

# Notación asintótica

## Análisis de algoritmos

Leonardo Flórez-Valencia

Pontificia Universidad Javeriana  
Departamento de Ingeniería de Sistemas  
Bogotá

Sesión 2

# Ordenamiento burbuja (1)

---

```
1: procedure BUBBLESORT( $S$ )
2:   for  $i \leftarrow 1$  to  $|S|$  do
3:     for  $j \leftarrow 1$  to  $|S| - 1$  do
4:       if  $S[j + 1] < S[j]$  then
5:          $aux \leftarrow S[j]$ 
6:          $S[j] \leftarrow S[j + 1]$ 
7:          $S[j + 1] \leftarrow aux$ 
8:       end if
9:     end for
10:  end for
11: end procedure
```

---

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

6 5 3 1 8 7 2 4

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)


6 5 3 1 8 7 2 4

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

5 6 3 1 8 7 2 4

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

5   6   3   1   8   7   2   4


A diagram illustrating a step in the bubble sort algorithm. The numbers 5, 6, 3, 1, 8, 7, 2, and 4 are arranged horizontally. The numbers 6 and 3 are each enclosed in a red square box, and these two boxes are connected by a red horizontal line, indicating they are the current pair being compared or swapped.

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

5 3 6 1 8 7 2 4

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)


5   3   6   1   8   7   2   4

The diagram shows a sequence of numbers: 5, 3, 6, 1, 8, 7, 2, 4. The numbers 6 and 1 are each enclosed in a red square box, and these two boxes are connected by a red horizontal line, indicating they are the current pair being compared in the bubble sort algorithm.



## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

5   3   1   6   8   7   2   4

A diagram illustrating the bubble sort algorithm. The numbers 5, 3, 1, 6, 8, 7, 2, and 4 are arranged horizontally. The numbers 1 and 6 are each enclosed in a red square box, and these two boxes are connected by a red vertical line, indicating they are the current pair being compared or swapped.

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

5   3   1   6   8   7   2   4

The diagram shows a sequence of numbers: 5, 3, 1, 6, 8, 7, 2, 4. The numbers 6 and 8 are each enclosed in a red square box, and these two boxes are connected by a red vertical line, indicating they are the current pair being compared or swapped in the bubble sort algorithm.

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

5   3   1   6   8   7   2   4

A diagram illustrating a comparison step in bubble sort. The numbers 5, 3, 1, 6, 8, 7, 2, and 4 are arranged horizontally. The numbers 8 and 7 are each enclosed in a red square box, and these two boxes are connected by a vertical red line, indicating they are the current pair being compared.

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

5   3   1   6   7   8   2   4

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

5   3   1   6   7   **8**   **2**   4

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

5   3   1   6   7   2   8   4

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

5   3   1   6   7   2   **8**   **4**

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

5   3   1   6   7   2   **4**   **8**



## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

5   3   1   6   7   2   4   **8**

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

5 3 1 6 7 2 4 8

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

3 5 1 6 7 2 4 8

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

3   5   1   6   7   2   4   8

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

3   1 5   6   7   2   4   8

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

3   1   5   6   7   2   4   8

The diagram shows a sequence of numbers: 3, 1, 5, 6, 7, 2, 4, 8. The numbers 5 and 6 are enclosed in a red rectangular box, indicating they are the current pair being compared. The number 8 is enclosed in a black rectangular box, indicating it is in its final sorted position.

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

3   1   5   6 7   2   4   8

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

3   1   5   6   7 2   4   8



## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

3   1   5   6   2 7   4   8

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

3   1   5   6   2   **7**   **4**   **8**

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

3   1   5   6   2   4 7 8

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

3   1   5   6   2   4   

7	8
---	---

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

3 1 5 6 2 4 7 8

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

1 3 5 6 2 4 7 8

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

1   3   5   6   2   4   7   8

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

1   3   5   6   2   4   7   8



## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

1   3   5   6   2   4   7   8

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

1   3   5   2 6   4   7 8

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

1   3   5   2   6 4 7 8

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

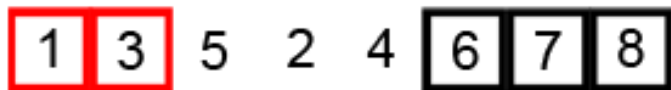
1   3   5   2   4   6   7   8

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

1   3   5   2   4   

6	7	8
---	---	---

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)



## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

1 3 5 2 4 6 7 8

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

1   3   5   2   4   6   7   8



## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

1   3   2 5   4   6 7 8

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

1   3   2   **5**   **4**   6   7   8

A horizontal sequence of numbers: 1, 3, 2, 5, 4, 6, 7, 8. The numbers 5 and 4 are each enclosed in a red square border, and these two red squares are joined together by a red vertical line. The numbers 6, 7, and 8 are each enclosed in a black square border, and these three black squares are joined together by a black vertical line. The numbers 1, 3, and 2 are not enclosed in any border.

