EJERCICIO 5

Sistema operativo: Ubuntu 18.04.1 LTS
Compilador usado: ii g++ 4:7.3.0-3ubuntu2 amd64 GNU C++ compiler

He creado el archivo “ordenacion_ej5.cpp”, en el que primero he ordenado el vector y luego he calculado cuánto tardaría en ordenarlo con la función nueva:

```
#include <iostream>
#include <ctime>    // Recursos para medir tiempos
#include <cstdlib>  // Para generación de números pseudoaleatorios

using namespace std;

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;
      }
    }
}

void ordenar2 (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;
      }
    }
  }
}

void sintaxis()
{
  cerr << "Sintaxis:" << endl;
  cerr << "  TAM: Tamaño del vector (>0)" << endl;
  cerr << "  VMAX: Valor máximo (>0)" << endl;
  cerr << "Se genera un vector de tamaño TAM con elementos aleatorios en [0,VMAX[" << endl;
  exit(EXIT_FAILURE);
}

int main(int argc, char * argv[])
{
  // Lectura de parámetros
  if (argc!=3)
```

```
    sintaxis();
  int tam=atoi(argv[1]);     // Tamaño del vector
  int vmax=atoi(argv[2]);    // Valor máximo
  if (tam<=0 || vmax<=0)
    sintaxis();

  // Generación del vector aleatorio
  int *v=new int[tam];       // Reserva de memoria
  srand(time(0));            // Inicialización del generador de números pseudoaleatorios
  for (int i=0; i<tam; i++)  // Recorrer vector
    v[i] = rand() % vmax;    // Generar aleatorio [0,vmax[

  ordenar(v,tam);

  clock_t tini;    // Anotamos el tiempo de inicio
  tini=clock();

  ordenar2(v,tam);

  clock_t tfin;    // Anotamos el tiempo de finalización
  tfin=clock();

  // Mostramos resultados
  cout << tam << "\t" << (tfin-tini)/(double)CLOCKS_PER_SEC << endl;

  delete [] v;     // Liberamos memoria dinámica
}
```

“ejecuciones_ordenacion_ej5.csh”:

```
#!/bin/csh
@ inicio = 100
@ fin = 30000
@ incremento = 500
set ejecutable = ordenacion_ej5
set salida = tiempos_ordenacion_ej5.dat

@ i = $inicio
echo > $salida
while ( $i <= $fin )
  echo Ejecución tam = $i
  echo `./{$ejecutable} $i 10000` >> $salida
  @ i += $incremento
end
```

“tiempos_ordenacion_ej5.dat”:

```
100 2e-06
600 3e-06
```

1100 6e-06
1600 6e-06
2100 7e-06
2600 8e-06
3100 1e-05
3600 1e-05
4100 1.1e-05
4600 1.9e-05
5100 1.3e-05
5600 1.5e-05
6100 1.6e-05
6600 1.9e-05
7100 1.9e-05
7600 1.9e-05
8100 2.1e-05
8600 2.2e-05
9100 2.3e-05
9600 2.5e-05
10100 2.6e-05
10600 2.7e-05
11100 4.3e-05
11600 3e-05
12100 3e-05
12600 3.2e-05
13100 3.3e-05
13600 3.5e-05
14100 3.6e-05
14600 3.7e-05
15100 3.8e-05
15600 4.8e-05
16100 4.2e-05
16600 4.2e-05
17100 4.5e-05
17600 4.4e-05
18100 4.7e-05
18600 4.7e-05
19100 4.9e-05
19600 4.8e-05
20100 5e-05
20600 5.1e-05
21100 5.2e-05
21600 7.7e-05
22100 5.7e-05
22600 5.8e-05
23100 5.7e-05
23600 6e-05
24100 6.3e-05
24600 6.1e-05
25100 6.2e-05
25600 6.3e-05
26100 6.5e-05
26600 6.6e-05

27100 6.7e-05
27600 6.8e-05

Y la gráfica obtenida: