Universidad de Granada

GRADO EN INGENIERÍA INFORMÁTICA

ESTRUCTURA DE DATOS
PRÁCTICA 1

EJERCICIOS 7.1, 7.2 Y 7.3

Fernández Fernández, Sergio
2ºB GRUPO BII

Granada, octubre de 2017

EJERCICIO 7.1

```cpp
#include<iostream>
#include<ctime>
#include<cstdlib>
using namespace std;

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;
            int aux = v[j];
            v[j] = v[j+1];
            v[j+1] = aux;
        }
    }
}

int main(int argc, char * argv[]) {

    int tam=atoi(argv[1]);      // Tamaño del vector
    int vmax=atoi(argv[2]);     // Valor máximo

    int *v= new int[tam];
    srand(time(0));
    for(int i=0; i<tam; i++)
        v[i]= rand() % vmax;

    clock_t tini;
    tini=clock();

    ordenar(v,tam);

    clock_t tfin;
    tfin=clock();

    cout << tam << "\t" << (tfin-tini)/(double)CLOCKS_PER_SEC << endl;
    delete [] v;


}
```

CPU: Intel(R) Celeron(R) CPU N2820 @ 2.13GHz

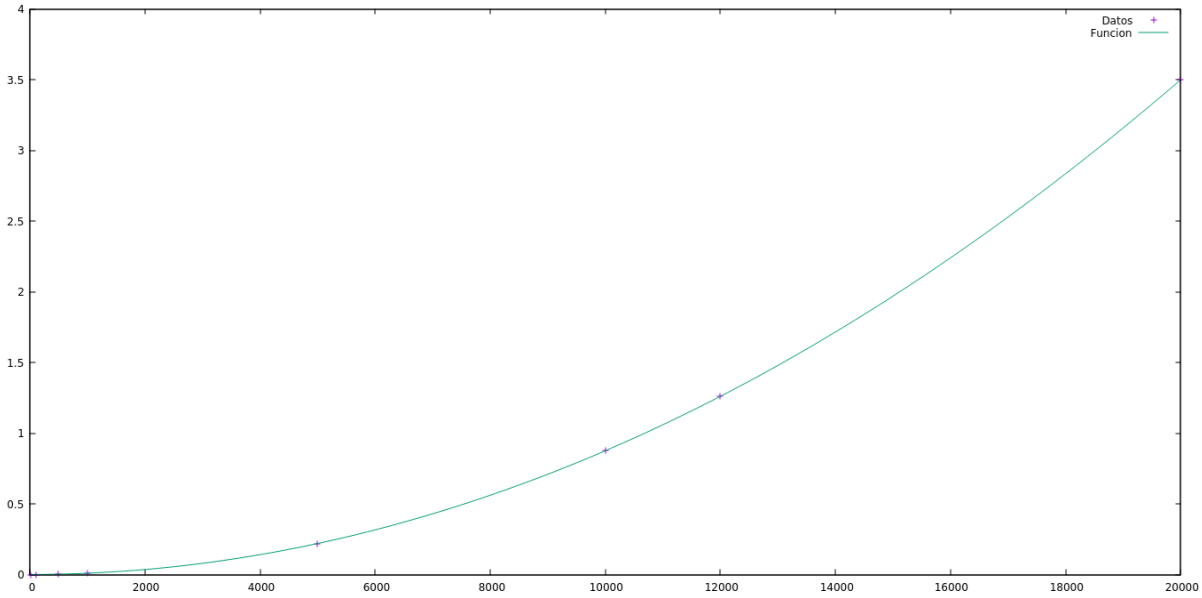
Velocidad Reloj: 2181.355

S.O: Ubuntu 16.04

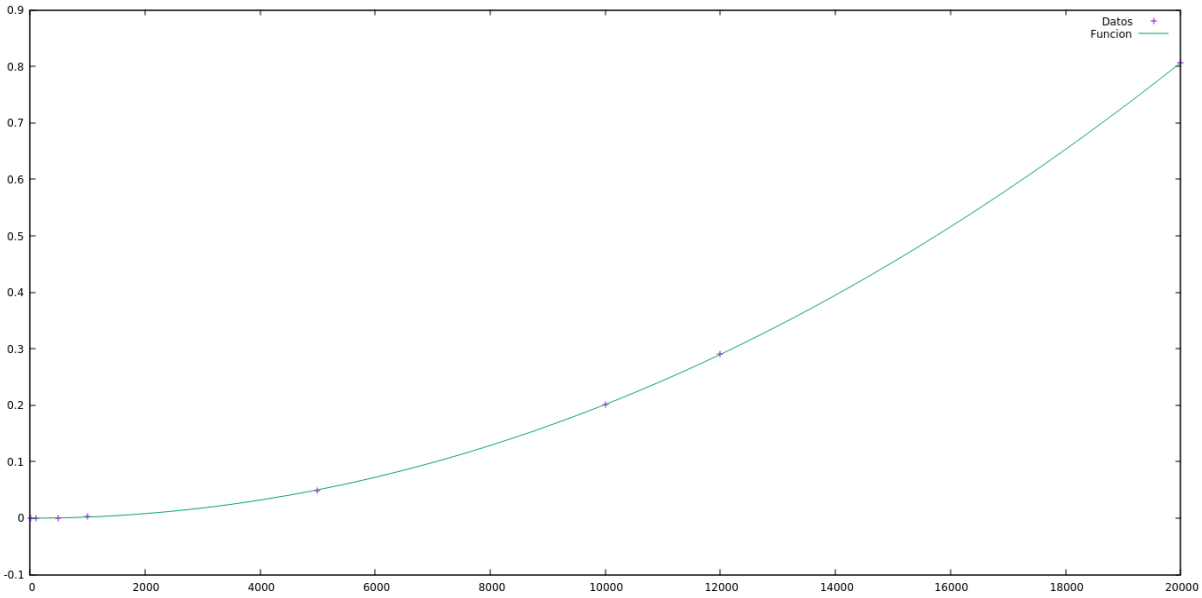
Opciones compilacion: g++ -O3 -std=c++11 -o eficiencia_ordenar_exe eficiencia_ordenar.cpp

