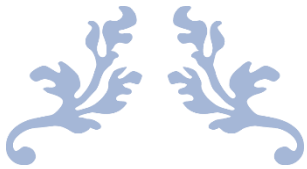


TECNICAS DE PROGRAMACION AVANZADAS

DIGITAL BLOCK 1

VICTOR PEREZ PEREZ

23/02/2021

Bucle For

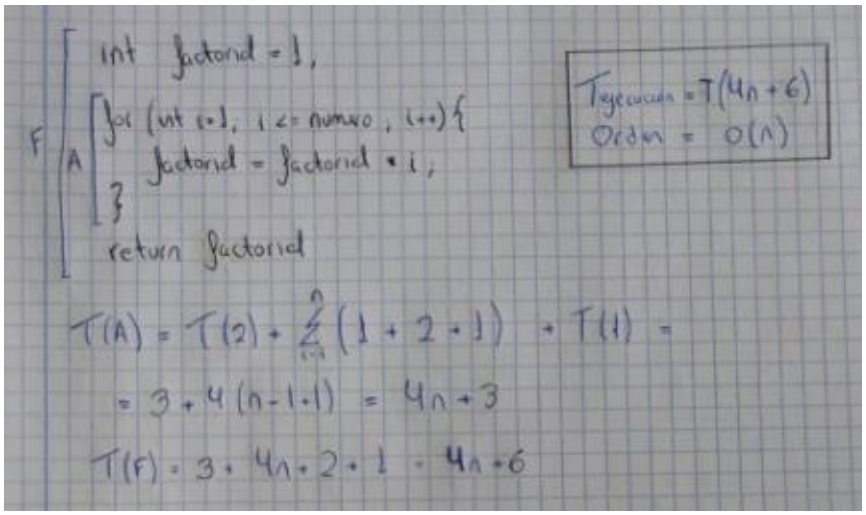
```java
//Bucle for
public static int factorialFor(int numero) {
    int factorial = 1;

    for (int i=1; i<=numero; i++) {
        factorial = factorial * i;
    }

    return factorial;

}
```

Operaciones para calcular el tiempo y el orden de ejecución



T(A) = T(declaración e inicialización)+ $\sum_{i=1}^{n}$(T(condición) + T(cuerpo) + T(incremento)) + T(condición)

T(F) = T(declaración e inicialización) + T(bucle For) + T(return)

Solución:

- Tiempo de ejecución = T(4n + 6)
- Orden de ejecución = O(n)

Bucle While

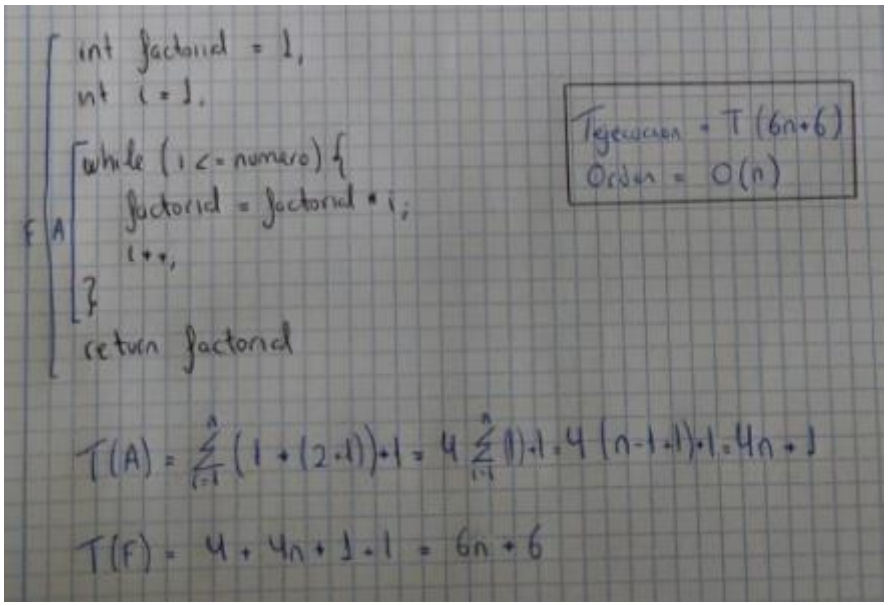
```java
//Bucle while
public static int factorialWhile(int numero) {
    int factorial = 1;
    int i = 1;

    while (i<=numero) {
        factorial = factorial * i;
        i++;
    }

    return factorial;

}
```

Operaciones para calcular el tiempo y el orden de ejecución



T(A) = $\sum_{i=1}^{n}$(T(condición) + T(cuerpo)) + T(condición while)

T(F) = T(declaración e inicialización) + T(bucle While) + T(return)

Solución:

- Tiempo de ejecución = T(4n + 6)
- Orden de ejecución = O(n)

Recursividad

```
//Recursiva
public static int factorialRecursiva(int numero) {

    if(numero == 0) {
        return 1;
    }else {
        return numero * factorialRecursiva(numero-1);
    }

}
```

$$T(n) = \begin{cases} C_0 n^{k0} \rightarrow 2 & \rightarrow k_0 = 0 \\ a*T(n-b) + C_1 n^{k1} \rightarrow 1*T(n-1) + 2n^{k1} & \rightarrow k_1 = 0 \end{cases}$$

$T(n) = n-1+2+2 \rightarrow T(n) = n + 3$

Solución:

- Tiempo de ejecución = T(n + 3)
- Orden de ejecución = O(n)

TABLA COMPARATIVA DE TIEMPO, ORDEN Y GESTION DE MEMORIA

	Tiempo ejecucion	Orden ejecucion	Gestion de memoria
FOR	4n + 6	n	Buena
WHILE	4n + 6	n	Buena
RECURSIVA	n + 3	n	Mala

En este ejercicio, no tenemos ningún caso mejor ni peor como seria por ejemplo si queremos buscar un número en un array. Podemos observar que todas las funciones tienen orden de ejecución O(n), y que el bucle for y el while, son similares. Además, estos dos últimos bucles gestión mejor la memoria ya que no tienen que hacer copia de los elementos cuando se vuelve a llamar a la función.

Creo que en este caso seria mejor utilizar tanto el bucle for como el while, ya que tienen las mismas instrucciones, y al no existir un caso mejor, siempre van a tardar lo mismo en ejecutarse. Además, su gestión de memoria es mejor que si lo hacemos de forma recursiva.