

**Práctica 1: Eficiencia**

*Alberto Llamas González*

*2º Grado en Ingeniería Informática*

**Índice:**

0. Características del hardware y del SO.

1. Ejercicio 1: Ordenación de la burbuja.

2. Ejercicio 2: Ajuste en la ordenación de la burbuja.

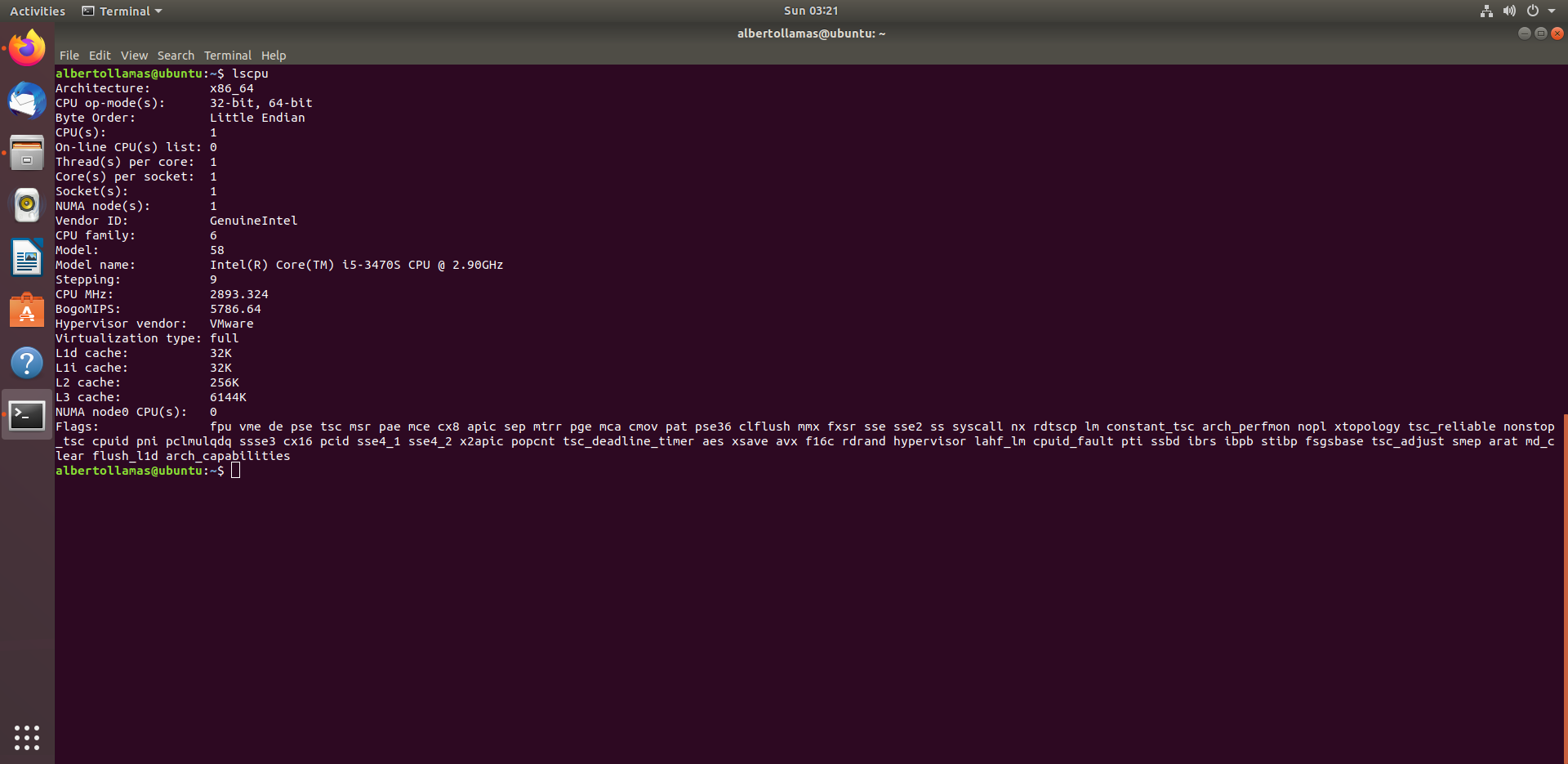
3. Ejercicio 3: Problemas de precisión.

4. Ejercicio 4: Mejor y peor caso.

5. Ejercicio 5: Dependencia de la implementación.

0. Características del hardware y del SO

Para realizar esta práctica, he utilizado una máquina virtual con el Sistema Operativo Ubuntu 18.04.5. Además, el ordenador que he utilizado cuenta con un procesador Intel Core i5-3470S con una velocidad de 2.90 GHz con 1 núcleo de procesador y una arquitectura de 64 bits. La RAM del ordenador (iMac 2013) es 8GB.



1. Ejercicio 1: *Ordenación de la burbuja*.

void ordenar (int \*v, int n) {

for (int i=0; i<n-1; i++)

for (int j=0; j<n-i-1; j++)

if (v[j]>v[j+1]) {

int aux = v[j];

v[j] = v[j+1];

v[j+1] = aux;

}

}

Calcule la eficiencia teórica de este algoritmo. A continuación, replique el experimento que se ha hecho antes (búsqueda lineal) con este nuevo código. Debe:

• Crear un fichero ordenacion.cpp con el programa completo para realizar una ejecución del algoritmo.

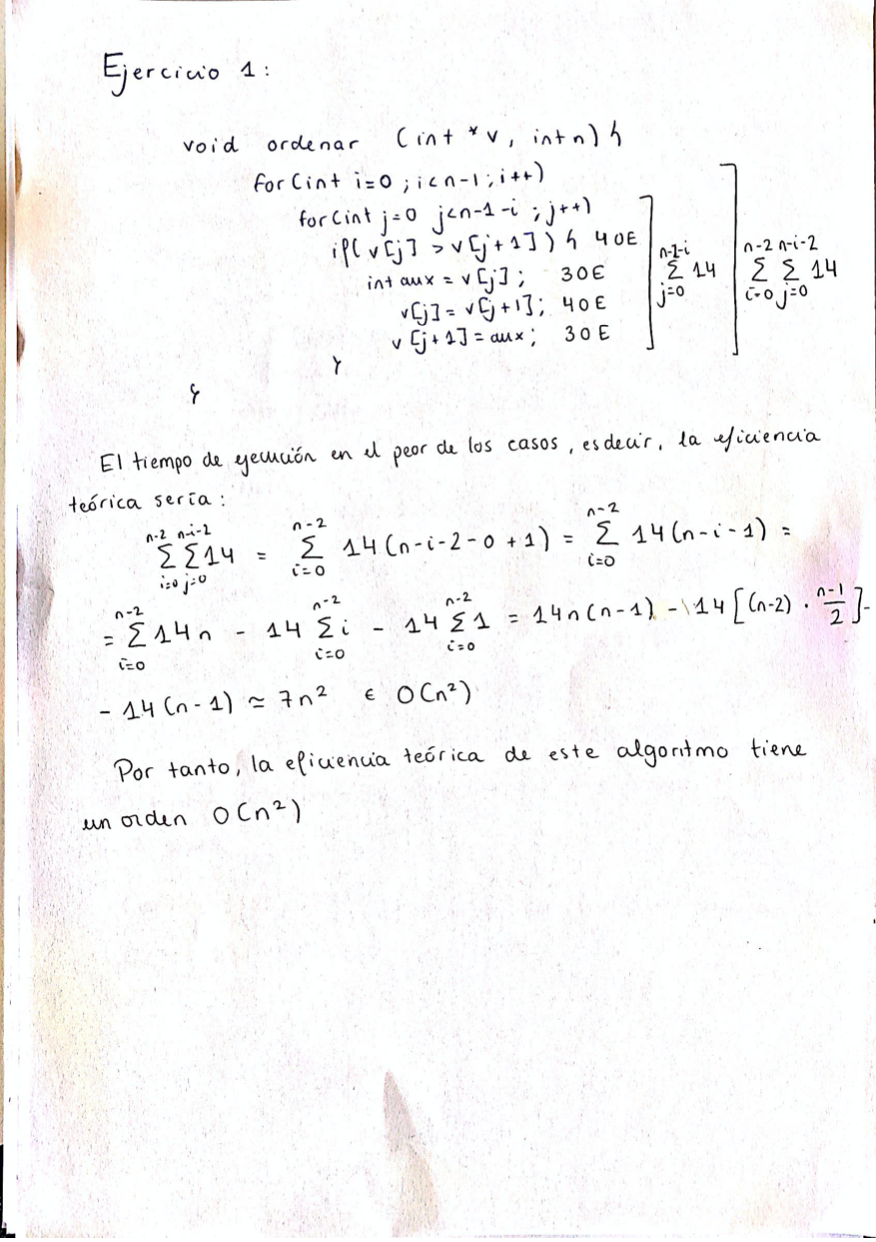
• Crear un script ejecuciones\_ordenacion.csh en C-Shell que permite ejecutar varias veces el programa anterior y generar un fichero con los datos obtenidos.

