

TECNICAS DE PROGRAMACION AVANZADAS

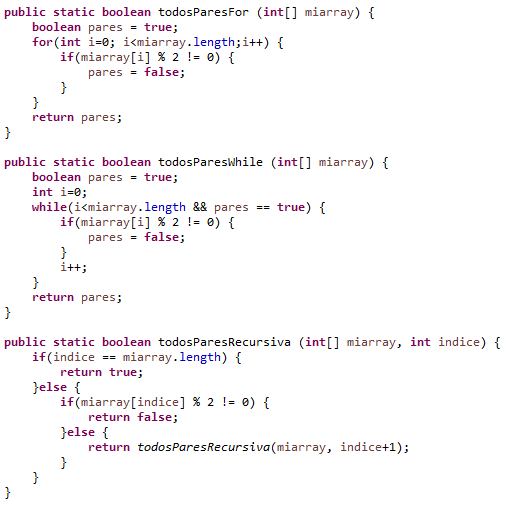
DIGITAL BLOCK 1



VICTOR PEREZ PEREZ

23/02/2021

**Código de las funciones**



**Tabla de eficiencia**



**Análisis de resultados**

**Bucle FOR**: este bucle tiene el mismo orden de ejecución para todos los casos, es decir, no importa que el primer numero sea impar y el resto pares, ya que se va a recorrer el array entero. En cuanto al uso de memoria, hace un uso bastante eficaz, ya que no hace copia de los elementos cada vez que comprueba una posición.

**Bucle WHILE**: este bucle tiene una condición que le hace mejor, salvo en el caso peor, del bucle for. Esta condición hace que, cuando el bucle encuentra un numero impar, sale de la función y no llega a recorrer el array entero. Su uso de memoria es similar al del bucle for, ya que cada vez que recorre una posición del array, no hace copia de ninguno de los elementos.

**RECURSIVA**: esta función, al igual que el bucle while, se va a ejecutar hasta que encuentre el primer numero impar del array, y en ese momento el programa sale de la función. Sin embargo, el uso de la memoria que tiene la recursividad es memos eficaz que los bucles anteriores, ya que cada vez que recorre una posición del array, se van a hacer copias de las variables de la función.

**Mi elección**

Una vez analizada la tabla y estudiadas las características de cada una de las funciones, para mí, la mejor opción seria utilizar un **bucle** **while**. Con el bucle while, vamos a recorrer el array hasta encontrar el primero que sea impar, y en ese momento salimos de la función avisando que hay un elemento impar en el array. Además, es una forma muy sencilla a la hora decodificar, y en comparación con los otros bucles, nos ahorramos mucha memoria que, si utilizamos la recursividad, y tardamos menos tiempo que el bucle for (quitando el peor de los casos).