Universidad Del Valle de Guatemala
Departamento de Ciencias de la Computación
Teoría de la computación
Catedrático: Gabriel Brolo

Laboratorio 8

Gabriel Estuardo García Donis - 21352

Guatemala, 13 de octubre 2023

Ejercicio 1

```
void function (int n) {
        int i, j, k, counter = 0;                      -> O (1)
        for (i = n/2; i <= n; i++) {                   -> O (n)
                for (j = 1; j+n/2 <= n; j++) {         -> O (n)
                        for (k = 1; k <= n; k = k*2) { -> O (log2(n))
                                counter++;             -> O (log2(n))
                        }
                }
        }
}
```

$$f(n) = 1 + (n+2)(n/2)(\log_2(n))$$
$$= 1 + (n^2/4)\log_2(n)$$
$$f(n) = n^2 \log_2(n)$$

¿ $f(n) = O\,g(n)$?

$$f(n) \leq c \cdot g(n)$$

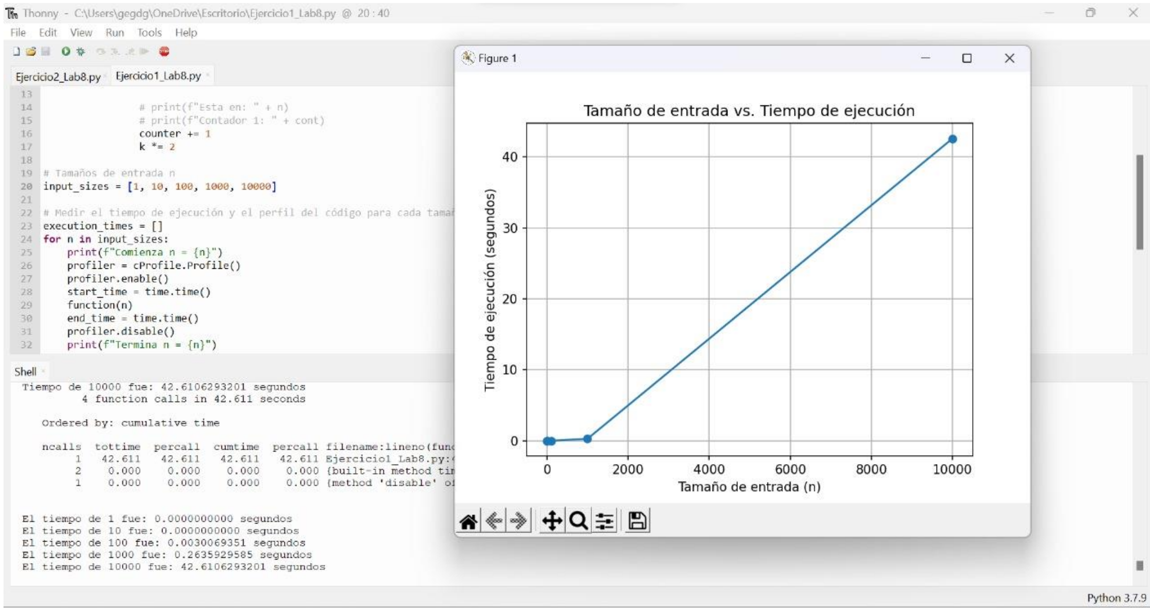
$$1 + (n^2/4)\log_2 n \leq c \cdot n^2 \log_2 n \quad \rightarrow c = 2$$
$$1 + (n^2/4)\log_2 n \leq 2n^2 \log_2 n$$
$$1 \leq 2n^2 \log_2 n - (n^2/4)\log_2 n$$
$$4 \leq 8n^2 \log_2 n - n^2 \log_2 n$$
$$4 \leq 7n^2 \log_2 n$$
$$\frac{4}{7} \leq n^2 \log_2 n$$

$$f(n) = cg(n)$$
$$n \geq n_0$$
$$n \geq 2 \qquad n_0 = \frac{4}{7}$$

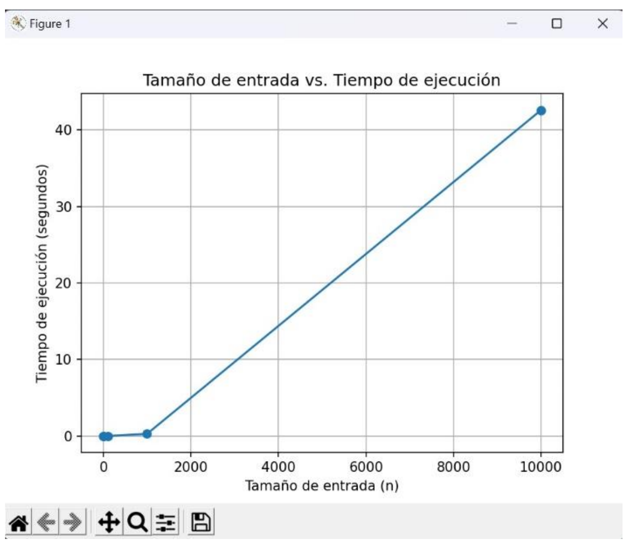
Complejidad

$$f(n) = O(n^2 \log n)$$

Programa:



Grafica:



Tiempo de ejecuciones:

El tiempo de 1 fue: 0.0000000000 segundos
El tiempo de 10 fue: 0.0000000000 segundos
El tiempo de 100 fue: 0.0030069351 segundos
El tiempo de 1000 fue: 0.2635929585 segundos
El tiempo de 10000 fue: 42.6106293201 segundos

Ejercicio 2

```
void function (int n) {
        if (n <= 1) return;                       -> O (n)
        int i, j;                                 -> O (1)
        for (i = 1; i <= n; i++) {                -> O (n)
                for (j = 1; j <= n; j++) {        -> O (1)
                        printf ("Sequence\n");    -> O (1)
                        break;                    -> O (1)
                }
        }
}
```

$f(n) = (n/3)((n-1)/4) + 1 = ((x^2 - x)/12) + 1$

$g(n) = x^2 - x$

¿ $f(n) = O(g(n))$?

$$
\begin{aligned}
f(n) &\le cg(n) \\
((x^2-x)/12)+1 &\le c(x^2-x) \qquad \rightarrow c = 2 \\
((x^2-x)/12)+1 &\le 2(x^2-x) \\
1 &\le 2(x^2-x) - ((x^2-x)/12) \\
12 &\le 24(x^2-x) - (x^2-x) \\
12 &\le 23(x^2-x) \\
\frac{12}{23} &\le (x^2-x)
\end{aligned}
$$

