítmica y Complejidad (Plan 2014) — **16 de octubre de 2023**

Nº matrícula:_____ **Nombre:** _____

_____ **Apellidos:** _____

_____

1) Problema (2.5 puntos). Dado un vector de enteros ordenados mayores que cero, se quiere obtener cuántos múltiplos de una cifra *x* hay en el vector (para cualquier *x>0*). Por ejemplo, dado el siguiente vector con *x=3*, la salida sería 2 porque encuentra 2 múltiplos de 3.

0	1	2	3	4	5	6
1	**1**	**2**	**4**	**6**	**17**	**18**

a) (2 puntos) Diseña e implementa en Java un algoritmo basado en la estrategia de **Divide y Vencerás** con complejidad en el caso peor[1] de **O(N)** (donde N es el tamaño del vector) que compute el problema anterior. Se pide implementar un algoritmo que deberá tener la siguiente cabecera:

```
int numMultiplos(int[] v, int x)
```

Aclaraciones: Se podrán implementar todos los métodos adicionales que se consideren necesarios.

```
int numMultiplos(int[] v, int x){
    return numMultiplosAux(v, 0, v.length-1, x);
}
int numMultiplosAux(int[] v, int i0, int iN, int x) {
    if(i0==iN){
        // Caso base: solo si el objetivo es divisor del elemento en i0, devolvemos 1
        if(v[i0]%x == 0) return 1;
        else return 0;
    }else{
        // Dividimos el vector en 2
        int k = (i0+iN)/2;
        // Conquistar: la parte derecha siempre hay que resolverla, pero la izquierda
        // puede ser prescindible cuando v[k] < x.
        int p0 = 0;
        if(v[k]>=x) p0 = numMultiplosAux(v, i0, k, x);
        int p1 = numMultiplosAux(v, k+1, iN, x);
        // Combinar, como la suma de las dos partes
        int comb = p1+p0;
        return comb;
    }

}
```

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

b) (0.5 puntos) Justifica que la complejidad del algoritmo desarrollado en el apartado anterior para el caso peor es de O(N).

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

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

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