

## Practica 1 – Eficiencia

### Ejercicio 6

#### Características del computador usado:

- Procesador: Intel(R) Core(TM) i7-2600 CPU @ 3.40GHz
- Ram: 8 GB
- SO: Ubuntu 16.04 (Linux version 4.4.0-96-generic)
- Compilador: gcc version 5.4.0 20160609 (Ubuntu 5.4.0-6ubuntu1~16.04.4)

NOTA: En la compilación de las pruebas realizadas no se han utilizado flags de optimización ni de otro tipo.

#### Algoritmo

```
#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 sintaxis()
{
    cerr << "Sintaxis:" << endl;
    cerr << " TAM: Tamaño del vector (>0)" << endl;
    cerr << "Se genera un vector de tamaño TAM con elementos aleatorios" << endl;
    exit(EXIT_FAILURE);
}

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

    // Lectura de parámetros
```

Angel Barrilao Bensorhir

```
    if (argc!=2){
        sintaxis();
    }

    int tam=atoi(argv[1]);    // Tamaño del vector

    // 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();    // Generar aleatorio [0,vmax[

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

    ordenar(v,tam); // Llamamos a la funcion

    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
}
```

## **Script**

```
#!/bin/csh
@ inicio = 100
@ fin = 30000
@ incremento = 100
set ejecutable = ordenacion_optimizado
set salida = tiempos_ordenacion.dat

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

Eficiencia Empírica comparativa

