ESTRUCTURA DE DATOS

PRACTICA 1: EFICIENCIA

MANUEL FERNANDEZ LA CHICA
EMILIO JOSE OCHANDO PANTIGAS

Practica 1: Eficiencia

Descripción de hardware:

Nombre del modelo:	*MacBook Pro*
Identificador del modelo:	*MacBookPro11,2*
Nombre del procesador:	*Intel Core i7*
Velocidad del procesador:	*2,2 GHz*
Cantidad de procesadores:	*1*
Cantidad total de núcleos:	*4*
Caché de nivel 2 (por núcleo):	*256 KB*
Caché de nivel 3:	*6 MB*
Memoria:	*16 GB*
Versión de la ROM de arranque:	*MBP112.0146.B00*
Versión SMC (sistema):	*2.18f15*
Número de serie (sistema):	*C02N6KS8G3QC*
UUID de hardware:	*3C265C7D-2669-5DAD-94DF-78EF023C701D*

Descripción de software:

g++ —version:

Configured with: --prefix=/Library/Developer/CommandLineTools/usr --with-gxx-include-dir=/Library/Developer/CommandLineTools/SDKs/MacOSX10.14.sdk/usr/include/c++/4.2.1
Apple LLVM version 10.0.0 (clang-1000.10.44.2)
Target: x86_64-apple-darwin18.0.0
Thread model: posix
InstalledDir: /Library/Developer/CommandLineTools/usr/bin

gnuplot —version:

gnuplot 5.2 patchlevel 4

Ejercicio 1: Ordenación de la burbuja

Eficiencia Teorica:

```
1.void ordenar(int *v, int n) {
2.         for (int i=0; i<n-1; i++)
3.           for (int j=0; j<n-i-1; j++)
4.             if (v[j]>v[j+1]) {
5.               int aux = v[j];
6.                 v[j+1] = aux;
          }

   }
```

Línea 2: 4 OE (Asignación i=0, Comparación y Decrecimiento i<n-1, Incremento i++)

Línea 3: 4 OE (Asignación j=0, Comparación y Decrecimiento j<n-i-1, Incremento j++)

Línea 4: 4 OE (Acceso elemento v[j], Acceso elemento e Incremento v[j+1], Comparación v[j]>v[j+1])

Línea 5: 2 OE (Asignación aux, Acceso elemento v[j])

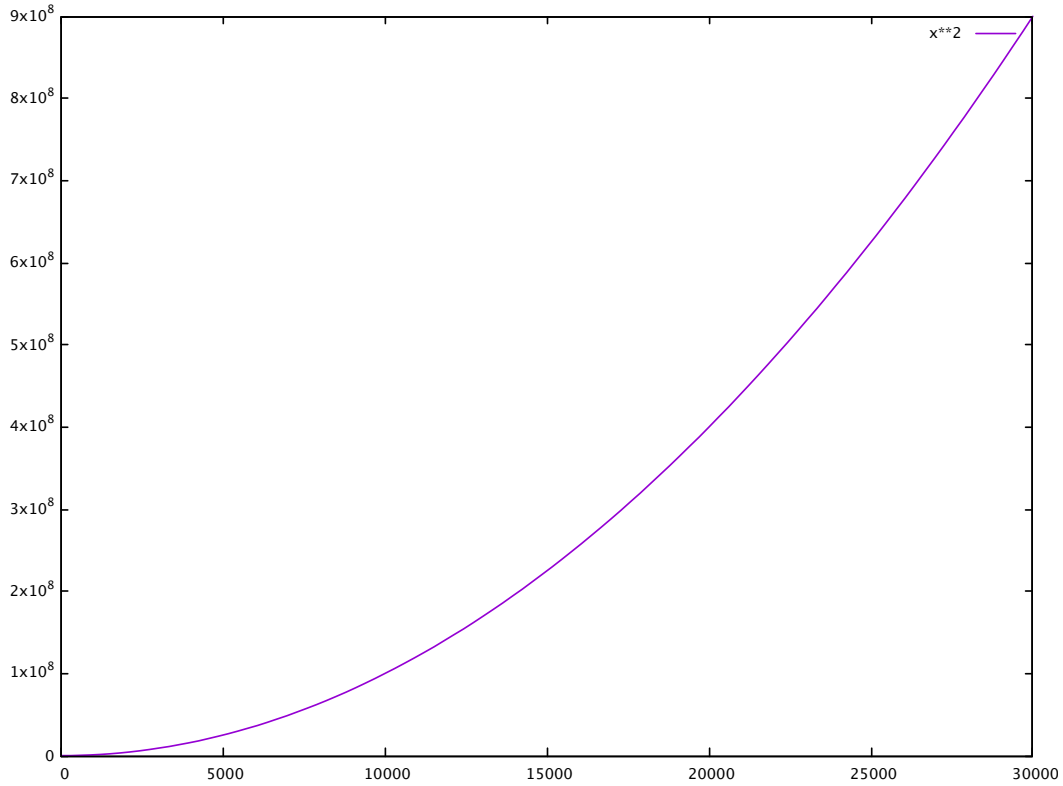
Línea 6: 4 OE (Acceso el evento v[j], Acceso elemento e Incremento v[j+1], Asignación v[j]=v[j+1])

Línea : 6 OE (Acceso elemento e Incremento v[j+1], Asignación v[j+1] = aux)

$$\sum_{i=0}^{n-2}\left(\sum_{j=0}^{n-i-2}(4+4+2+4+3)\right)=\sum_{i=0}^{n-2}(n-i-2)=\sum_{i=0}^{n-2}n-\sum_{i=0}^{n-2}i-\sum_{i=0}^{n-2}2=\in O(n^2)$$

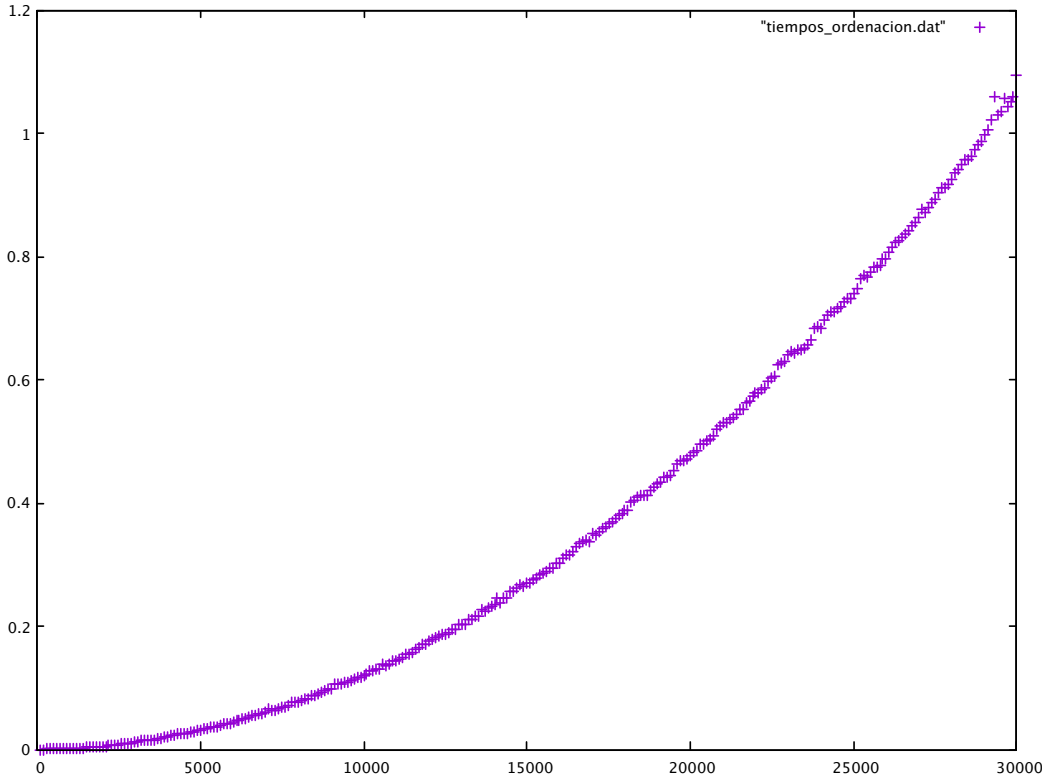
Representación Teórica:

set xrange [0:30000]
gnuplot> plot x**2



Representación empírica:

Esta representación el limite se propone a 60.000 con un incremento de 100



Ejercicio 2: Ajuste de la ordenación de la burbuja

gnuplot
>f(x) = a*x**2+b*x+c
fit f(x) “tiempos_burbuja.dat” via a,b,c

iter	chisq	delta/lim	lambda	a	b	C
0	5.1769434944e-01	0.00e+00	1.09e+00	1.172127e-0	1.002108e-06	-2.371929e-03
1	5.1769434944e-01	-1.72e-10	1.09e-01	1.172127e-09	1.002108e-06	-2.371929e-03
iter	chisq	delta/lim	lambda	a	b	C

After 1 iterations the fit converged.
final sum of squares of residuals : 0.517694
rel. change during last iteration : -1.71564e-15

degrees of freedom (FIT_NDF) : 597
rms of residuals (FIT_STDFIT) = sqrt(WSSR/ndf) : 0.0294476
variance of residuals (reduced chisquare) = WSSR/ndf : 0.00086716

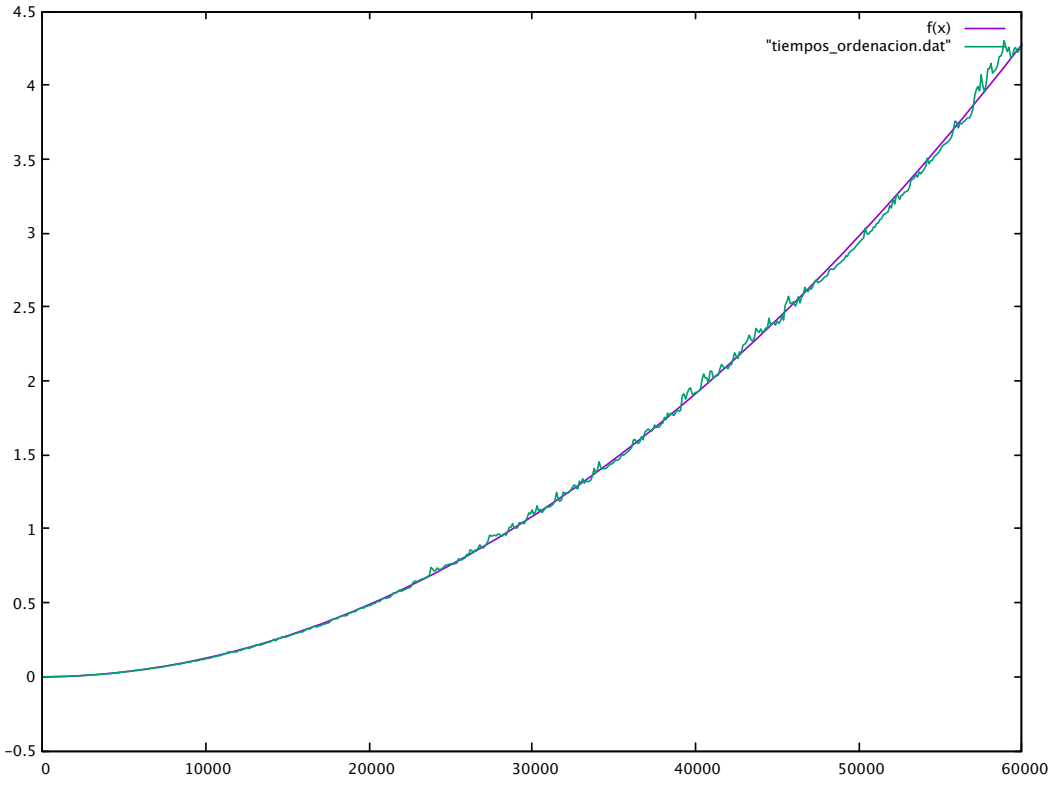
Final set of parameters		Asymptotic Standard Error	
a	= 1.17213e-09	+/- 4.48e-12	(0.3822%)
b	= 1.00211e-06	+/- 2.781e-07	(27.75%)
c	= -0.00237193	+/- 0.003619	(152.6%)

correlation matrix of the fit parameters:

	a	b	c
a	1.000		
b	-0.968	1.000	
c	0.747	-0.867	1.000

Eficiencia Empírica:

plot f(x), “tiempos_ordenacion.dat” w l



Ejercicio 3: Problemas de precisión

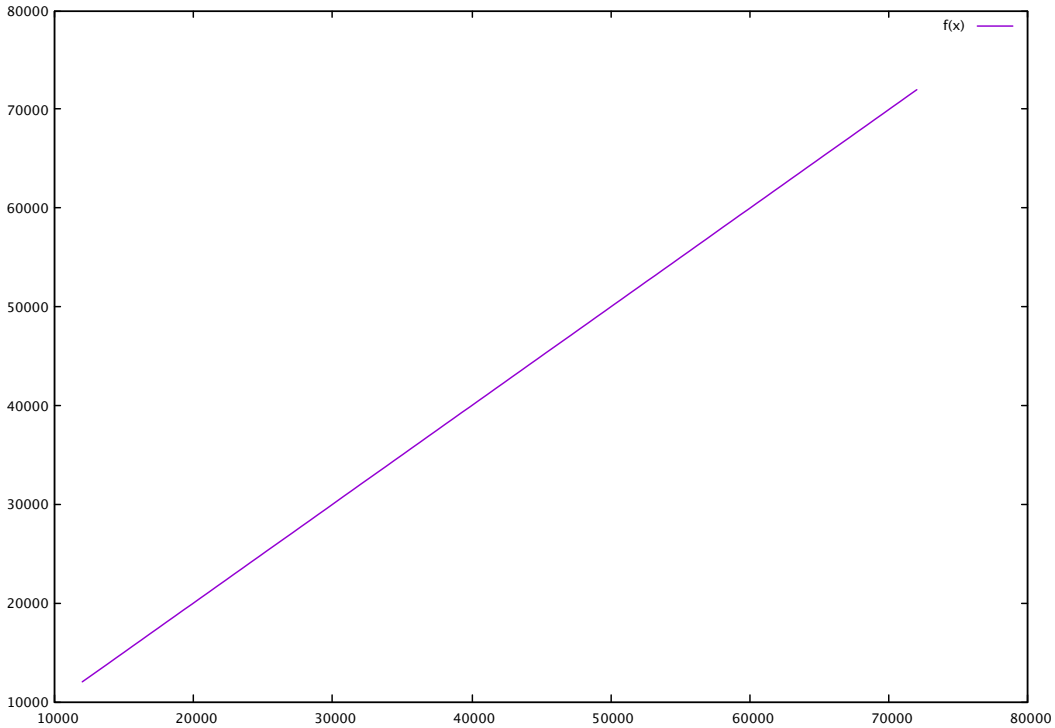
Explicación que realiza el Algoritmo:

El algoritmo es el método de búsqueda binaria.
El programa lee unos parámetros de entrada, genera un vector aleatorio y calcula el tiempo empleado en ordenar dicho vector con el algoritmo.
Finalmente imprime por pantalla los resultados.