• Usar gnuplot para dibujar los datos obtenidos en el apartado previo.

Los datos deben contener tiempos de ejecución para tamaños del vector 100, 600, 1100, ..., 30000. Pruebe a dibujar superpuestas la función con la eficiencia teórica y la empírica. ¿Qué sucede?

**Solución al ejercicio 1**

Calculemos la eficiencia teórica del algoritmo de ordenación burbuja contando el número de operaciones elementales de éste (se ha realizado en papel el cálculo de dicha eficiencia):



A continuación, calculemos la eficiencia de forma experimental. Para ello se ha utilizado el siguiente código en lenguaje C++ proporcionado en el material de prácticas:

*ordenacion\_burbuja.cpp*

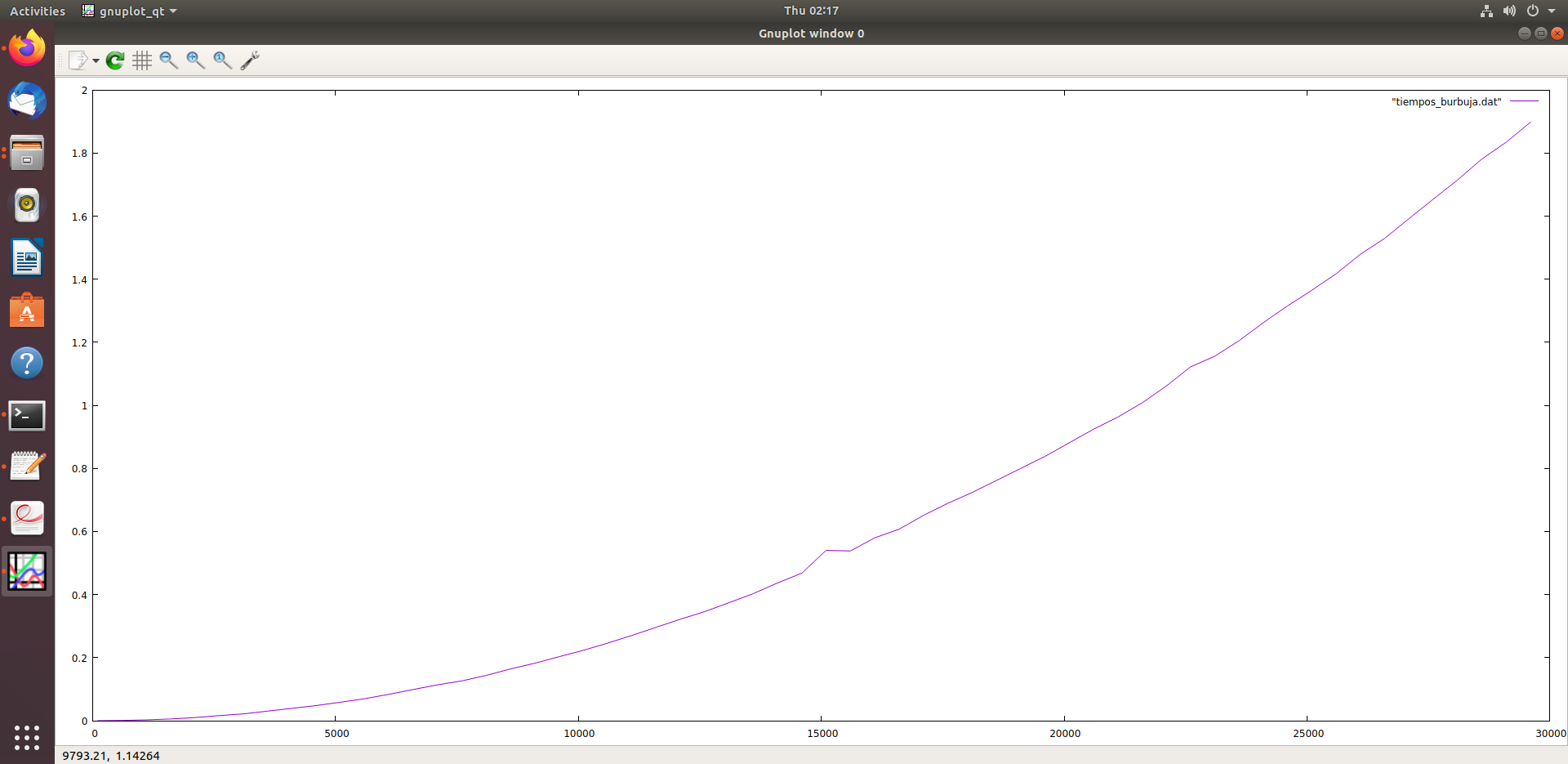


Y el siguiente script:

