**Análisis del algoritmo propuesto en la solución #2**

/\*

Mauricio Iván Cortés García

A00816689

Análisis y Diseño de Algoritmos

Prof. Roman Martinez

Tarea 2

Solución 2

SolucionDos.cpp

\*/

#include <iostream>

#include <fstream>

#include <algorithm>

#include <vector>

#include <time.h>

using namespace std;

/\*

Nombre: BusquedaBinaria.

Entradas: Vector\_Mats(vector<int>), Busca\_Mats(int).

Salidas: Booleano.

Función: Aplicar la busqueda binaria a un vector con las matrículas y regresar true si la encuentra y false si no la encuentra.

\*/

bool BusquedaBinaria(vector<int> Vector\_Mats, int Busca\_Mats)

{

O(n log n)

int Central, Bajo, Valor\_Central, Largo, Alto;

Bajo=0;

Largo=Vector\_Mats.size();

Alto=Largo-1;

while(Bajo<=Alto)

{

Central=(Bajo+Alto)/2;

Valor\_Central = Vector\_Mats[Central];

if(Busca\_Mats==Valor\_Central)

return true;

else if(Busca\_Mats<Valor\_Central)

Alto=Central-1;

else

Bajo=Central+1;

}

return false;

}

int main()

{

int Matriculas, Busca\_Mat;

vector <int> Vector\_Mats;

ifstream Archivo\_Entrada;

ofstream Archivo\_Salida;

//Pregunta al usuario por la matrícula a buscar en el vector de matrículas y lo almacena en Busca\_Mat.

cout << "Que matricula deseas buscar?" << endl;

cin >> Busca\_Mat;

clock\_t startTime = clock();

Archivo\_Entrada.open("Archivo\_Matriculas.txt");//Abre el archivo.

while(!Archivo\_Entrada.eof())//Recorre el archivo hasta que se acaben los datos.

O(n)

{

//Leer los datos del archivo y almacenarlos en el vector Vector\_Mats.

Archivo\_Entrada >> Matriculas;

Vector\_Mats.push\_back(Matriculas);

}

sort(Vector\_Mats.begin(), Vector\_Mats.end());//Se aplica el sort de STL al Vector para ordenar las matrículas de menor a mayor.

O(n log n)

//Si la función BusquedaBinaria regresa true, la matrícula si es de un EXATEC, de lo contrario no lo es.

BusquedaBinaria(Vector\_Mats, Busca\_Mat) ? cout << "Si es EXATEC." << endl : cout << "No es EXATEC." << endl;

Archivo\_Entrada.close();//Se cierra el archivo de matriculas.

//Despliega en un archivo .txt el tiempo de ejecucion del programa.

Archivo\_Salida.open("Tiempo de ejecucion Solucion Dos.txt");

Archivo\_Salida << double(clock()-startTime)/(double)CLOCKS\_PER\_SEC << " segundos." << endl;

Archivo\_Salida.close();

system("PAUSE");

return 0;

}

Orden de complejidad de tiempo: O(n log n)

Orden de complejidad de espacio: O(n)

Tipo de algoritmo: No Every-case