Eficiencia Teórica:

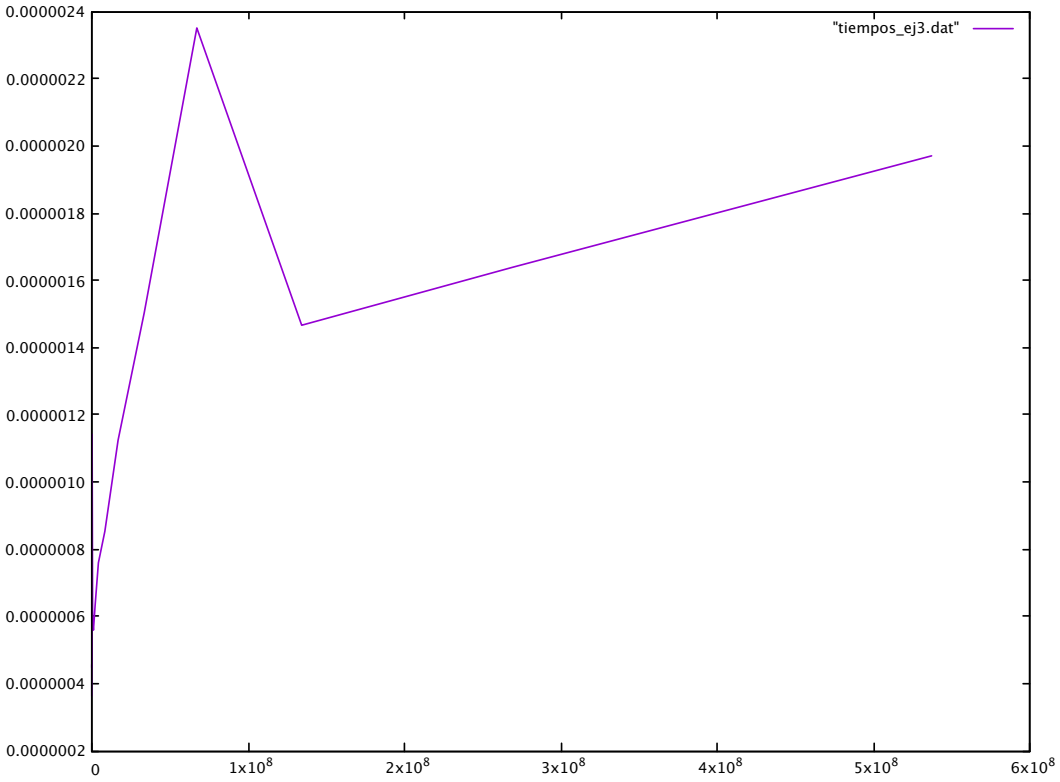
Línea 2: 1 OE (Asignación)
Línea 3: 1 OE (Asignación enc = false)
Línea 4: 3 OE (Uso de elemento &, Comprobación !enc, Comparación inf < sup)
Línea 5: 3 OE (Asignación med = (inf + sup) / 2, Operación inf + sup, Operación (inf + sup) / 2)
Línea 6: 2 OE (Acceso el evento v[med], Comparación v[med] = = x)
Línea 7: 1 OE (Asignación enc = true)
Línea 8: 2 OE (Acceso elemento v[med], Comparación v[med] < x)
Línea 9: 2 OE (Asignación inf = med + 1, Incremento med +1)
Línea 10: 1 OE (Comprobación)
Línea 11: 2 OE (Asignación sup = med + 1, Decrecimiento med -1)
Línea 13: 1 OE (Comprobación)
Línea 14: 1 OE (Devolución return med)
Línea 15: 1 OE (Comprobación)
Línea 16: 1 OE (Devolución return -1)

A la complejidad de esta operación se le denomina O(n).

