UNIDAD TEMÁTICA 2: DISEÑO Y ANÁLISIS DE ALGORITMOS

PRACTICOS DOMICILIARIOS INDIVIDUALES 3

Realiza y documenta detalladamente la resolución de los siguientes ejercicios del capítulo 5 “Análisis de Algoritmos” del libro “Estructuras de Datos en Java” de Mark Allen Weiss:

- 5.4,
- 5.5,
- 5.6,
- 5.10,
- 5.11,
- 5.12,
- 5.13,
- 5.14,
- 5.15,
- 5.16

5.4 Suponga que $T_1(N) = O(F(N))$ y $T_2(N) = O(F(N))$. ¿Cuáles de las siguientes afirmaciones son ciertas?

a. $T_1(N) + T_2(N) = O(F(N))$ ←

b. $T_1(N) - T_2(N) = O(F(N))$ ← Son Verdaderas

c. $T_1(N) / T_2(N) = O(1)$

d. $T_1(N) = O(T_2(N))$

5.5 Resolver un problema requiere ejecutar un algoritmo $O(N)$ y después un segundo algoritmo $O(N)$. ¿Cuál es el coste total de resolver el problema? →

Sigue siendo $O(N)$ ya que:

$$O(N) + O(N) = 2O(N) = O(N)$$

5.6 Agrupe las siguientes funciones según su equivalencia desde el punto de vista del análisis O mayúscula:

$\cancel{x}, \cancel{x^2}, \cancel{x^3 + x}, \cancel{x^2 - x}$ y $(x^4 / (x-1))$

Grupo Cuadrático → $x^2, x^2 - x$

Grupo Lineal → x

Grupo Cúbico → $x^3 + x, (x^4/(x-1))$

N	Figura 5.4 $O(N^3)$	Figura 5.5 $O(N^2)$	Figura 7.20 $O(N \log N)$	Figura 5.8 $O(N)$
10	0,000001	0,000000	0,000001	0,000000
100	0,000288	0,000019	0,000014	0,000005
1.000	0,223111	0,001630	0,000154	0,000053
10.000	218	0,133064	0,001630	0,000533
100.000	218000	13,17	0,017467	0,005571
1.000.000	218.000.000	131.770	0,185363	0,056338

Figura 5.10 Tiempos de ejecución (en segundos) observados para varios algoritmos de suma máxima de subsecuencia contigua.

5.10 Complete la Figura 5.10 con estimaciones para los tiempos de ejecución que eran demasiado largos como para simularlos. Interpole los tiempos de ejecución para los cuatro algoritmos y estime el tiempo requerido para calcular la suma máxima de subsecuencia contigua de 10.000.000 de números. ¿Qué suposiciones ha hecho?

$$\frac{218}{10.000^3} = \frac{x}{100.000^3} = 218.000$$

$$\frac{218}{10.000^3} = \frac{x}{1000000^3} = 218000000$$

$$\frac{13.17}{100.000^2} = \frac{x}{1000000^2} = 131.770$$

5.11 Evalúe directamente el sumatorio triple que precede al Teorema 5.1. Verifique que las respuestas son idénticas. } ??

5.12 Un algoritmo requiere 0,4 ms para un tamaño de la entrada de 100. ¿Cuánto tiempo requerirá para un tamaño de entrada igual a 500 (suponiendo que los términos de menor orden sean despreciables), si el tiempo de ejecución es:

a. lineal? → 2,0 ms

b. $O(N \log N)$? → 2,7 ms

c. cuadrático? → 10 ms

d. cúbico? → 50 ms

Lineal

$$\frac{0.4}{100} = \frac{x}{500} = \frac{2}{500}$$

Logaritmico

$$\frac{0.4}{100 \log_{10} 100} = \frac{x}{500(\log_{10} 500)} = 2.7$$

Cuadratico

$$\frac{0.4}{100^2} = \frac{x}{500^2} = \frac{10}{500^2}$$

Cubico

$$\frac{0.4}{100^3} = \frac{x}{500^3} = 50$$

5.13 Para los algoritmos típicos que emplee para realizar cálculos a mano, determine el tiempo de ejecución necesario para

a. Sumar dos enteros de N dígitos.

b. Multiplicar dos enteros de N dígitos.

} Lineal

5.14 Para el algoritmo cuadrático correspondiente al problema de la suma máxima de subsecuencia contigua, determine de forma precisa cuántas veces se ejecuta la instrucción más interna.

```
/**
 * Algoritmo de suma máxima de subsecuencia contigua.
 * seqStart y seqEnd representan la mejor secuencia actual.
 */
public static int maxSubsequenceSum( int [ ] a )
{
    int maxSum = 0;

    for( int i = 0; i < a.length; i++ )
    {
        int thisSum = 0;

        for( int j = i; j < a.length; j++ )
        {
            thisSum += a[ j ];

            if( thisSum > maxSum )
            {
                maxSum = thisSum;
                seqStart = i;
                seqEnd = j;
            }
        }
    }

    return maxSum;
}
```

Figura 5.5 Un algoritmo cuadrático de suma máxima de subsecuencia contigua.

La sumatoria de 0 hasta i $\left(\sum_{n=0}^{i} n\right)$ es la cantidad de veces que se ejecuta la linea