EJERCICIO 2

En este caso el tamaño del problema vendrá dado por el valor de m.

Este algoritmo suma una serie de números que son potencias de 3, su ejecución depende de m y de operaciones aritméticas fijas, sin ningún tipo de condición. Por tanto, el algoritmo NO presentará caso mejor y peor.

En la primera iteración del bucle externo, $i = 1$ (el bucle interno realiza una iteración). En la segunda, $i = 3$ (el bucle interno realiza tres iteraciones). Así sucesivamente hasta que $i > m$.

Por tanto, el total de operaciones (iteraciones del bucle interno) será:

$$T(m) = 1 + 3 + 9 + \dots + 3^{k-1} \longleftarrow k = \text{total operaciones bucle externo}$$

$$\downarrow$$

$$T(m) = \sum_{k=1}^{k} 3^{k-1}$$

Si igualamos con la m-ésima iteración $\rightarrow i = 3^{k-1} \longrightarrow k = \log_3(m) + 1$, de lo que deducimos que $3^k \in \Theta(m)$.

Estamos ante una progresión geométrica con primer término 1, razón 3 y k elementos:

$$S_k = a_1 \cdot \frac{r^k - 1}{r - 1} \longrightarrow S_k = 1 \cdot \frac{3^k - 1}{3 - 1} = \frac{3^k - 1}{2} = T(m)$$

Por tanto:

Propiedad $b^{\log_b x} = x$

$$T(m) = \frac{3^{\log_3 m} - 1}{2} = \frac{m - 1}{2} \in \Theta(m)$$

EJERCICIO 3

El tamaño del problema viene dado por el tamaño del vector, es decir, el número de elementos. Por tanto diremos que $n = v.size()$.

Este algoritmo es similar a BubbleSort solo que empleando un flag para detectar si en cada iteración se ha realizado un cambio o no. Puesto que trabaja con vectores y dada su similitud con BubbleSort, el algoritmo presentará caso mejor y peor, dependiendo de las veces que haya que hacer cambios en el vector.

CASO MEJOR ⟶ Vector ordenado de forma creciente

En este caso nunca se ejecutará el if y tras completar una vez el bucle for se romperá el while ya que el flag swaped será falso. De modo que se harán unas $n-1$ comparaciones, que podemos simplificar como n.

Complejidad: $\boxed{\Omega(n)}$

CASO PEOR ⟶ Vector ordenado de forma decreciente o desordenado de forma que siempre haya intercambios.

En este caso en cada iteración hay intercambios y el flag se activa, obligando a repetir el bucle. El contador i se incrementa en cada iteración, de modo que en la primera pasada se hacen $n-1$ comparaciones, en la segunda $n-2$, y así sucesivamente.

Esto se puede expresar como:

Límite superior

$$\sum_{i=1}^{n-1} (n-i) = \frac{n(n-1)}{2} \in O(n^2)$$

Valor inicial (primera pasada)

Número de comparaciones en la pasada