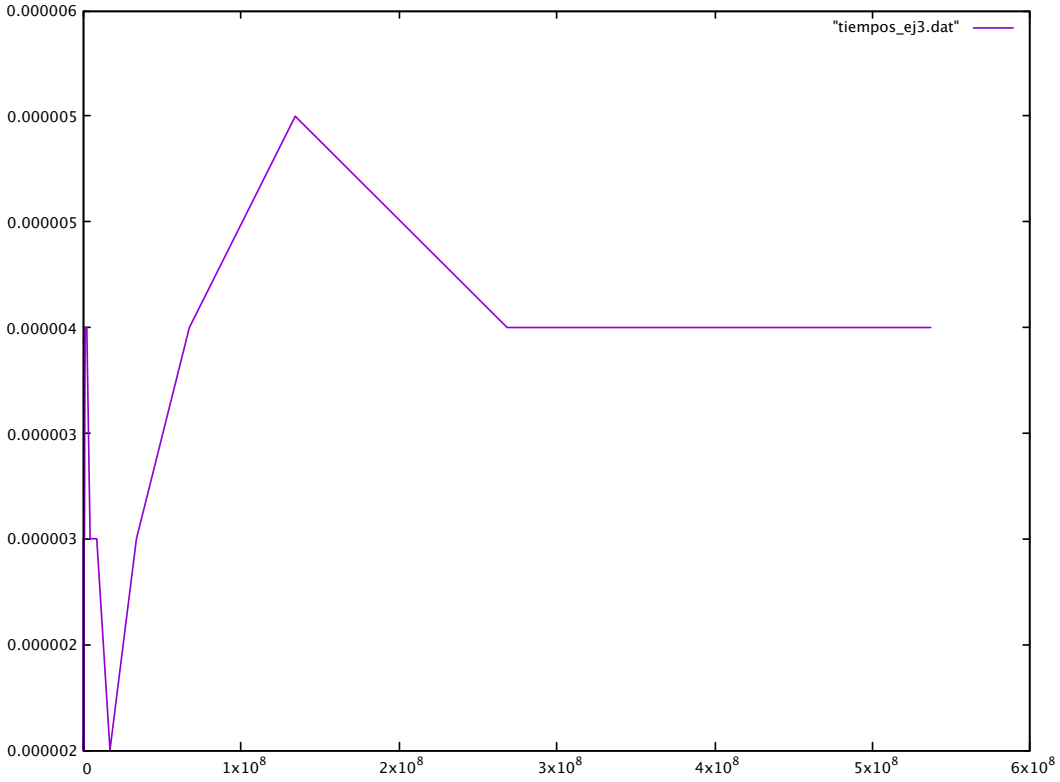


Eficiencia Empirica:

ejercicio_desc.cpp



ejercicio3.cpp con crono



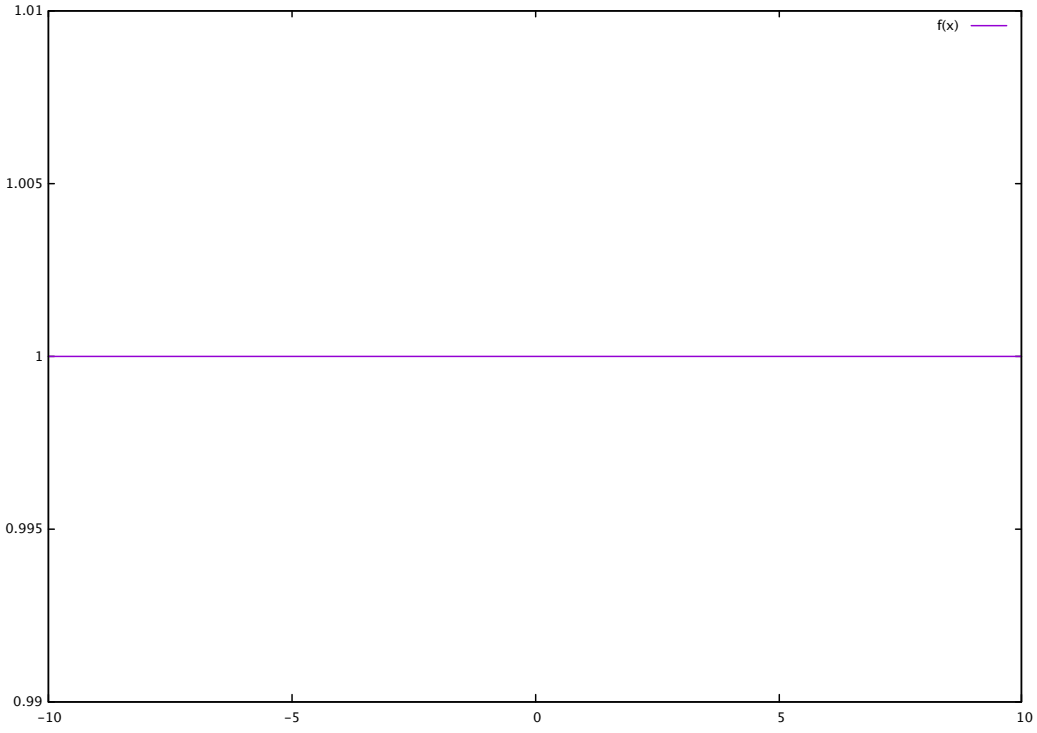
Ejercicio 4: Dependencia de la implementación

Considere esta otra implementación del algoritmo de la burbuja:

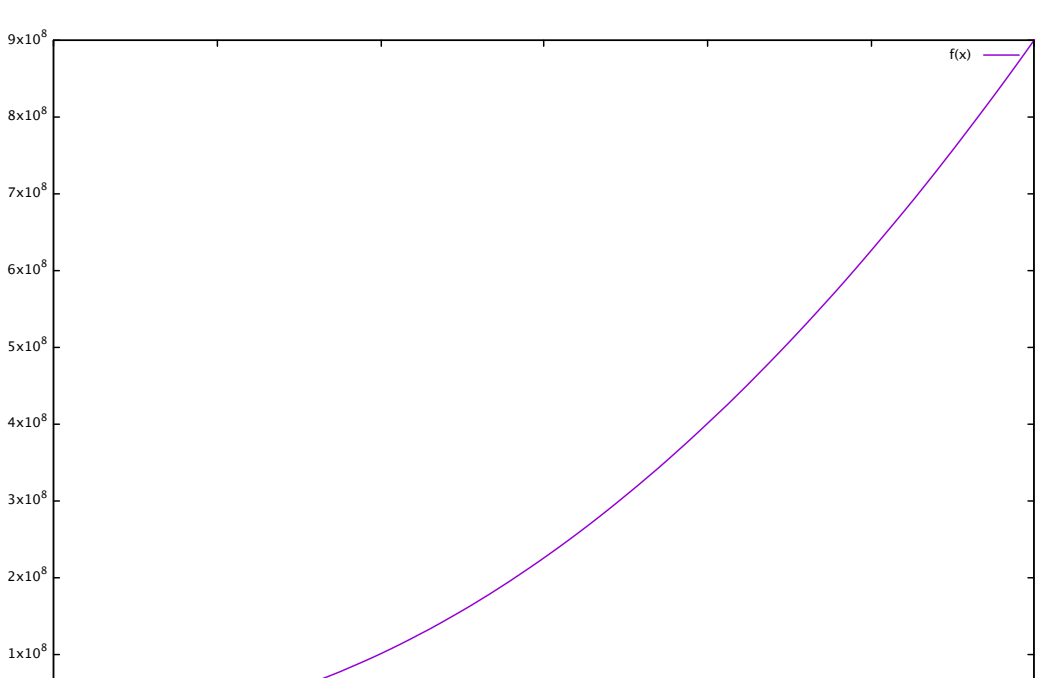
```
        void ordenar(int *v, int n) {
          bool cambio=true;
          for (int i=0; i<n-1 && cambio; i++) {
            cambio=false;
            for (int j=0; j<n-i-1; j++)
              if (v[j]>v[j+1]) {
                cambio=true;
                swap (v[j],v[j+1]); //incluir algorithm
} }

}
```

Eficiencia Teórica:

En el mejor caso posible es $O(n)$, ya que solo realiza la comprobación.