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

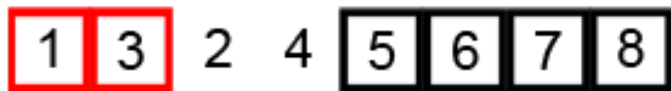
1   3   2   4   5   6   7   8

## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

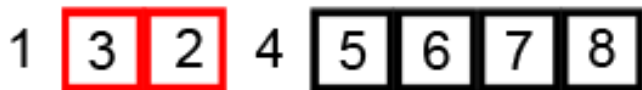
1   3   2   4   

5	6	7	8
---	---	---	---

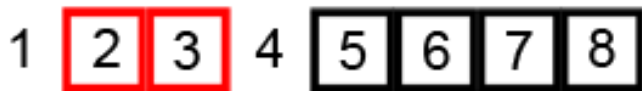
## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)



## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)



## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)



## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)



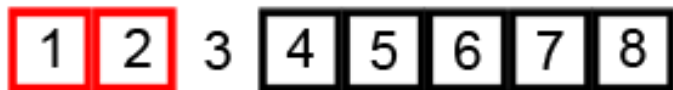


## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)

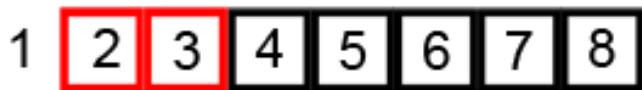
1   2   3   

4	5	6	7	8
---	---	---	---	---

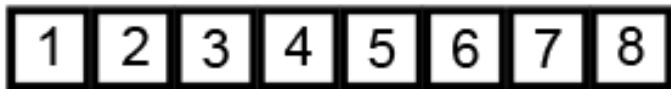
## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)



## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)



## Ordenamiento burbuja: seguimiento (tomado de Wikipedia)



## Ordenamiento burbuja (2)

---

```
1: procedure BUBBLESORT( $S$ )
2:   for  $i \leftarrow 1$  to  $|S|$  do
3:     for  $j \leftarrow 1$  to  $|S| - i$  do
4:       if  $S[j + 1] < S[j]$  then
5:          $aux \leftarrow S[j]$ 
6:          $S[j] \leftarrow S[j + 1]$ 
7:          $S[j + 1] \leftarrow aux$ 
8:       end if
9:     end for
10:  end for
11: end procedure
```

---

# Notación asintótica: motivación

- Un buen diseño, debe llevar a escoger entre un conjunto de soluciones.

# Notación asintótica: motivación

- Un buen diseño, debe llevar a escoger entre un conjunto de soluciones.
- El problema  $P$  tiene un conjunto de  $x$  soluciones algorítmicas numeradas  $A_1, A_2, \dots, A_x$ . ¿Cuál es el mejor?

# Notación asintótica: motivación

- Un buen diseño, debe llevar a escoger entre un conjunto de soluciones.
- El problema  $P$  tiene un conjunto de  $x$  soluciones algorítmicas numeradas  $A_1, A_2, \dots, A_x$ . ¿Cuál es el mejor?
  - ¿La más pequeña?
  - ¿La más fácil de implementar?
  - ¿La más rápida?
  - ¿La más barata?
  - ¿La de microsoft?
  - ¿La de linux?
  - ¿La de mac?
  - ...



# Notación asintótica: motivación

- Un buen diseño, debe llevar a escoger entre un conjunto de soluciones.
- El problema  $P$  tiene un conjunto de  $x$  soluciones algorítmicas numeradas  $A_1, A_2, \dots, A_x$ . ¿Cuál es el mejor?
  - ¿La más pequeña?
  - ¿La más fácil de implementar?
  - ¿La más rápida?
  - ¿La más barata?
  - ¿La de microsoft?
  - ¿La de linux?
  - ¿La de mac?
  - ...

Respuesta:

La de complejidad **adecuada**

# ¿Complejidad?

- ¿Quién ejecuta un algoritmo?
  - ¿Cómo se “pega” el “*modelo*” con la “*tecnología*”?