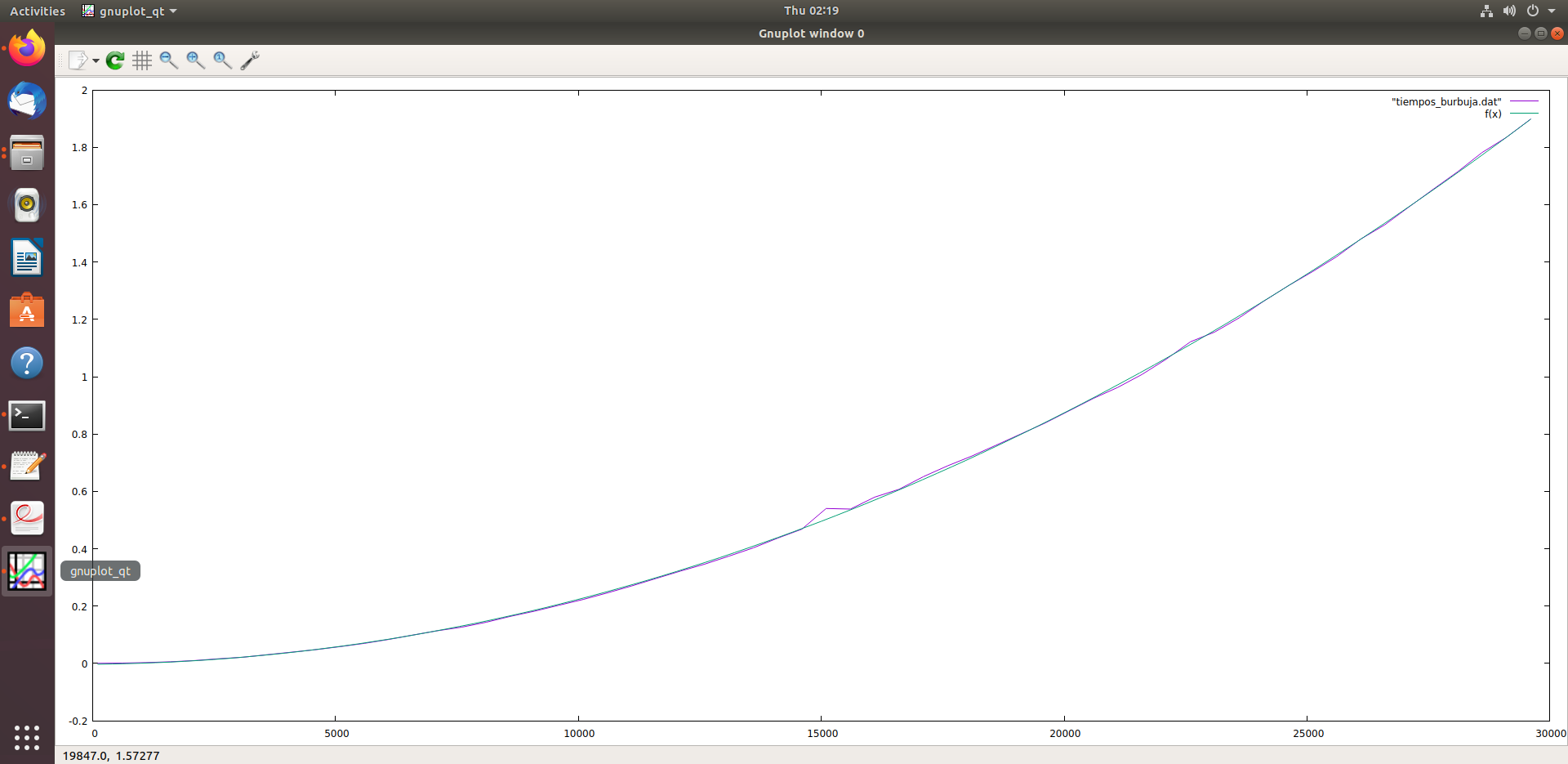
*ejecuciones\_burbuja.csh*



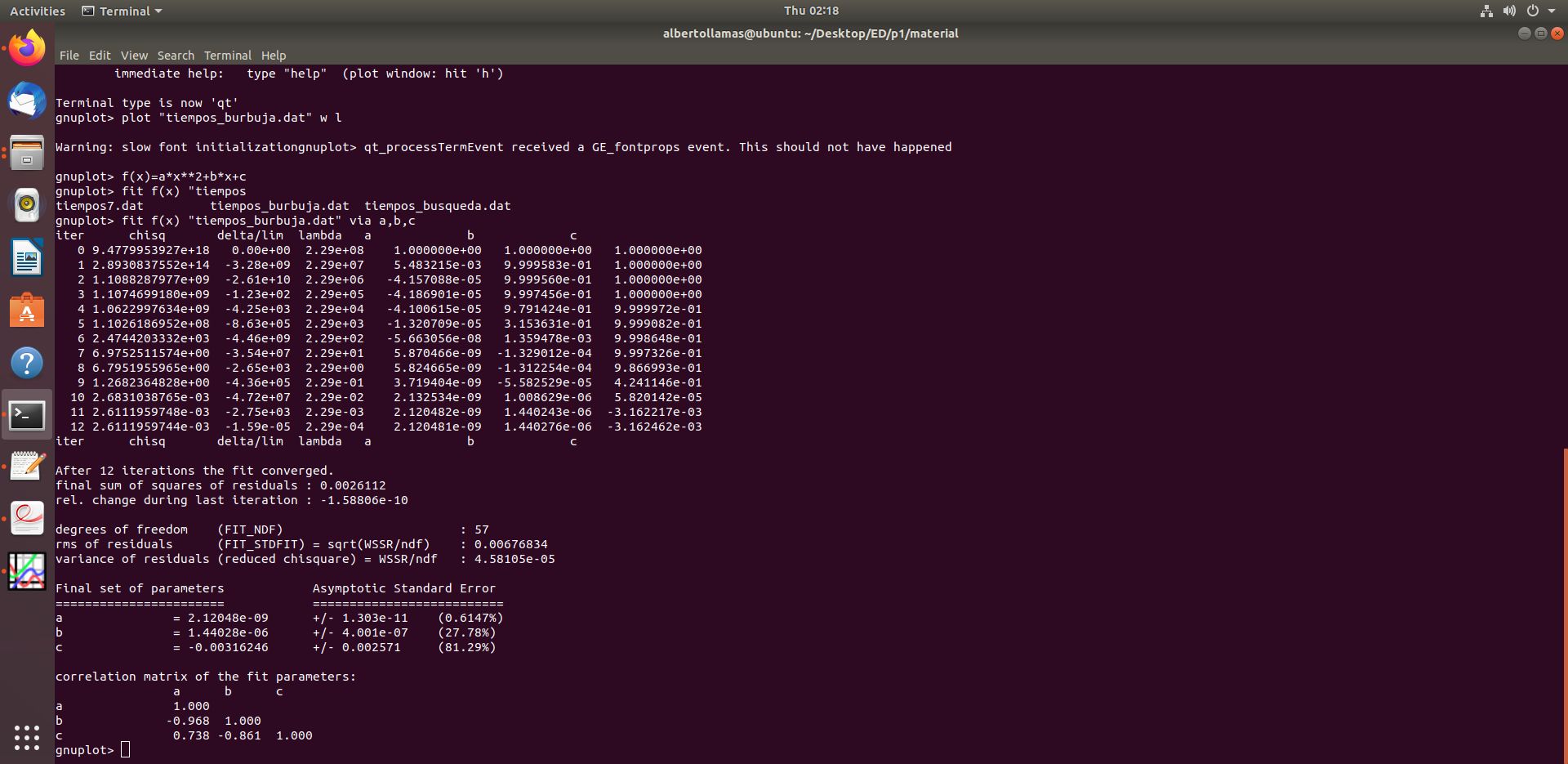
Una vez ejecutado *ejecuciones\_burbuja.csh* vemos que el plot de gnuplot para estos datos es:



El cual, claramente coincide con la gráfica teórica, como podemos ver en la siguiente gráfica donde f(x)=a\*x\*\*2+b\*x+c.



Como podemos ver coinciden prácticamente en todos los puntos debido a que el error es casi 0 como se comprueba en la siguiente imagen:

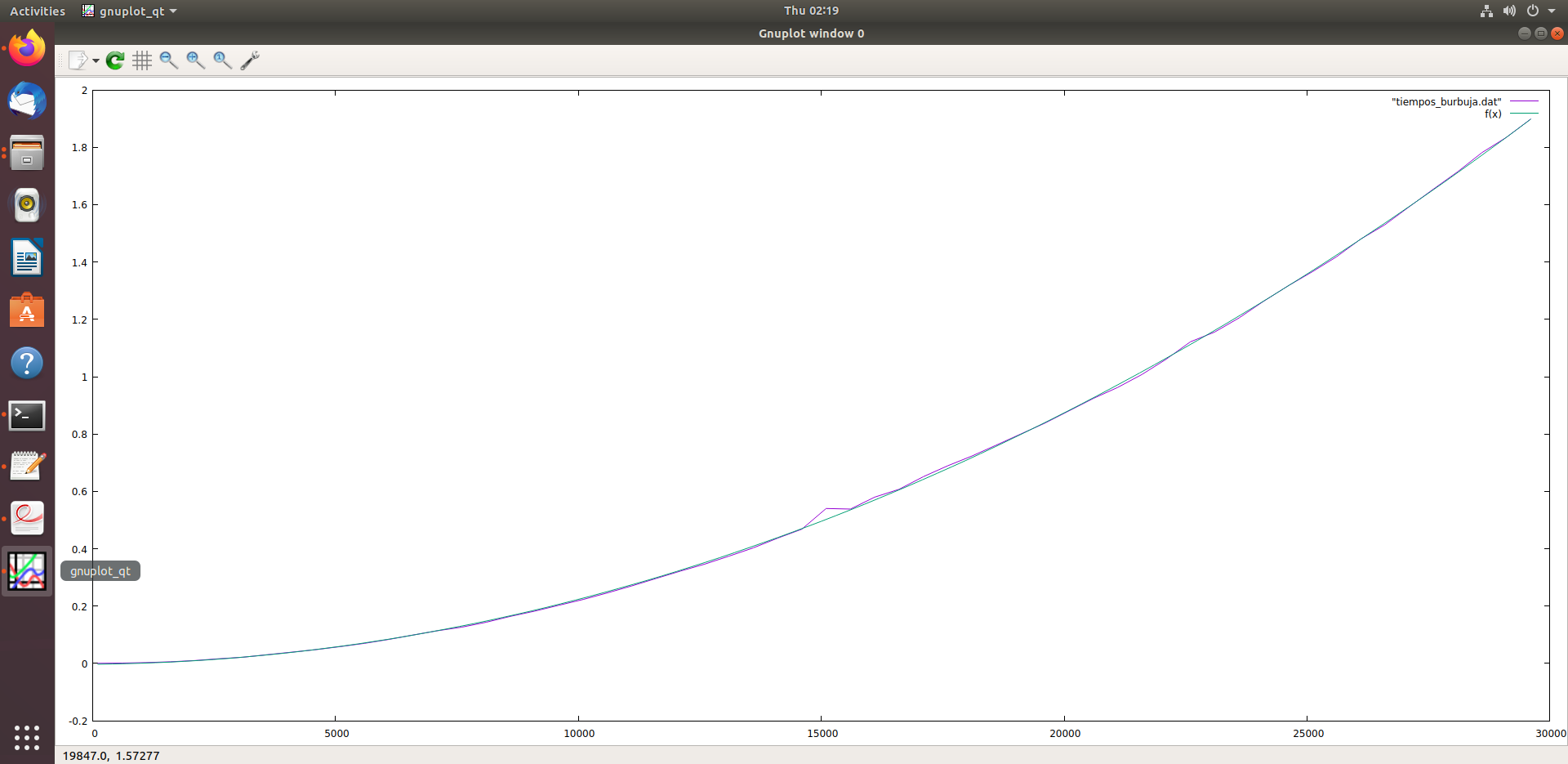


*2.* Ejercicio 2*: Ajuste en la ordenación de la burbuja*

Replique el experimento de ajuste por regresión a los resultados obtenidos en el ejercicio 1 que calculaba la eficiencia del algoritmo de ordenación de la burbuja. Para ello considere que f(x) es de la forma ax^2+bx+c

**Solución al ejercicio 2**

Tomando los datos del ejercicio anterior y usando gnuplot:



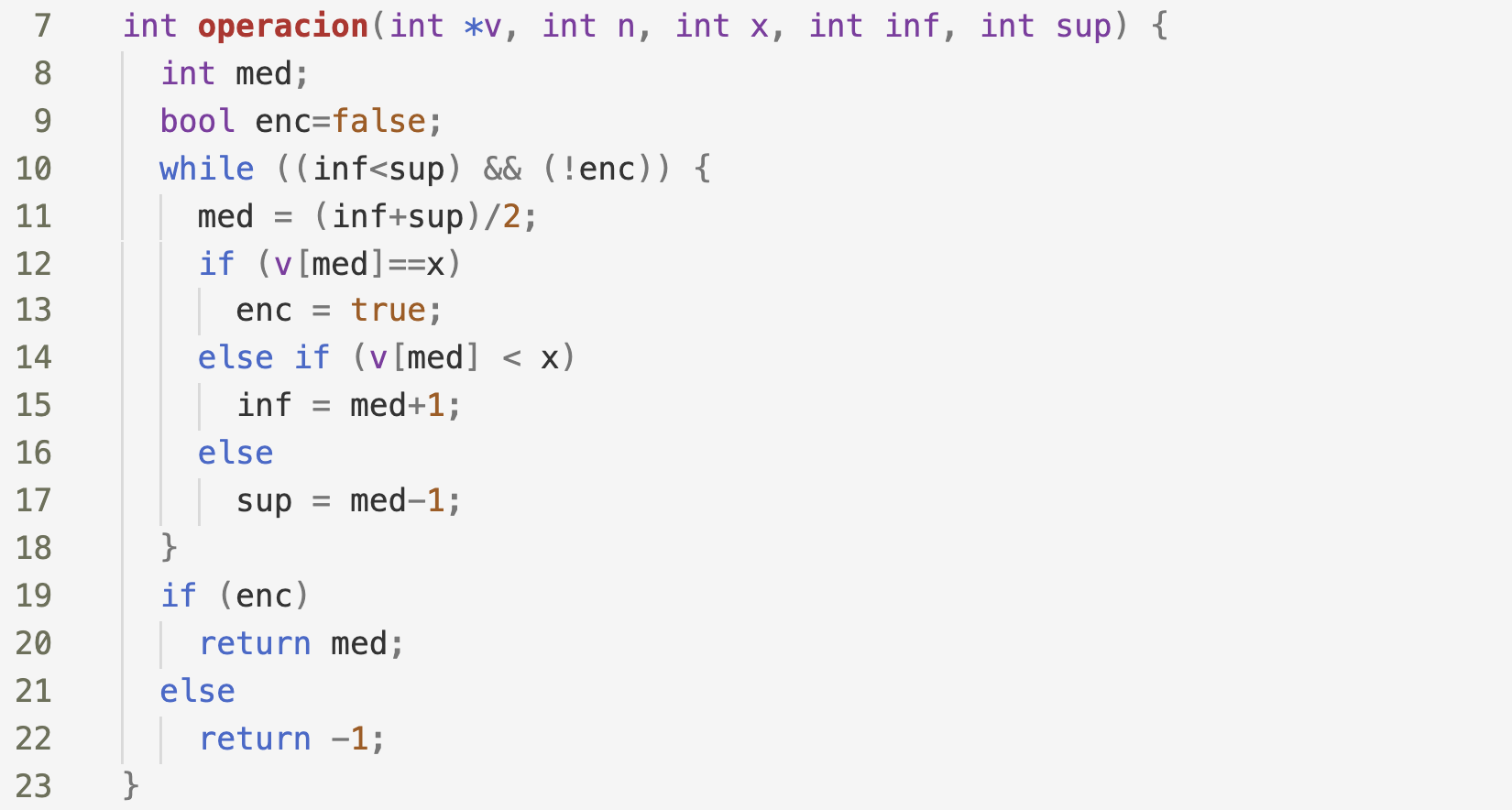
3. Ejercicio 3*: Problemas de precisión*

Junto con este guión se le ha suministrado un fichero ejercicio\_desc.cpp. En él se ha implementado un algoritmo. Se pide que:

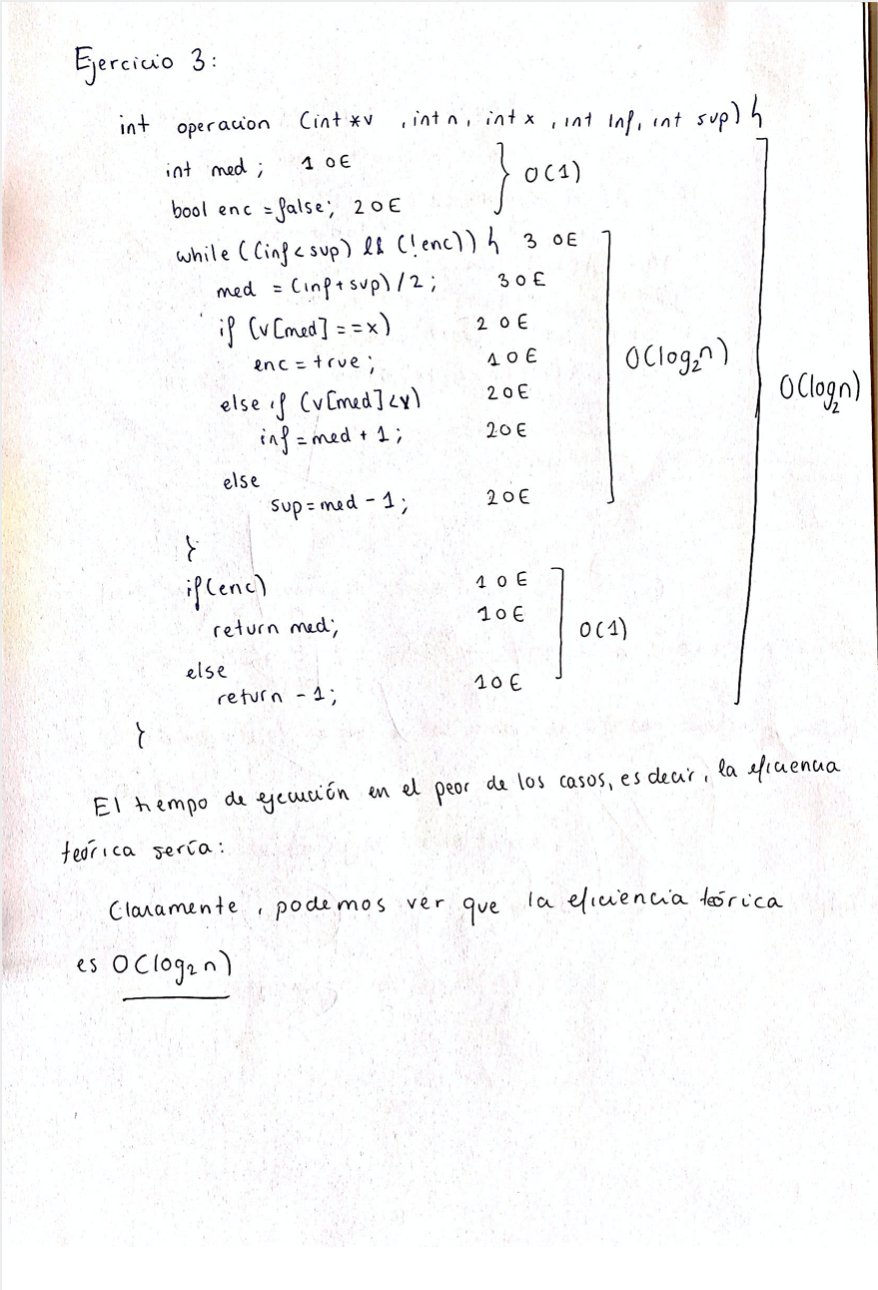
* Explique qué hace este algoritmo.
* Calcule su eficiencia teórica.
* Calcule su eficiencia empírica.  Si visualiza la eficiencia empírica debería notar algo anormal. Explíquelo y proponga una solución. Compruebe que su solución es correcta. Una vez resuelto el problema realice la regresión para ajustar la curva teórica a la empírica.