g++ -std=c++11 -o eficiencia_ordenar_exe eficiencia_ordenar.cpp

GRÁFICA SIN OPTIMIZACIÓN



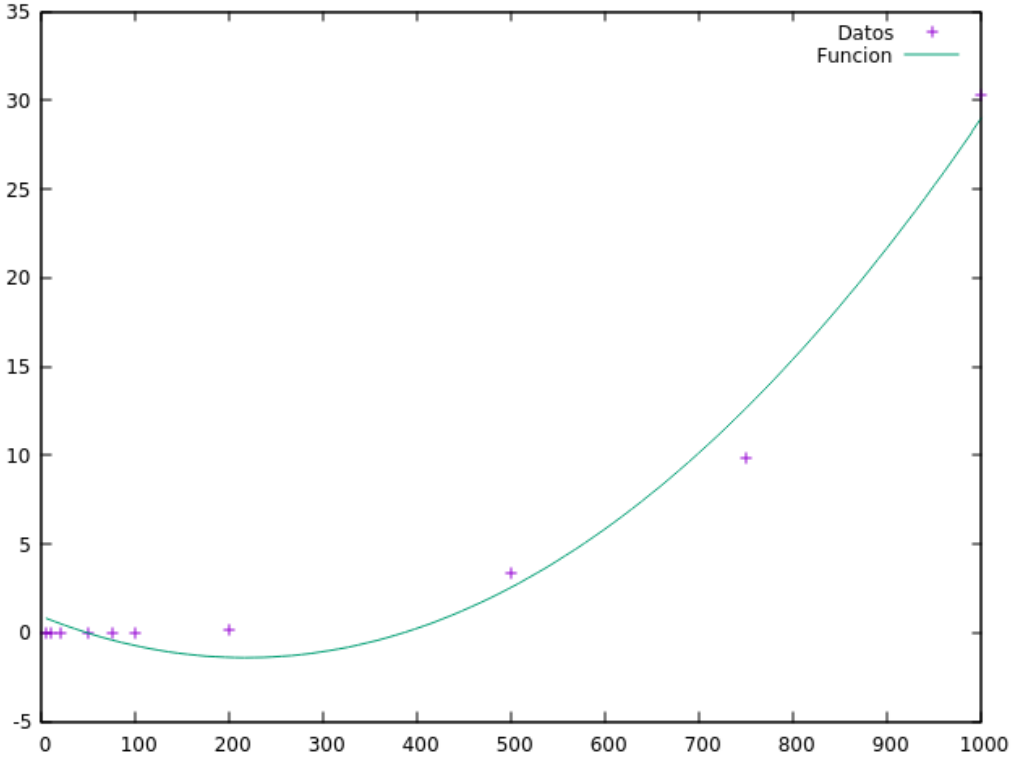
GRÁFICA CON OPTIMIZACIÓN



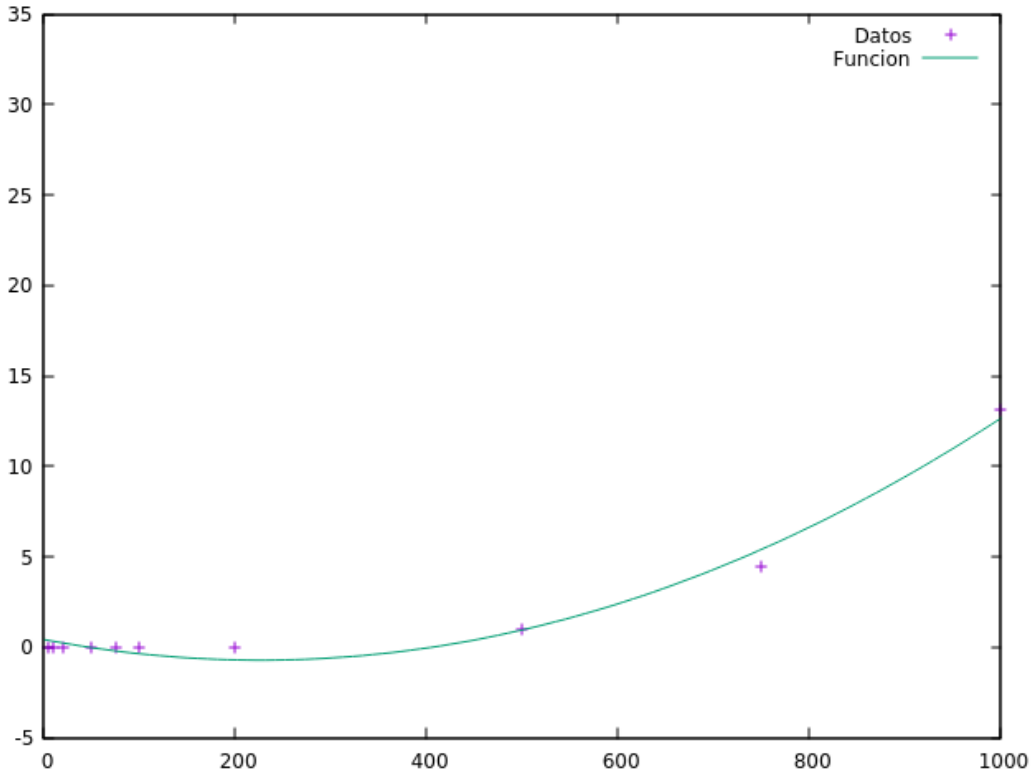
EJERCICIO 7.2

```cpp
#include<iostream>
#include<ctime>
#include<cstdlib>
using namespace std;

//Programa principal
int main(int argc, char * argv[]){

   int util_filas = atoi(argv[1]);
   int util_columnas = atoi(argv[2]);

   int *matriz_1;
   int *matriz_2;
   int *matriz_resultado;
   matriz_1 = new int[util_filas*util_columnas];
   matriz_2 = new int[util_filas*util_columnas];
   matriz_resultado = new int[util_filas*util_columnas];

   //Insertar valores dentro de la primera matriz
   srand(time(0));
   for (int i=0; i<util_filas; i++){
        for (int j=0; j<util_columnas; j++){
            matriz_1[i*util_columnas+j] = rand() % 50000;
        }
   }

   //Insertar valores dentro de la segunda matriz
   srand(time(0));
   for (int i=0; i<util_filas; i++){
        for (int j=0; j<util_columnas; j++){
            matriz_2[i*util_columnas+j] = rand() % 50000;
        }
   }


    clock_t tini;
    tini=clock();

   //Calcula la multiplicacion entre las dos matrices
   for(int i=0; i<util_filas; i++){
    for(int j=0; j<util_columnas; j++){
        for(int k=0; k<util_columnas; k++)
            matriz_resultado[i*util_columnas+j] += matriz_1[i*util_columnas+k] * matriz_2[k*util_columnas+j] ;

        }
    }

    clock_t tfin;
    tfin=clock();

    cout << util_filas << "\t" << (tfin-tini)/(double)CLOCKS_PER_SEC << endl;

    delete [] matriz_1;
    delete [] matriz_2;
    delete [] matriz_resultado;
}
```

SIN OPTIMIZAR

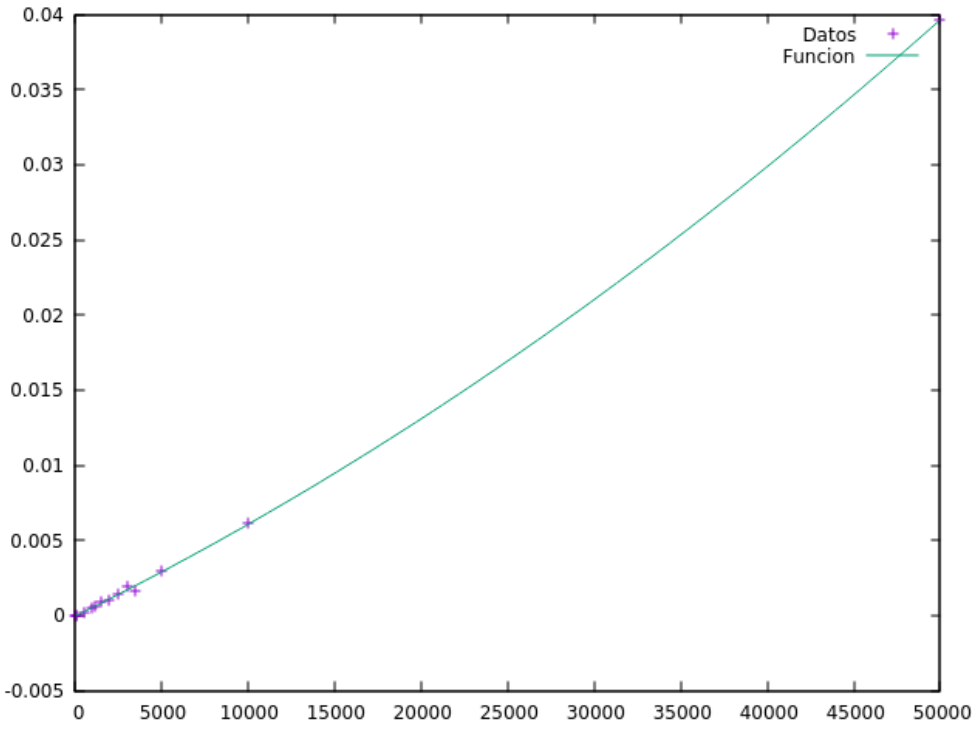


OPTIMIZADO



EJERCICIO 7.3

SIN OPTIMIZAR



OPTIMIZADO