**“main.cpp”**

#include "iostream"

#include "medicos.h"

#include "citas.h"

using namespace std;

void procesarPeticiones(const tListaCitas& listaCitas, tListaMedicos& listaMedicos);

int main()

{

bool ok1, ok2;

tListaMedicos listaMedicos;

tListaCitas listaCitas;

\_CrtSetDbgFlag(\_CRTDBG\_ALLOC\_MEM\_DF | \_CRTDBG\_LEAK\_CHECK\_DF);

inicializarListaMedicos(listaMedicos);

inicializarListaCitas(listaCitas);

ok1 = leerMedicos(listaMedicos);

ok2 = leerCitas(listaCitas);

if (ok1 && ok2)

{

mostrarMedicos(listaMedicos);

mostrarPeticiones(listaCitas);

procesarPeticiones(listaCitas,listaMedicos);

mostrarMedicos(listaMedicos);

destruir(listaCitas);

destruir(listaMedicos);

}

return 0;

}

void procesarPeticiones(const tListaCitas& listaCitas, tListaMedicos& listaMedicos)

{

int codigo\_medico, libres;

tMedicoPtr ptr\_medico;

float total = 0;

int num\_citas = getNumElems(listaCitas);

for (int i = 0; i < num\_citas; i++)

{

codigo\_medico = getCodigoMedico(listaCitas, i);

ptr\_medico = buscar(listaMedicos, codigo\_medico);

if (ptr\_medico != nullptr)

{

total += getTarifaMedico(ptr\_medico);

libres = getLibresMedico(ptr\_medico);

setLibresMedico(ptr\_medico, libres - 1);

if (libres-1 == 0)

eliminar(listaMedicos, ptr\_medico);

}

}

cout << "El importe total es :" << total << " euros" << endl << endl;

}

**“médicos.h”**

#include <string>

using namespace std;

const int MAX\_MEDICOS = 100;

typedef struct

{

int codigo;

int libres;

float tarifa;

} tMedico;

typedef tMedico\* tMedicoPtr;

typedef tMedicoPtr tArray[MAX\_MEDICOS];

typedef struct

{

tArray medicos;

int contador;

} tListaMedicos;

void inicializarListaMedicos(tListaMedicos& listaMedicos);

bool leerMedicos(tListaMedicos &lista);

void mostrarMedicos(const tListaMedicos& listaMedicos);

tMedicoPtr buscar(const tListaMedicos &listaMedicos, int codigo);

int getTarifaMedico(tMedicoPtr ptr\_medico);

int getLibresMedico(tMedicoPtr ptr\_medico);

void setLibresMedico(tMedicoPtr ptr\_medico, int libres);

void eliminar(tListaMedicos& listaMedicos, tMedicoPtr ptr\_medico);

void destruir(tListaMedicos& listaMedicos);

**“médicos.cpp”**

#include <fstream>

#include <iostream>

#include "medicos.h"

using namespace std;

int buscarPosMedico(const tListaMedicos& listaMedicos, tMedicoPtr ptr\_medico);

void inicializarListaMedicos(tListaMedicos& listaMedicos)

{

listaMedicos.contador = 0;

}

bool leerMedicos(tListaMedicos& lista)

{

bool cargado = true;

ifstream entrada;

tMedico aux;

int codigo;

entrada.open("medicos.txt");

if (entrada.is\_open())

{

entrada >> codigo;

while (codigo != -1 && lista.contador < MAX\_MEDICOS)

{

aux.codigo = codigo;

entrada >> aux.libres >> aux.tarifa;

lista.medicos[lista.contador] = new tMedico(aux);

lista.contador++;

entrada >> codigo;

}

entrada.close();

}

else

{

cout << "Error, no se pudo abrir el archivo 'Medicos.txt'" << endl;

cargado = false;

}

return cargado;

}

tMedicoPtr buscar(const tListaMedicos& listaMedicos, int codigo)

{

tMedicoPtr ptr\_medico = nullptr;

bool encontrado = false;

int inicio = 0;

int fin = listaMedicos.contador - 1;

int mitad;

while (inicio <= fin && !encontrado)

{

mitad = (inicio + fin) / 2;

if (listaMedicos.medicos[mitad]->codigo == codigo)

{

ptr\_medico = listaMedicos.medicos[mitad];

encontrado = true;

}

else if (codigo > listaMedicos.medicos[mitad]->codigo)

fin = mitad - 1;

else

inicio = mitad + 1;

}

return ptr\_medico;