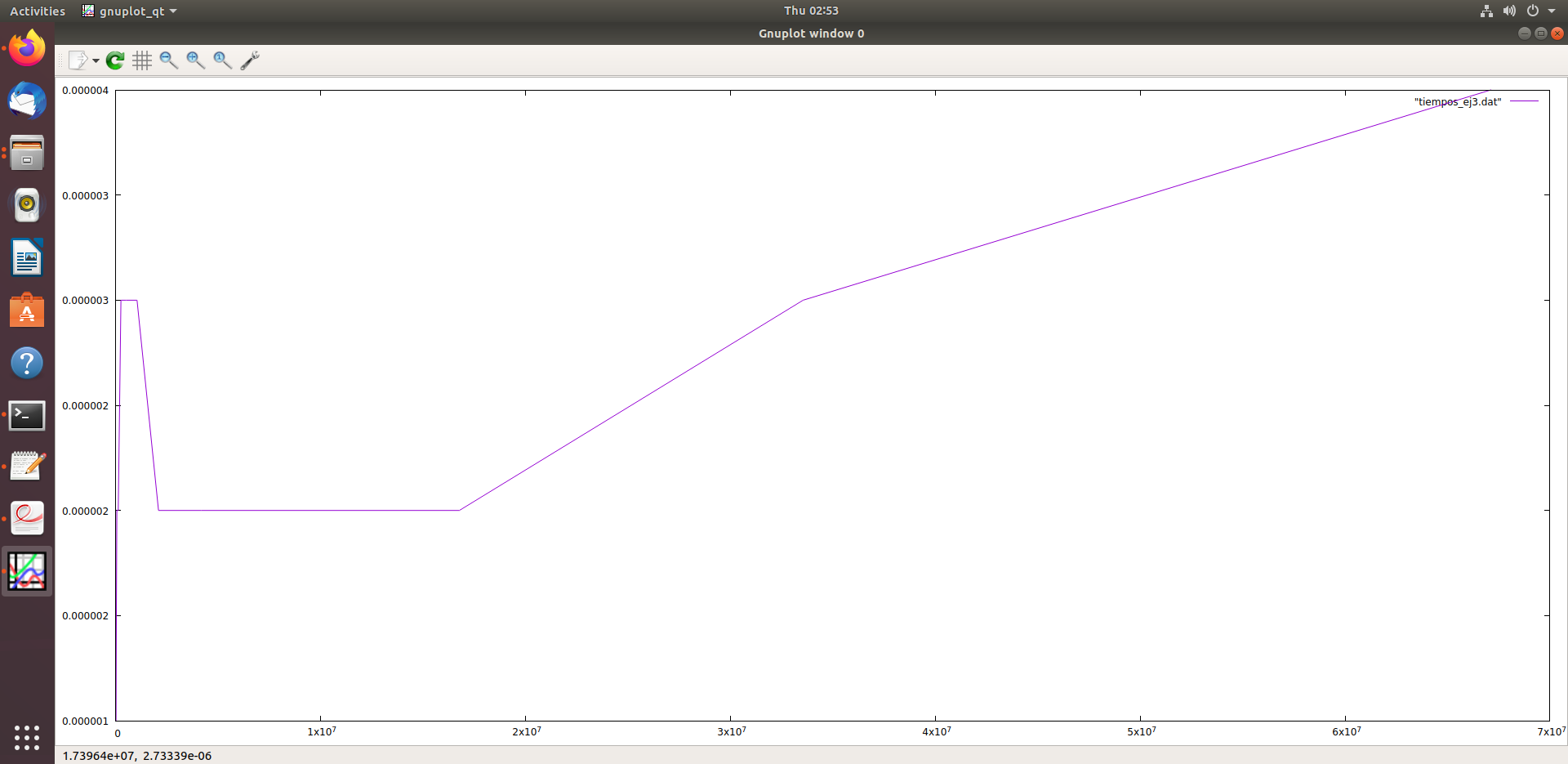
**Solución al ejercicio 3**

El algoritmo implementado en el fichero *ejercicio\_desc.cpp* es el siguiente:

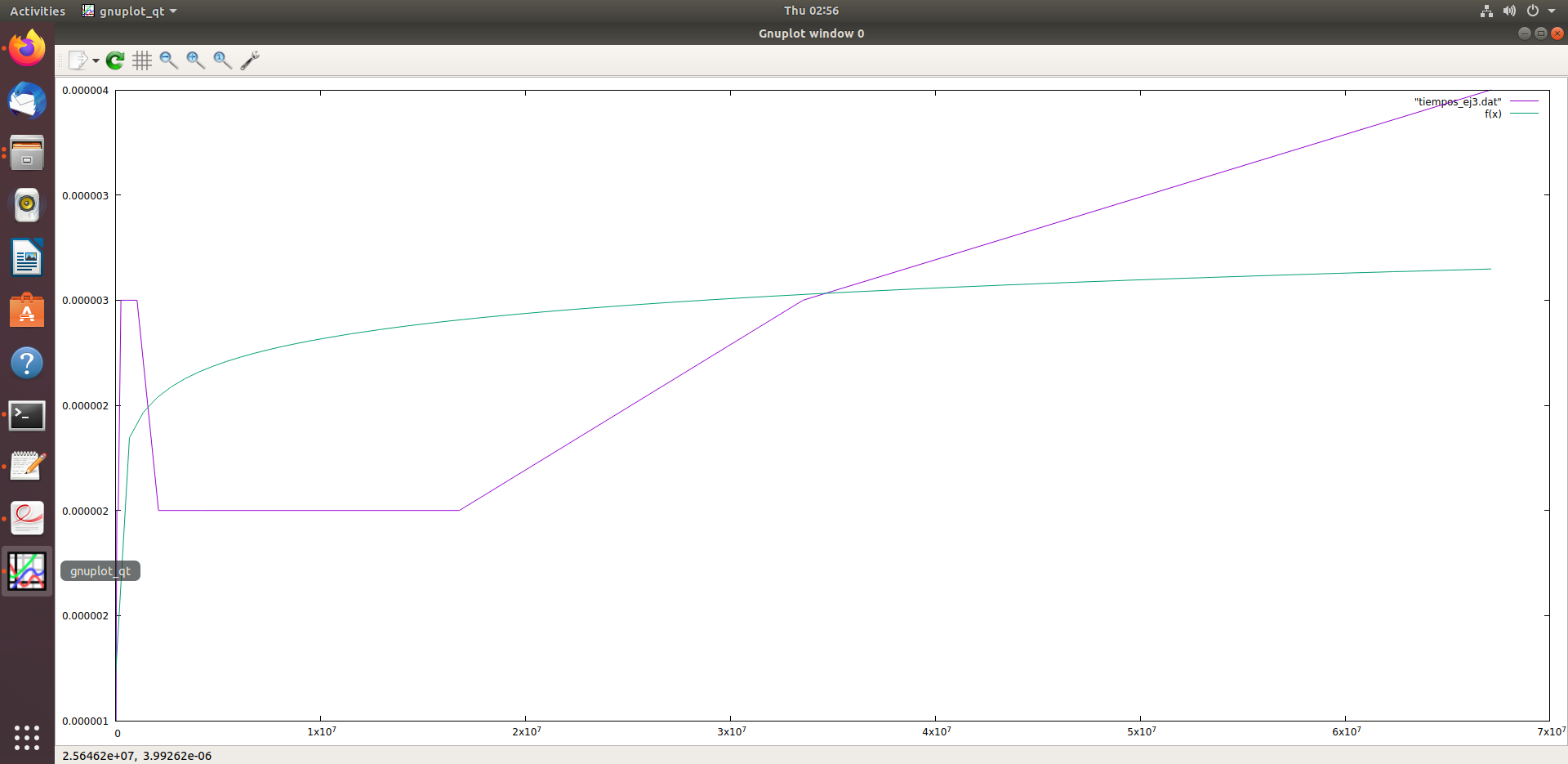


El algoritmo, es el de búsqueda binaria, el cual encuentra un elemento en un vector ordenado. Calculemos su eficiencia teórica (en papel):