# ¿Complejidad?

- ¿Quién ejecuta un algoritmo?
  - ¿Cómo se “pega” el “*modelo*” con la “*tecnología*”?
- Espacial  $S(n)$ 
  - El espacio no es muy importante hoy en día (históricamente lo era, y lo va a volver a ser).
  - RAM, cache, virtualización, paginación.
- Temporal  $T(n)$ 
  - Depende del tamaño de los datos de entrada ( $n$ ).
  - ¡El tiempo real depende de la máquina!

# Escritura básica de un algoritmo

- ❶ Describir (analizar y diseñar) formalmente el “problema”.
- ❷ Describir formalmente el “algoritmo” que lo resuelve en términos de una *caja negra*:
  - ❶ Diseño del algoritmo (contrato)
    - ❶ Entradas.
    - ❷ Salidas.
  - ❷ Idea general de solución.
  - ❸ **Escribir el algoritmo.**
- ❸ Analizar los órdenes de complejidad del algoritmo.
- ❹ Describir la invariante del algoritmo.
  - ❶ Mostrar que la invariante se mantiene.

# Problema del ordenamiento

- **Problema:** Ordenar una “lista” / “arreglo” / “vector” / “conjunto” / “montón” de números.
- **Formalización:** Dada una secuencia  $\mathcal{S}$  de elementos  $a_i \in \mathbb{T}$ , donde se define la relación de orden parcial  $\leq$ , producir una nueva secuencia  $\mathcal{S}'$  donde los elementos contiguos cumplan la relación de orden parcial  $\leq$ .
- **Entradas:** Una secuencia  $\mathcal{S}$  de  $n$  números:  $\mathcal{S} = \langle a_1, a_2, \dots, a_n \rangle$  donde  $a_i \in \mathbb{T}$  y en  $\mathbb{T}$  está definida la relación de orden parcial  $\leq$ .
- **Salidas:** Una permutación  
$$\mathcal{S}' = \langle a'_1, a'_2, \dots, a'_n \rangle \mid a'_1 \leq a'_2 \leq \dots \leq a'_n \wedge a'_i \in \mathcal{S} \forall i$$

¿Cuántos algoritmos conoce usted que solucionan este problema?

Burbuja, Inserción, Mezclas, Montículos, *Quicksort*, ...

# Ordenamiento por inserción

---

```
1: procedure INSERTIONSORT( $\mathcal{S}$ )
2:   for  $j \leftarrow 2$  to  $|\mathcal{S}|$  do
3:      $k \leftarrow \mathcal{S}[j]$ 
4:      $i \leftarrow j - 1$ 
5:     while  $0 < i \wedge k < \mathcal{S}[i]$  do
6:        $\mathcal{S}[i + 1] \leftarrow \mathcal{S}[i]$ 
7:        $i \leftarrow i - 1$ 
8:     end while
9:      $\mathcal{S}[i + 1] \leftarrow k$ 
10:  end for
11: end procedure
```

---

- ¿Qué propiedades de los datos (intermedios y resultados) son (deben) siempre ciertas?
  - Inicialización
  - Mantenimiento/Iteración/Actualización/Procesamiento
  - Terminación
- Prueba de escritorio para  $\mathcal{S} = \langle 5, 2, 7, 3, 1, 9, 8 \rangle$

# Invariantes del algoritmo

- ¿Qué propiedades de los datos (intermedios y resultados) son (deben) siempre ciertas?
  - Inicialización
  - Mantenimiento/Iteración/Actualización/Procesamiento
  - Terminación
- Prueba de escritorio para  $S = \langle 5, 2, 7, 3, 1, 9, 8 \rangle$
- Al inicio de cada iteración for, se cumple  $S \langle 1..j-1 \rangle \equiv S' \langle 1..j-1 \rangle$ 
  - “La subsecuencia entre 1 y  $j-1$  estará ordenada”
- ¿Y el while?



# Invariantes del algoritmo

- ¿Qué propiedades de los datos (intermedios y resultados) son (deben) siempre ciertas?
  - Inicialización
  - Mantenimiento/Iteración/Actualización/Procesamiento
  - Terminación
- Prueba de escritorio para  $S = \langle 5, 2, 7, 3, 1, 9, 8 \rangle$
- Al inicio de cada iteración for, se cumple  $S \langle 1..j - 1 \rangle \equiv S' \langle 1..j - 1 \rangle$ 
  - “La subsecuencia entre 1 y  $j - 1$  estará ordenada”
- ¿Y el while?
- Una primera **prueba de correctitud**: demostrar que la invariante se mantiene siempre.

