

Universidad Autónoma de Santo Domingo (UASD)

Nombre: Ismael Bernardo Nuñez Feliz

Matrícula: 100284129

# Asignatura: INF5260 Estructura De Datos

Sección: Z01

# Maestro Asignado: Silverio Del Orbe Abad

## **Tema**: Tarea 4.2

## **Fecha de Entrega**:Domingo, 15 de octubre de 2023

Tome las siguientes funciones y saque sus polinomios de crecimiento:

-- mejor escenario

-- peor escenario

-- y ver si puede optimizar el algoritmo, para bajar el grado o coeficiente del algoritmo.

1) **Buscar el factorial de un numero mediante iteración. HACIA DELANTE**

int getFactor(int Num)

{

    int F = 1;                         //O(1)

    for (int k = 2; k <= Num; k++)     //O(n)

    {

        F \*= k;                        //O(n)

    }

    return F;                          //O(1)

}

## Peor escenario:

**El polinomio**: 2n + 2 = **O(n)**

## Mejor escenario:

**El polinomio**: **O(1)**

El mejor de los casos es que Num sea igual a 1.

## Optimizar:

No se ha encontrado una forma de optimizar.

2) **Buscar el factorial de un numero mediante recursión. HACIA detras**

int getFarctor(int Num)

{

    if (Num <= 0)                         //O(1)

        return 1;

    else

        return Num \* getFactor(Num - 1);  //O(n-1) \* O(1) = O(n)

}

## Peor escenario:

**El polinomio**: 2n + 2 = **O(n)**

## Mejor escenario:

**El polinomio**: **O(1)**

El mejor de los casos es que Num sea igual a 1.

## Optimizar:

No se ha encontrado una forma de optimizar.

3) **Fibonacci por recursión**

private long fibonacci(long d) {

        if (d < 2)

            return d;                                   //O(1)

        else

            return  fibonaci(d - 2) + fibonaci(d - 1);//(n-2)\*(n-1)= O(n^2)

    }

## Peor escenario:

**El polinomio**: O(n^2) + O(1) = **O(n^2)**

## Mejor escenario:

**El polinomio**: **O(1)**

El mejor de los casos es que Num sea igual a 0 o 1.

## Optimizar:

No se ha encontrado una forma de optimizar.

4) **Fibonaci por iteracion**

 private long fibonaciIter(long d)

        {

            long fibo0 = 0;                 //O(1)

            long fibo1 = 1;                 //O(1)

            long tp=0;                      //O(1)

            for (int k = 0; k < d; k++) {   //O(n)

                tp = fibo0 + fibo1;         //O(n)

                fibo0 = fibo1;              //O(n)

                fibo1 = tp;                 //O(n)

            }

            return tp;                      //O(1)

        }

## Peor escenario:

**El polinomio**: 4n + 4 = **O(n)**

## Mejor escenario:

**El polinomio**: **O(1)**

El mejor de los casos es que Num sea igual a 0 o 1.

## Optimizar:

No se ha encontrado una forma de optimizar.