Veamos si es cierto de forma empírica. para ello, ejecutaremos el algoritmo con distintos tamaños y realizaremos un plot, lo que nos da el siguiente resultado:

Como podemos ver en la gráfica, el ajuste de la gráfica respecto a la eficiencia teórica calculada con anterioridad no es muy bueno:



Esto ocurre debido a que, en el algoritmo, el vector debe de estar ordenado y el código genera un vector aleatorio.

4. Ejercicio 4*: Mejor y peor caso*

Retome el ejercicio de ordenación mediante el algoritmo de la burbuja. Debe modificar el código que genera los datos de entrada para situarnos en dos escenarios diferentes:

• El mejor caso posible. Para este algoritmo, si la entrada es un vector que ya está ordenado el tiempo de cómputo es menor ya que no tiene que intercambiar ningún elemento.

• El peor caso posible. Si la entrada es un vector ordenado en orden inverso estaremos en la peor situación posible ya que en cada iteración del bucle interno hay que hacer un intercambio.

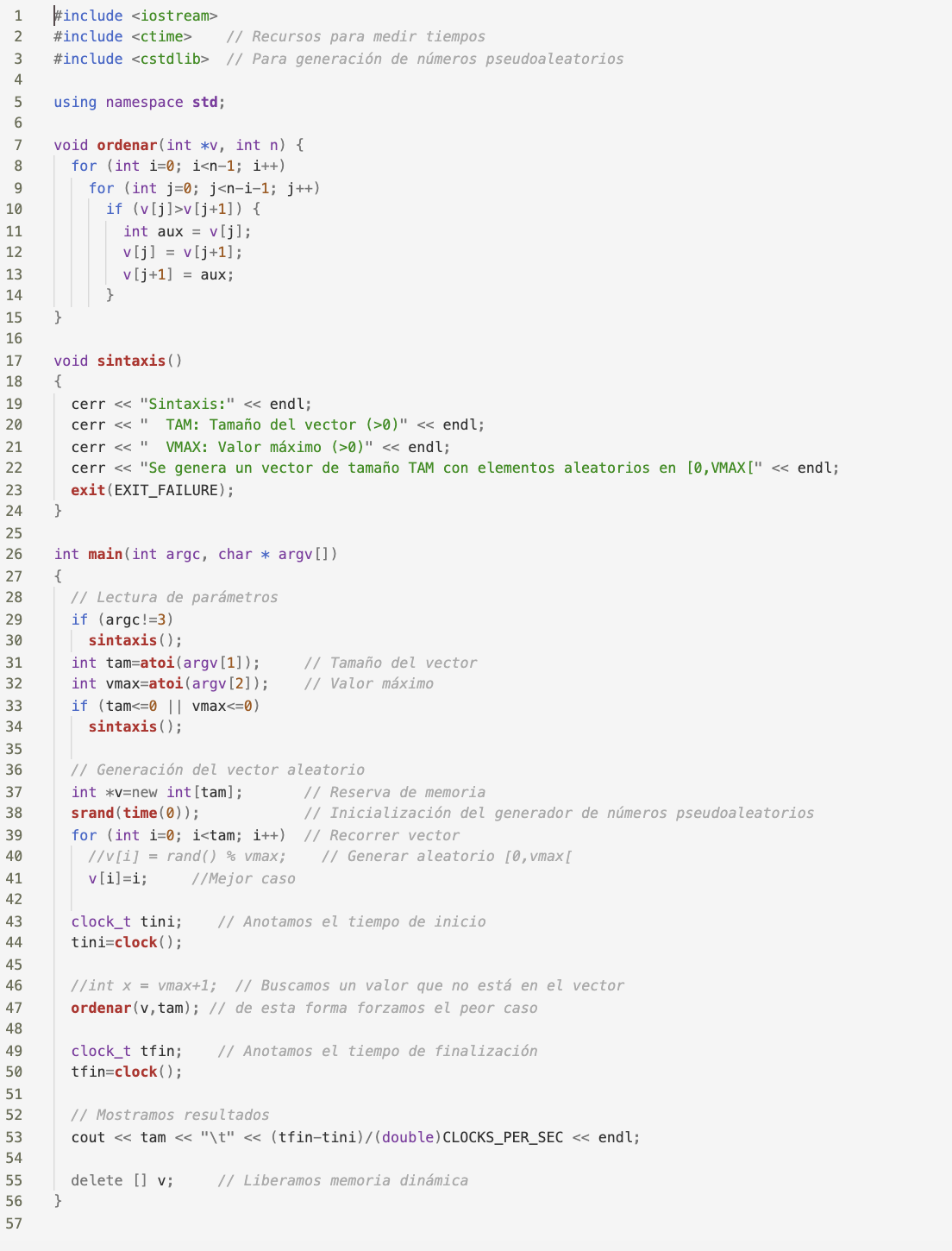
Calcule la eficiencia empírica en ambos escenarios y compárela con el resultado del ejercicio 1.

Aunque lo más frecuente será preguntar por números que estén en posiciones arbitrarias del vector (casos promedio).