# Invariantes del algoritmo

- ¿Qué propiedades de los datos (intermedios y resultados) son (deben) siempre ciertas?
  - Inicialización
  - Mantenimiento/Iteración/Actualización/Procesamiento
  - Terminación
- Prueba de escritorio para  $S = \langle 5, 2, 7, 3, 1, 9, 8 \rangle$
- Al inicio de cada iteración for, se cumple  $S \langle 1..j - 1 \rangle \equiv S' \langle 1..j - 1 \rangle$ 
  - “La subsecuencia entre 1 y  $j - 1$  estará ordenada”
- ¿Y el while?
- Una primera **prueba de correctitud**: demostrar que la invariante se mantiene siempre.
- Parece inducción, ¿cierto?

# Contemos instrucciones

---

1: <b>procedure</b> INSERTIONSORT( $\mathcal{S}$ )	$\triangleright n =  \mathcal{S} $
2: <b>for</b> $j \leftarrow 2$ <b>to</b> $ \mathcal{S} $ <b>do</b>	$\triangleright 1 + n + (n + 1)$
3: $k \leftarrow \mathcal{S}[j]$	$\triangleright 2$
4: $i \leftarrow j - 1$	$\triangleright 2$
5: <b>while</b> $0 < i \wedge k < \mathcal{S}[i]$ <b>do</b>	$\triangleright 4n$
6: $\mathcal{S}[i + 1] \leftarrow \mathcal{S}[i]$	$\triangleright 4$
7: $i \leftarrow i - 1$	$\triangleright 2$
8: <b>end while</b>	
9: $\mathcal{S}[i + 1] \leftarrow k$	$\triangleright 3$
10: <b>end for</b>	
11: <b>end procedure</b>	

---

# Contemos instrucciones

---

1: <b>procedure</b> INSERTIONSORT( $\mathcal{S}$ )	$\triangleright n =  \mathcal{S} $
2: <b>for</b> $j \leftarrow 2$ <b>to</b> $ \mathcal{S} $ <b>do</b>	$\triangleright 1 + n + (n + 1)$
3: $k \leftarrow \mathcal{S}[j]$	$\triangleright 2$
4: $i \leftarrow j - 1$	$\triangleright 2$
5: <b>while</b> $0 < i \wedge k < \mathcal{S}[i]$ <b>do</b>	$\triangleright 4n$
6: $\mathcal{S}[i + 1] \leftarrow \mathcal{S}[i]$	$\triangleright 4$
7: $i \leftarrow i - 1$	$\triangleright 2$
8: <b>end while</b>	
9: $\mathcal{S}[i + 1] \leftarrow k$	$\triangleright 3$
10: <b>end for</b>	
11: <b>end procedure</b>	

---

$$T(n) = 10n^2 + 9n + 2$$

$$S(n) = ?$$

## ¿Por qué nos interesa el *peor de los casos*?

- Límite superior del algoritmo.
- Ocurre la mayoría de las veces en la mayoría de los algoritmos.
- El caso promedio es “casi” como el peor de los casos.

# ¿Por qué nos interesa el *peor de los casos*?

- Límite superior del algoritmo.
- Ocurre la mayoría de las veces en la mayoría de los algoritmos.
- El caso promedio es “casi” como el peor de los casos.

## Peor de los casos

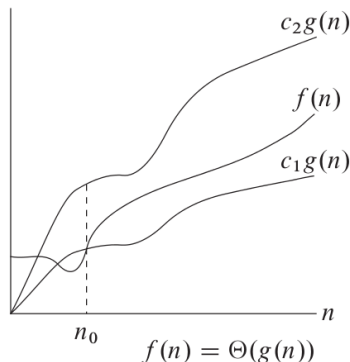
Una función de tiempo  $T(n) = \sum_{i=0}^n a_i x^i$  tiene un orden de crecimiento (complejidad)  $O(x^n)$  “O de  $x$  a la  $n$ ”.

$$\begin{aligned}\Theta(g(n)) = \{f(n) : \\ &\exists c_1, c_2, n_0 > 0 \mid \\ &\forall n \geq n_0, \\ &0 \leq c_1 g(n) \leq \\ &f(n) \leq c_2 g(n)\}\end{aligned}$$

$$\Theta(g(n)) = \{f(n) : \\ \exists c_1, c_2, n_0 > 0 \mid \\ \forall n \geq n_0, \\ 0 \leq c_1 g(n) \leq \\ f(n) \leq c_2 g(n)\}$$

## Cota

$g(n)$  es una cota ajustada asintótica de  $f(n)$ .

