Práctica 1. Módulo Divide y Vencerás.

Decimos que un *array v* de *N* enteros está ordenado circularmente si, o bien el vector está ordenado, o bien $v[N-1] \leq v[0]$ y $\exists k$ con $0<k<N$ tal que $\forall i \neq k\ v[i] \leq v[i+1]$ (esto es, está ordenado imaginando que fuera un *array* circular). Ejemplo:

0	1	2	3	4	5	6	7	8
-1	**0**	**2**	**3**	**10**	**12**	**-23**	**-14**	**-7**

Dado un vector con 10 elementos positivos a lo sumo, ordenado circularmente, deseamos calcular la suma de sus valores positivos. En el ejemplo anterior, este cálculo devolvería 27.

SE PIDE:

Apartado 1. En este apartado se pide:

- Implementar un algoritmo en Java, basado en el esquema divide y vencerás, que resuelva el problema. El algoritmo implementado deberá tener una complejidad O(*N*) en el caso peor, donde *N* es la longitud del vector. (**3 puntos**)
- Entregar una memoria que contenga los siguientes apartados:
  - El funcionamiento del algoritmo implementado. (**1 punto**)
  - Demostración de que la complejidad del algoritmo es O(*N*) en el caso peor, identificando qué condición debe cumplir el vector para que se dé ese caso peor. (**2 puntos**)

Apartado 2. En este apartado se pide:

- Implementar un algoritmo en Java, basado en el esquema divide y vencerás, que resuelva el problema. El algoritmo implementado deberá tener una complejidad O(log *N*) en el caso peor considerando que en el vector no hay elementos repetidos. (**1 punto**)
- Entregar una memoria que contenga los siguientes apartados:
  - El funcionamiento del algoritmo implementado. (**1 punto**)
  - Demostración de que la complejidad del algoritmo es O(log *N*) en el caso peor, identificando qué condición debe cumplir el vector para que se dé ese caso peor. (**2 puntos**)

Nota: Es requisito para la corrección de la práctica que el programa entregado compile sin errores, funcione correctamente, y siga el esquema divide y vencerás.

DESARROLLO:

Se proporcionan dos clases: *Pruebas* (clase lanzadera para probar el algoritmo) y *Principal* (clase donde hay que implementar el algoritmo pedido).

Pruebas.Java

Esta clase proporciona una lanzadera para realizar las pruebas con las que poder verificar que la codificación realizada por el alumno cumple con los requisitos.

Principal.Java

Esta clase contiene los dos métodos correspondientes a la implementación de los dos apartados.

public static int sumaPositivos1(int[] vector)
La cabecera del método a implementar para el primer apartado.

public static int sumaPositivos2(int[] vector)
La cabecera del método a implementar para el segundo apartado.

***NORMAS DE ENTREGA*:**

La entrega consistirá en subir a la plataforma Moodle un archivo en formato comprimido (ZIP o RAR) que incluya un fichero .pdf de la memoria donde se encuentre explicado el algoritmo y las ecuaciones de recurrencia y el fichero Principal.java donde se encuentre implementado el algoritmo.

Si la práctica se ha desarrollado individual, el nombre del fichero comprimido deberá ser el número de matrícula del alumno (ejemplo: ai0260.zip). Si la práctica se ha desarrollado en pareja con otro compañero, el nombre del fichero comprimido deberá ser el número de matrícula de ambos (ejemplo: ai0340-bc0433.zip). En este último caso, **AMBOS** alumnos deberán subir la práctica al Moodle.

El archivo comprimido se deberá subir a la plataforma Moodle **antes de las 23:55 horas del 13 de octubre de 2019**. Aquellos alumnos que hayan subido la práctica deberán realizar un examen escrito de la misma el día **21 de octubre de 2019**. Para poder aprobar esta práctica será indispensable haberla subido a la plataforma, haber superado el examen de la misma y haberla desarrollado correctamente.