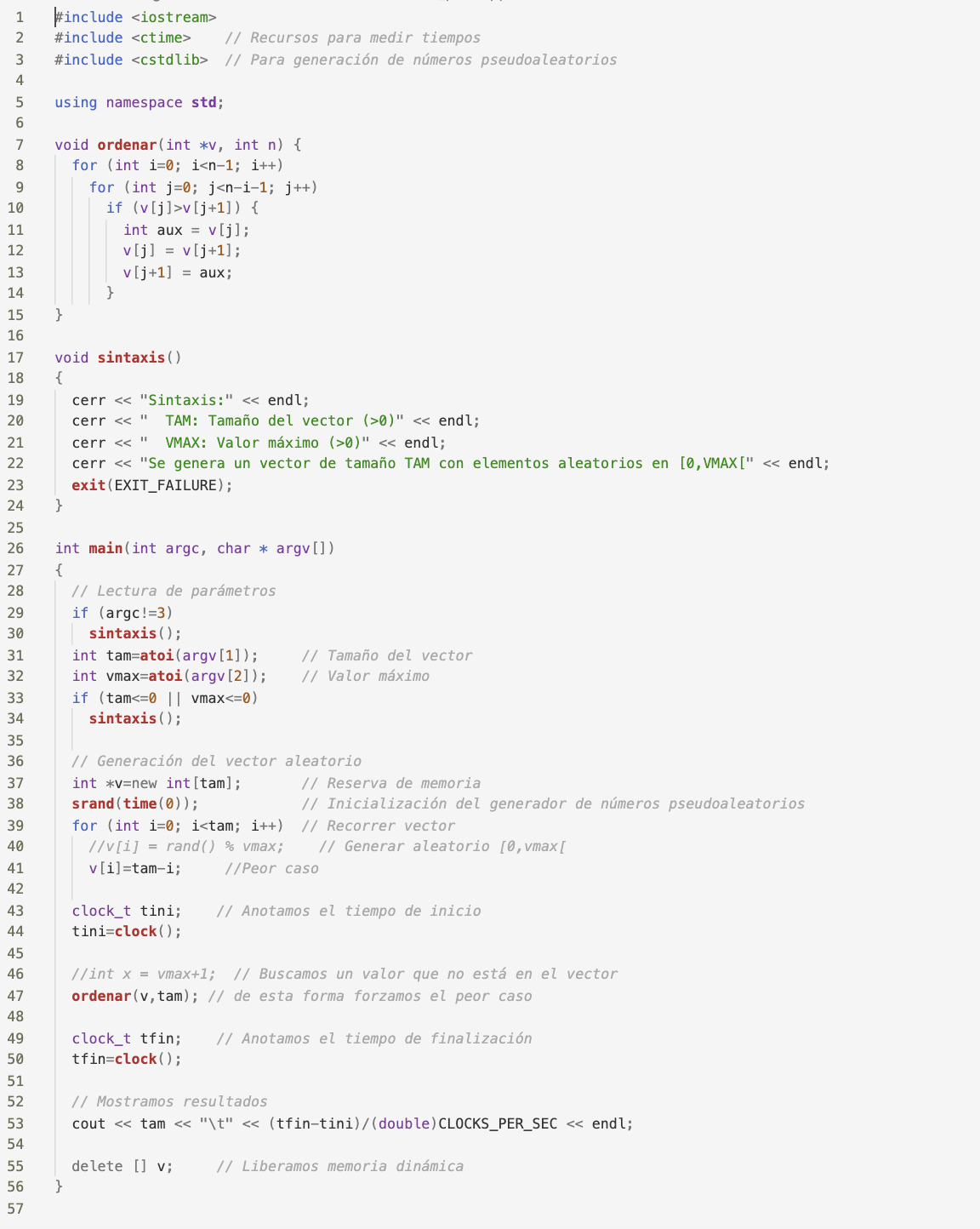
**Solución al ejercicio 4**

Tanto para el mejor como para el peor caso posible, la eficiencia teórica sería O(n2). Reajustaremos el programa *ordenacion\_burbuja.cpp* para que, en lugar de un vector ordenado aleatoriamente, genere un vector ordenado y un vector a la inversa, respectivamente.