$f(n) = cg(n)$

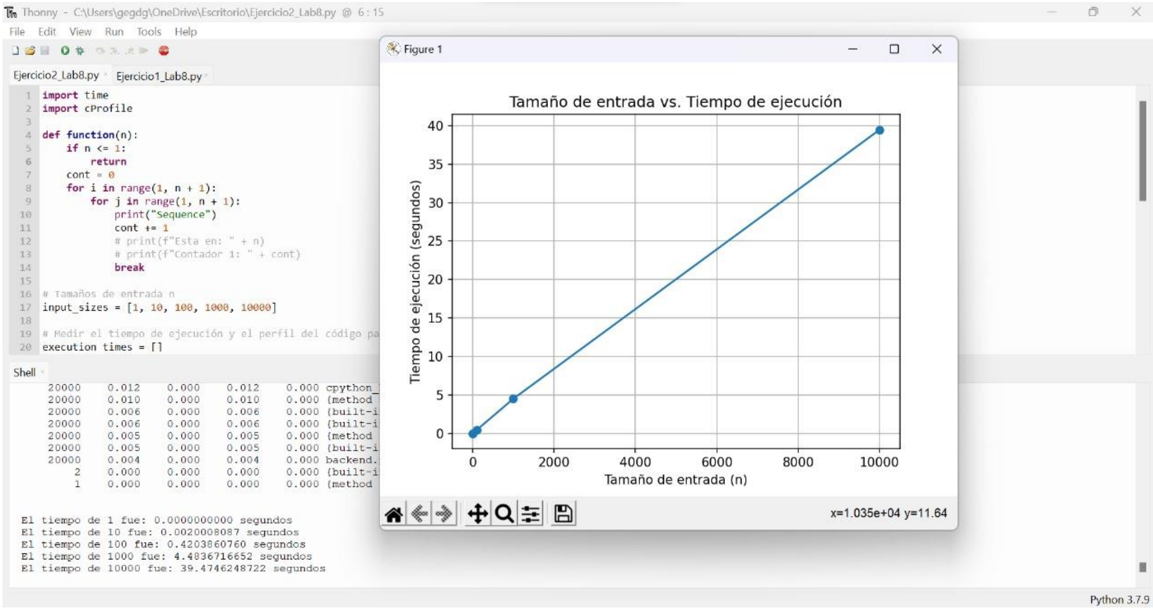
$n \ge n_0$

$c = 2$ $\quad n_0 = \frac{12}{23}$

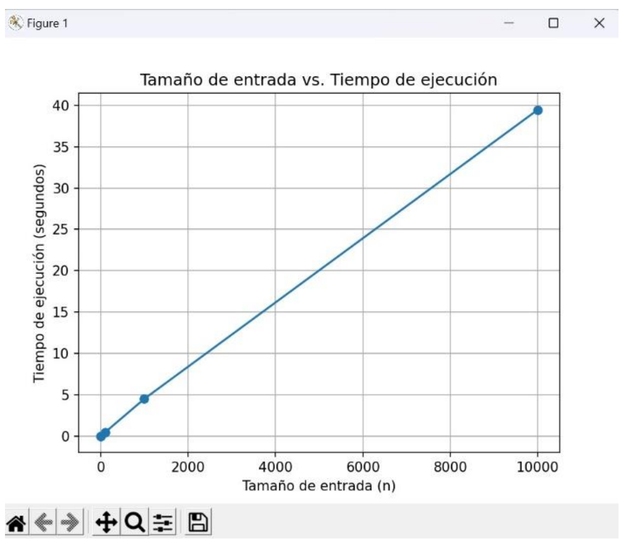
Complejidad

$f(n) = O(n^2)$

Programa:



Grafica:



Tiempo de ejecuciones:

El tiempo de 1 fue: 0.0000000000 segundos
El tiempo de 10 fue: 0.0020008087 segundos
El tiempo de 100 fue: 0.4203860760 segundos
El tiempo de 1000 fue: 4.4836716652 segundos
El tiempo de 10000 fue: 39.4746248722 segundos

Ejercicio 3

```
void function (int n) {
        int i, j;                                   -> O (1)
        for (i=1; i<=n/3; i++) {                    -> O (n)
                for (j=1; j<=n; j+=4) {             -> O (n)
                        printf("Sequence\n");       -> O (1)
                }
        }
}
```

$F(n) = (n/3)((n-1)/4) + 1 = ((x^2 - x)/12) + 1$

$g(n) = x^2 - x$

¿ $f(n) = O(g(n))$?