}

int getTarifaMedico(tMedicoPtr ptr\_medico)

{

return ptr\_medico->tarifa;

}

int getLibresMedico(tMedicoPtr ptr\_medico)

{

return ptr\_medico->libres;

}

void setLibresMedico(tMedicoPtr ptr\_medico, int libres)

{

ptr\_medico->libres = libres;

}

void mostrarMedicos(const tListaMedicos& listaMedicos) {

cout << " \*\*\*\*\* LISTA DE MÉDICOS \*\*\*\*\*" << endl <<endl;

cout << "========================================" << endl;

for (int i = 0; i < listaMedicos.contador; i++) {

cout << listaMedicos.medicos[i]->codigo << " "

<< listaMedicos.medicos[i]->libres << " "

<< listaMedicos.medicos[i]->tarifa << endl;

}

cout << "========================================" << endl <<endl;

}

void eliminar(tListaMedicos& listaMedicos, tMedicoPtr ptr\_medico)

{

int pos = buscarPosMedico(listaMedicos, ptr\_medico);

delete ptr\_medico;

for (int i = pos + 1; i < listaMedicos.contador; i++)

listaMedicos.medicos[i - 1] = listaMedicos.medicos[i];

listaMedicos.contador--;

}

int buscarPosMedico(const tListaMedicos& listaMedicos, tMedicoPtr ptr\_medico)

{

int i = 0;

while (i < listaMedicos.contador && listaMedicos.medicos[i] != ptr\_medico)

i++;

return i;

}

void destruir(tListaMedicos& listaMedicos)

{

for (int i = 0; i < listaMedicos.contador; i++) {

delete listaMedicos.medicos[i];

}

listaMedicos.contador = 0;

}

**“citas.h”**

#pragma once

#include <string>

using namespace std;

typedef struct

{

int codigoMedico;

string paciente;

} tCita;

typedef struct

{

tCita\* citas;

int contador, capacidad;

}tListaCitas;

void inicializarListaCitas(tListaCitas& listaCitas);

bool leerCitas(tListaCitas& lista);

void mostrarPeticiones(const tListaCitas& listaCitas);

void ampliar(tListaCitas& lista);

void destruir(tListaCitas& listaCitas);

int getNumElems(const tListaCitas& listaCitas);

int getCodigoMedico(const tListaCitas& listaCitas, int pos);

**“citas.cpp”**

#include "citas.h"

#include <fstream>

#include <iostream>

using namespace std;

void inicializarListaCitas(tListaCitas& listaCitas)

{

listaCitas.contador = 0;

listaCitas.capacidad = 20;

listaCitas.citas = new tCita[listaCitas.capacidad];

}bool leerCitas(tListaCitas& lista) {

bool cargado = true;

ifstream entrada;

int codigo;

char espacio;

entrada.open("Citas.txt");

if (entrada.is\_open())

{

entrada >> codigo;

while (codigo != -1)

{

if (lista.contador == lista.capacidad)

ampliar(lista);

lista.citas[lista.contador].codigoMedico = codigo;

entrada.get(espacio);

getline(entrada, lista.citas[lista.contador].paciente);

lista.contador++;

entrada >> codigo;

}

entrada.close();

}

else {

cout << "Error, no se pudo abrir el archivo 'Citas.txt'" << endl;

cargado = false;

}

return cargado;

}

void mostrarPeticiones(const tListaCitas& listaCitas) {

cout << " \*\*\*\*\* LISTA DE PETICIONES DE CITAS PENDIENTES DE PROCESAR \*\*\*\*\*" << endl << endl;

for (int i = 0; i < listaCitas.contador; i++) {

cout << listaCitas.citas[i].codigoMedico << " - "

<< listaCitas.citas[i].paciente << endl;

}

cout << endl;

}

void ampliar(tListaCitas& lista)

{

tCita\* aux = new tCita[2 \* lista.capacidad];

for (int i = 0; i < lista.contador; i++)

aux[i] = lista.citas[i];

delete[] lista.citas;

lista.citas = aux;

lista.capacidad \*= 2;

aux = nullptr;

}

void destruir(tListaCitas& listaCitas)

{

delete[] listaCitas.citas;

listaCitas.citas = nullptr;

listaCitas.contador = 0;

}

int getNumElems(const tListaCitas& listaCitas)

{

return listaCitas.contador;

}

int getCodigoMedico(const tListaCitas& listaCitas, int pos)

{

return listaCitas.citas[pos].codigoMedico;

}