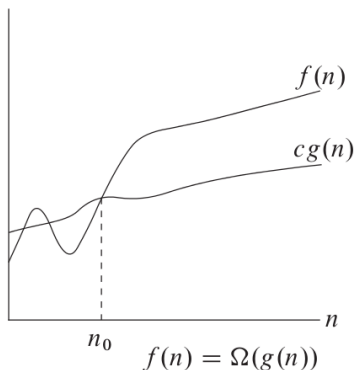


Tomado de [Cormen et al. 2009]



$$\Omega(g(n)) = \{f(n) : \exists c, n_0 > 0 \mid \forall n \geq n_0, 0 \leq cg(n) \leq f(n)\}$$

$$\Omega(g(n)) = \{f(n) : \exists c, n_0 > 0 \mid \forall n \geq n_0, 0 \leq cg(n) \leq f(n)\}$$

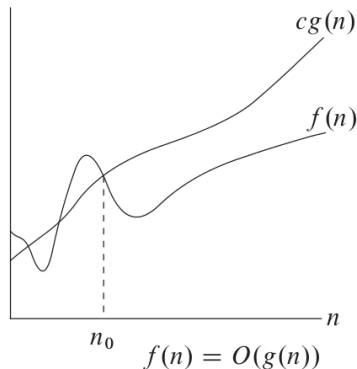


## Cota inferior

$g(n)$  es la cota inferior  
asintótica de  $f(n)$ .

Tomado de [Cormen et al. 2009]

$$O(g(n)) = \{f(n) : \exists c, n_0 > 0 \mid \forall n \geq n_0, 0 \leq f(n) \leq cg(n)\}$$



## Cota superior

$g(n)$  es la cota superior asintótica de  $f(n)$ .

Tomado de [Cormen et al. 2009]

## Tres reglas aritméticas de la complejidad $O(\cdot)$ :

- $T(n)$  es  $O(kf(n)) \rightarrow T(n)$  es  $O(f(n))$
- $T_1(n)$  es  $O(f_1(n)) \wedge T_2(n)$  es  $O(f_2(n)) \rightarrow T_{1;2}(n)$  es  $O(\max(f_1(n), f_2(n)))$
- $T_1(n)$  es  $O(f_1(n)) \wedge T_2(n)$  es  $O(f_2(n)) \rightarrow T_{1(2)}(n)$  es  $O(f_1(n)f_2(n))$

# Algunos tiempos\* para cotas inferiores

1 ins  $2.85 \times 10^{-10}s$ , procesador de 3.5GHz.

$f(n = 256)$	<b>Tiempo (ins)</b>
$\Omega(\log_2(n))$	8 ins
$\Omega(n)$	256 ins
$\Omega(n \log_2(n))$	2048 ins
$\Omega(n^2)$	65536 ins
$\Omega(n^3)$	16777216 ins
$\Omega(2^n)$	$1.15 \times 10^{77}$ ins
$\Omega(n!)$	$8.57 \times 10^{507}$ ins

# Algunos tiempos\* para cotas inferiores

1 ins  $2.85 \times 10^{-10}s$ , procesador de 3.5GHz.

$f(n = 256)$	<b>Tiempo (s)</b>
$\Omega(\log_2(n))$	$2.28ns$
$\Omega(n)$	$72.96ns$
$\Omega(n \log_2(n))$	$583.58ns$
$\Omega(n^2)$	$18.67\mu s$
$\Omega(n^3)$	$4.78ms$
$\Omega(2^n)$	$3.2 \times 10^{67}s$
$\Omega(n!)$	$2.44 \times 10^{498}s$

# Algunos tiempos\* para cotas inferiores

1 ins  $2.85 \times 10^{-10}s$ , procesador de 3.5GHz.

$f(n = 256)$	Tiempo (s)
$\Omega(\log_2(n))$	$2.28ns$
$\Omega(n)$	$72.96ns$
$\Omega(n \log_2(n))$	$583.58ns$
$\Omega(n^2)$	$18.67\mu s$
$\Omega(n^3)$	$4.78ms$
$\Omega(2^n)$	$3.2 \times 10^{67}s$
$\Omega(n!)$	$2.44 \times 10^{498}s$

Edad estimada del universo:  $4 \times 10^{27}s$ .

# Algunos tiempos\* para cotas inferiores

1 ins  $2.85 \times 10^{-10}s$ , procesador de 3.5GHz.

$f(n = 256)$	<b>Tiempo (s)</b>
$\Omega(\log_2(n))$	$2.28ns$
$\Omega(n)$	$72.96ns$
$\Omega(n \log_2(n))$	$583.58ns$
$\Omega(n^2)$	$18.67\mu s$
$\Omega(n^3)$	$4.78ms$
$\Omega(2^n)$	$8 \times 10^{39}U$
$\Omega(n!)$	$6.1 \times 10^{470}U$