$f(n) \leq cg(n)$

$((x^2 - x)/12) + 1 \leq c(x^2 - x) \quad \Rightarrow c = 2$

$((x^2 - x)/12) + 1 \leq 2(x^2 - x)$

$1 \leq 2(x^2 - x) - ((x^2 - x)/12)$

$12 \leq 24(x^2 - x) - (x^2 - x)$

$12 \leq 23(x^2 - x)$

$\frac{12}{23} \leq (x^2 - x)$

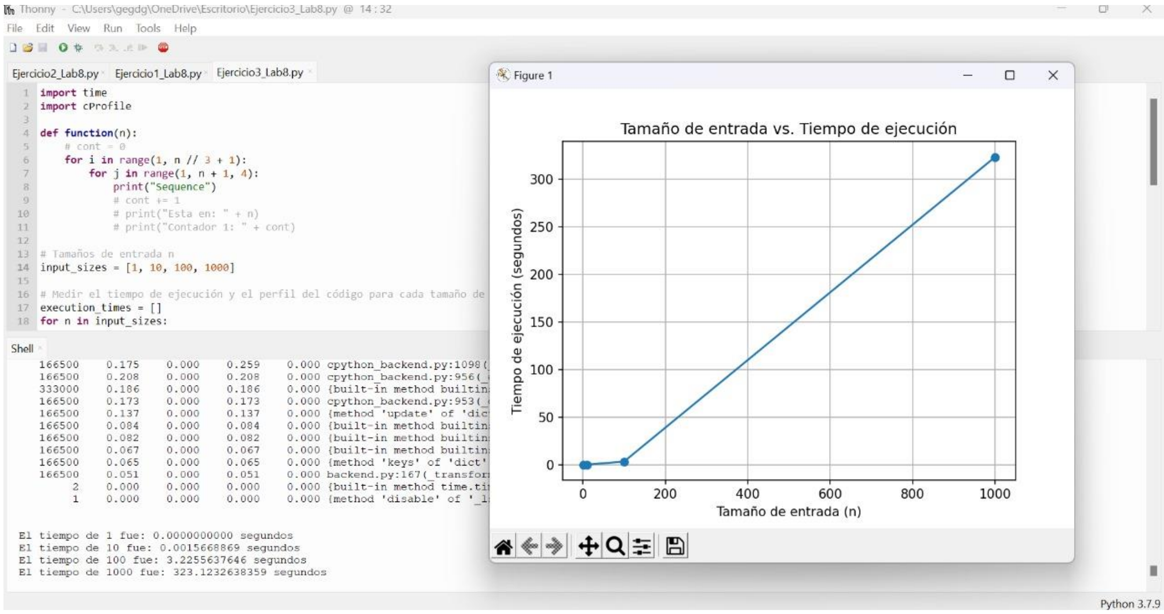
$f(n) = cg(n)$

$n = 2 \quad n_0 = \frac{12}{23}$

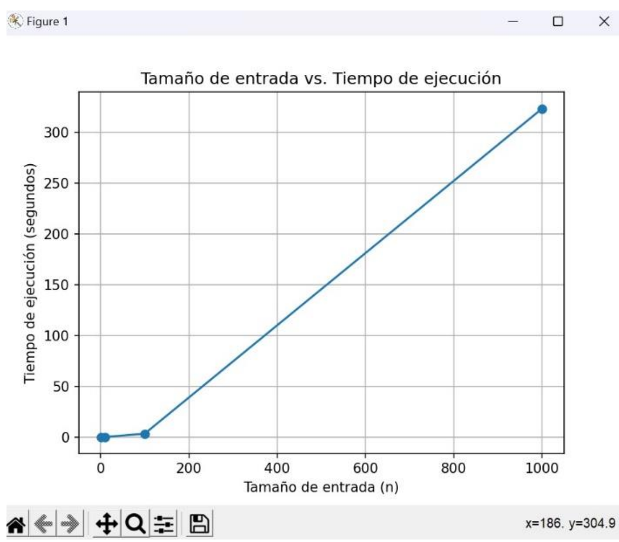
Complejdad

$f(n) = O(n^2)$

Programa:



Grafica:



Tiempo de ejecuciones:

El tiempo de 1 fue: 0.0000000000 segundos
El tiempo de 10 fue: 0.0015668869 segundos
El tiempo de 100 fue: 3.2255637646 segundos
El tiempo de 1000 fue: 323.1232638359 segundos

Ejercicio 4

El algoritmo de búsqueda lineal, también se conoce como búsqueda secuencial y es un método para encontrar un elemento especifico en una lista o en un array. El rendimiento del algoritmo depende según de lo que se esté buscando.

1. Mejor Caso: Esta trata de que el elemento que estamos buscando se encuentre al principio de la lista, entonces solo necesita una comparación para encontrar el elemento.
2. Caso promedio: En el caso promedio es que no se puede asumir la ubicación del elemento en la lista, en donde se encuentra en medio de ella.
3. Peor caso: Es que no se encuentre en la lista, ya que entonces se habrá hecho todo el proceso por gusto, ya que se recorrió toda la lista sin éxito.

Referencia:

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to Algorithms* (3rd ed.). The MIT Press. Página 13.

Ejercicio 5

a) Falso, la notación de θ nos muestra una relación de igualdad asintótica. Si decimos que $f(n)$ = θ ($g(n)$), significa que $f(n)$ y $g(n)$ crecen con la misma taza, sin embargo no por ello significa que $g(n)$ = θ ($h(n)$), por lo que $h(n)$ = θ ($f(n)$), eso significa que la relación de igualdad asintótica no es transitiva, por lo que $h(n)$ = θ ($f(n)$).
b) Verdadero, si $f(n)$ = $O(g(n))$ y $g(n)$ = $O(h(n))$, entonces $f(n)$ tiene una cota superior en términos de $g(n)$ y que $g(n)$ tiene una cota superior a $h(n)$, por lo que si ambas funciones tienen cotas superiores podemos decir que $f(n)$ también tiene una cota superior en términos de $h(n)$, lo que se expresa: $h(n)$ = $\Omega(f(n))$.
c) Verdadero. En el programa podemos observar que el tiempo de ejecución depende del tiempo de ejecución del tamaño de la entrada n, ya que tiene bucles anidados, siendo el que se encuentra en el ciclo interior donde i y j varían según el valor de n, ya que los bucles no se detendrán hasta que n realice n((n-1) /2) iteraciones.
EL tiempo de ejecución es n^2, por lo que $f(n)$ esta acotado por θ (n^2). Dado que el tiempo de ejecución del programa esta acotado por una función cuadrática se puede decir que $f(n)$ = θ (n^2).