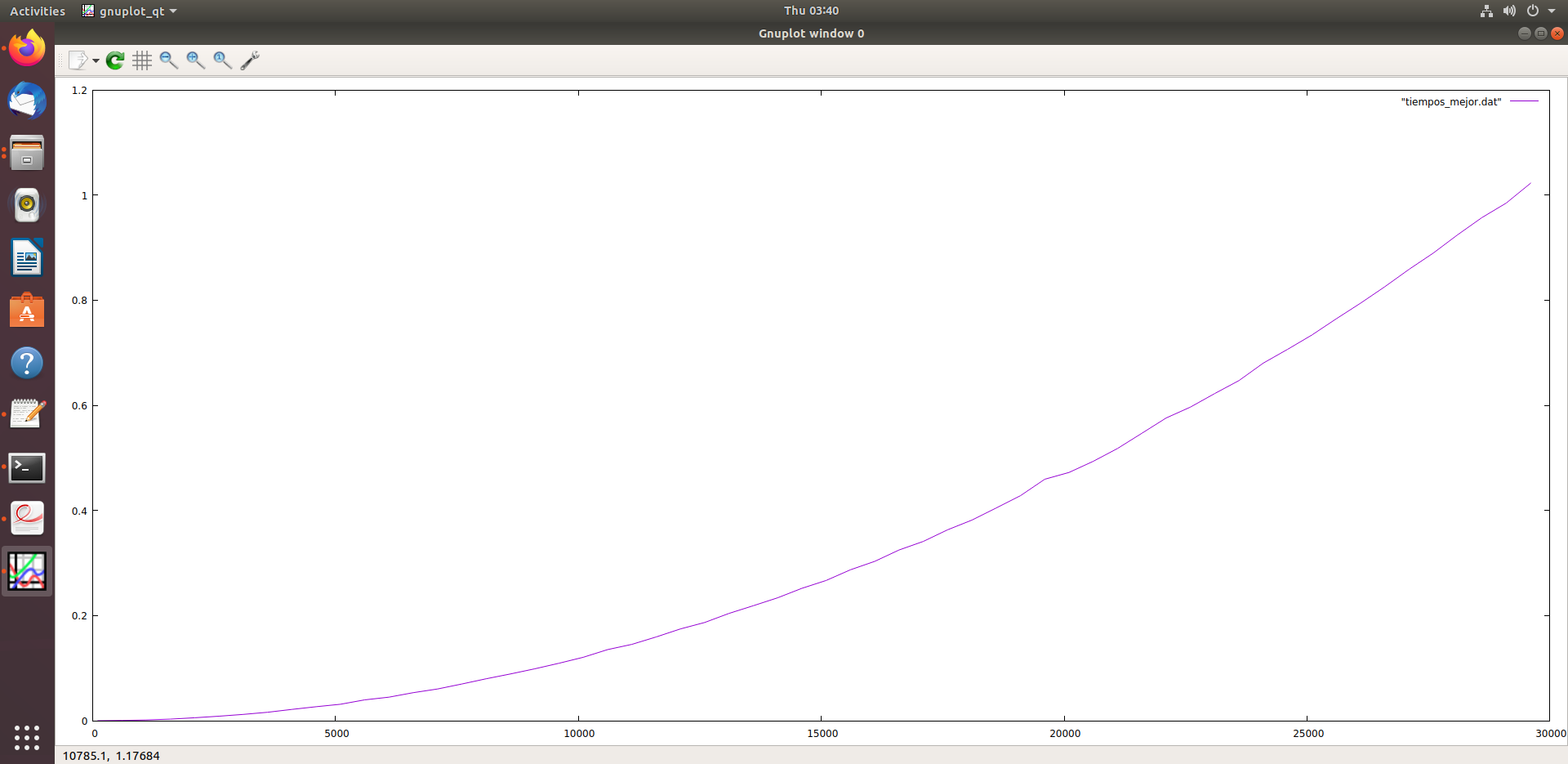
*ordenacion\_mejor.cpp*

**

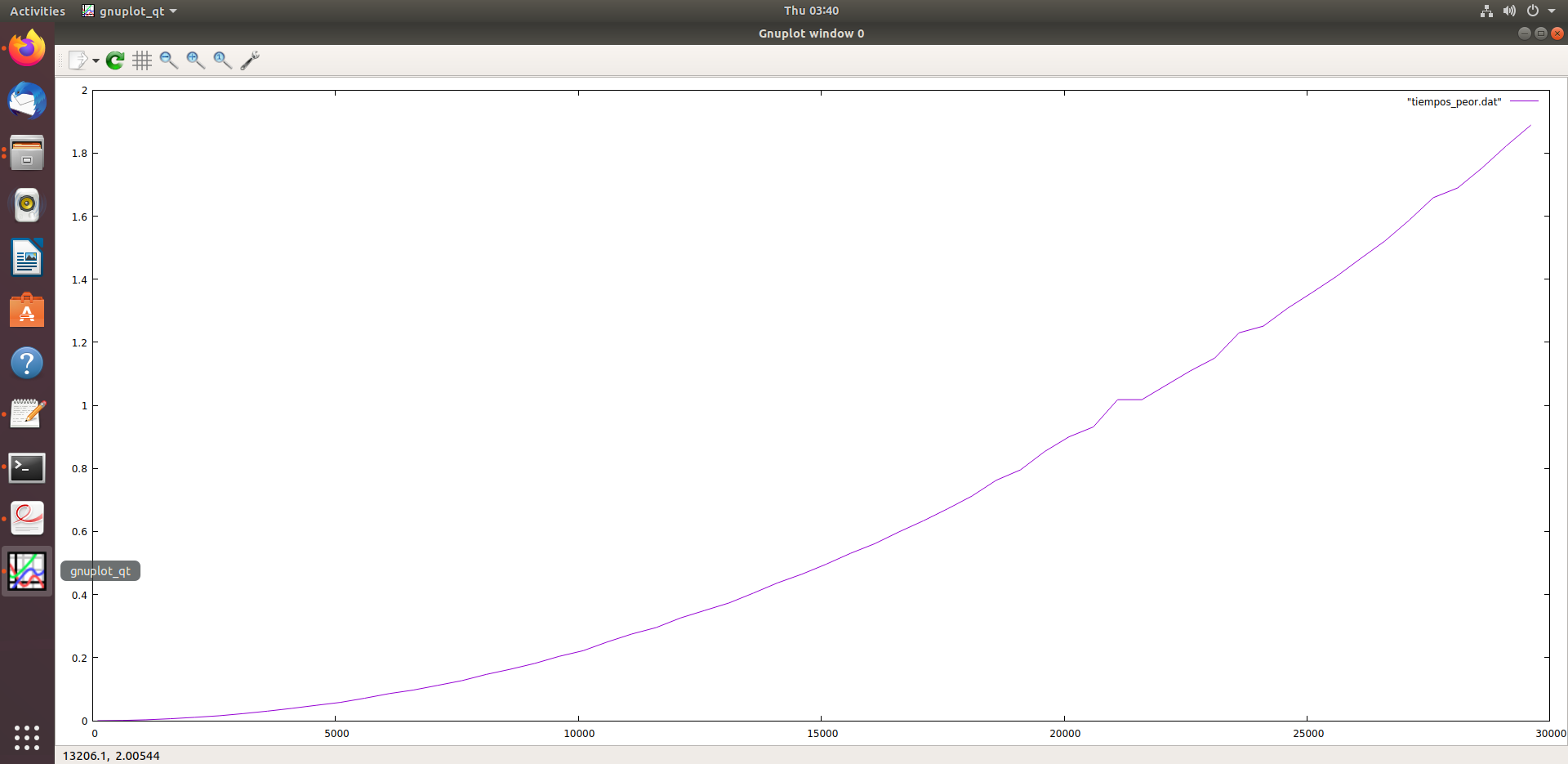
*ordenacion\_peor.cpp*



Si realizamos el plot de la gráfica de ambos ejercicios obtenemos para el **mejor** caso posible:

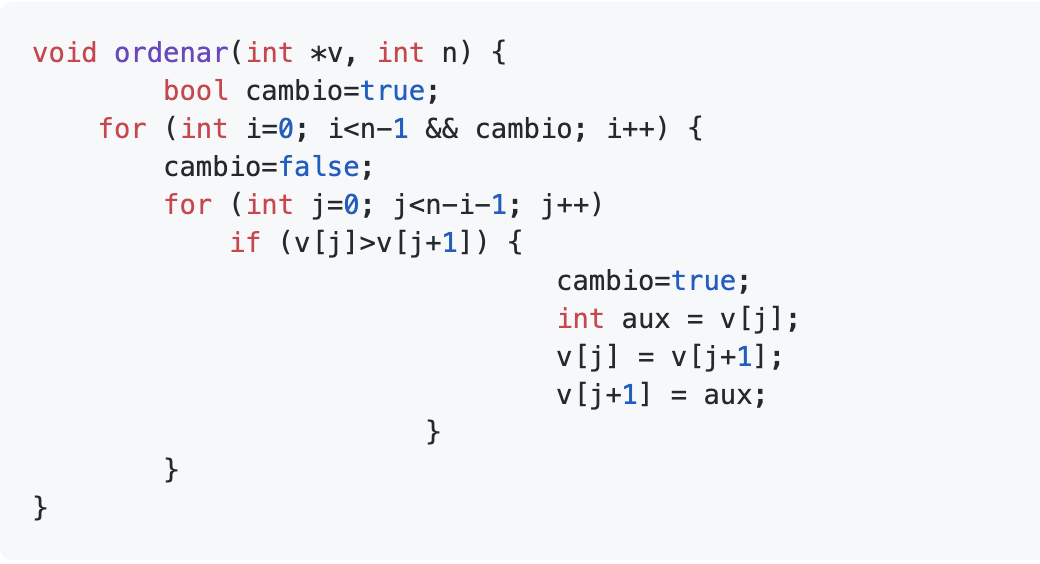


Y para el **peor** caso posible:



Nótese, que ambas gráficas son similares ya que ambas son del mismo orden. Sin embargo, es evidente que uno es más rápido que el otro al estar el vector ya ordenado.

5. Ejercicio 5*: Dependencia de la implementación*

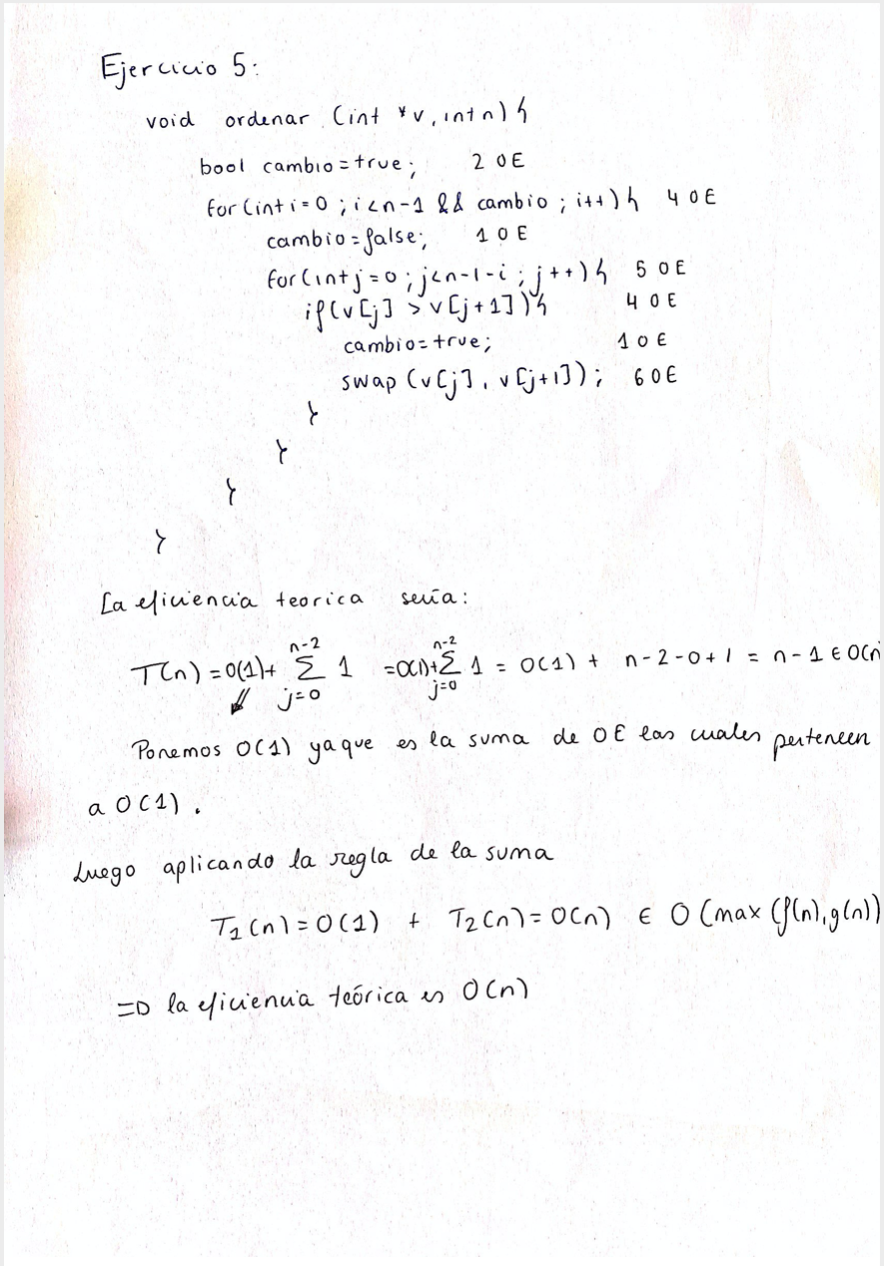
Considere esta otra implementación del algoritmo de la burbuja:

En ella se ha introducido una variable que permite saber si, en una de las iteraciones del bucle externo no se ha modificado el vector. Si esto ocurre significa que ya está ordenado y no hay que continuar.

Considere ahora la situación del mejor caso posible en la que el vector de entrada ya está ordenado. ¿Cuál sería la eficiencia teórica en ese mejor caso? Muestre la gráfica con la eficiencia empírica y compruebe si se ajusta a la previsión.

**Solución al ejercicio 5**

Consideremos la situación del mejor caso posible en la que el vector de entrada v ya este ordenado. Calculemos la eficiencia teórica:



Claramente solo se recorrería el vector una vez.

La eficiencia empírica, como podemos comprobar tiene forma lineal ajustándose a lo previsto anteriormente con la eficiencia teórica como vemos en ambas gráficas.