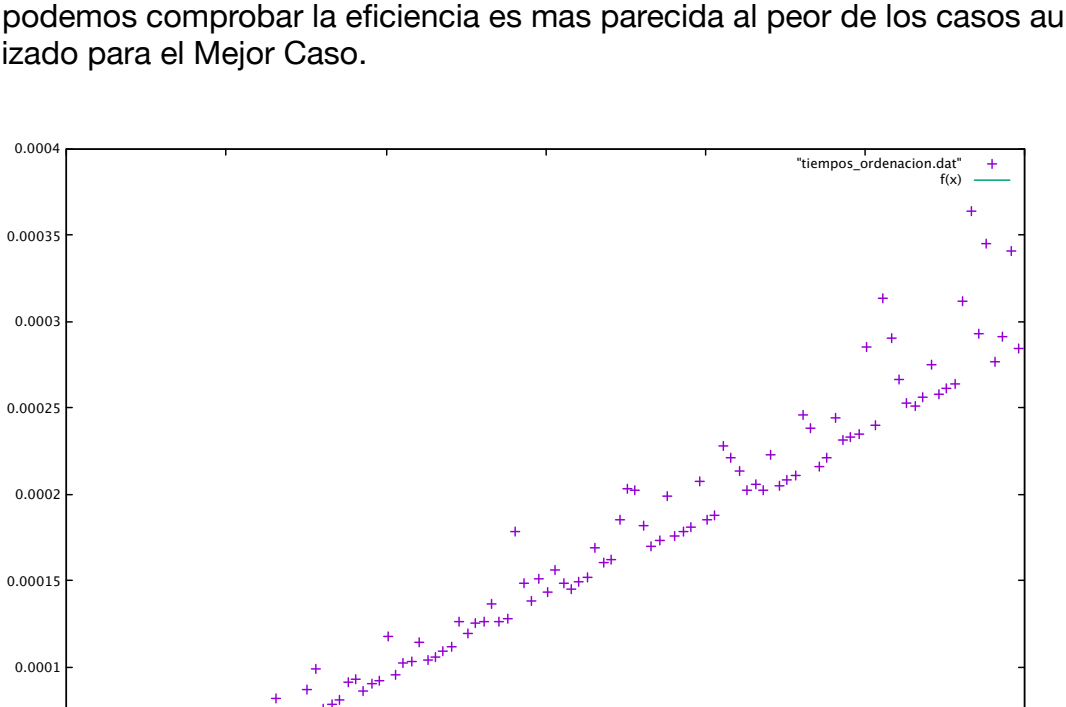


En el peor caso posible, que seria el vector estaría ordenado de mayor a menor(al contrario) por lo tanto realiza los dos bucles en todas las interacciones,
$O(n) * O(n) = O(n^2)$



Eficiencia Empírica:

Como podemos comprobar la eficiencia es mas parecida al peor de los casos aunque se ha realizado para el Mejor Caso.



Ejercicio 5: Mejor y Peor Caso

Eficiencia Teórica:

```
// Generación del vector aleatorio
   int *v=new int[tam];

   // Reserva de memoria
   srand(time(0));

   //INICIALIZACION MEJOR CASO
   for(int i=0; i<tam; i++)
      v[i]=i;
```

En el mejor caso disponemos del vector ordenado, es decir solo realiza la comprobación de los numero, por lo tanto obtenemos una eficiencia teórica de $O(n)$

```
   //INICIALIZACION PEOR CASO
   int aux = tam-1;
    for(int i=0; i<tam;i++)
    {
    v[i]=aux;
    aux--;
    }
```

En el peor de los casos disponemos del vector ordenado, al contrario de como el resultado final, por lo tanto tiene que realizar todos los cambios posibles para todos los tamaños lo que nos proporciona una eficiencia $O(n^2)$

Eficiencia Empírica:

Como podemos observa se cumple las dos condiciones en el gráfico del mejor y peor caso.