#ifndef paquetes\_h

#define paquetes\_h

#include <string>

using namespace std;

const int MAX\_PAQUETES = 50;

struct tPaquete

{

string id;

int codigo\_postal;

bool cargado = false;

};

typedef tPaquete\* tArrayPaquetes[MAX\_PAQUETES];

struct tListaPaquetes

{

tArrayPaquetes array\_paquetes;

int cont;

};

bool cargarPaquetes(tListaPaquetes& listaPaquetes);

void mostrarPaquetes(const tListaPaquetes& listaPaquetes);

void recogerPaquete(tListaPaquetes& listaPaquetes);

void liberarPaquetes(tListaPaquetes& listaPaquetes);

int getNumPaquetes(const tListaPaquetes& listaPaquetes);

int getCodigoPostalPaquete(const tListaPaquetes& listaPaquetes, int i);

string getIdPaquete(const tListaPaquetes& listaPaquetes, int i);

void setCargado(tListaPaquetes& listaPaquetes, int pos);

#endif

#ifndef furgo\_h

#define furgo\_h

#include <string>

using namespace std;

const int MAX\_IDS = 8;

const int NUM\_FURGONETAS = 10;

struct tCargados

{

string\* array;

int cont;

};

struct tFurgoneta

{

int codigo\_postal;

tCargados cargados;

};

typedef tFurgoneta tListaFurgonetas[NUM\_FURGONETAS];

bool cargarCodigos(tListaFurgonetas listaFurgonetas);

void mostrarFurgonetas(const tListaFurgonetas listaFurgonetas);

void liberarFurgonetas(tListaFurgonetas listaFurgonetas);

int buscarFurgoneta(const tListaFurgonetas listaFurgonetas, int codigo\_postal);

int getNumCargados(const tListaFurgonetas listaFurgonetas, int pos);

void setIdPaquete(tListaFurgonetas listaFurgonetas, int pos, string id\_paquete);

#endif

//**main.cpp**

#define \_CRTDBG\_MAP\_ALLOC

#include <crtdbg.h>

#include <cstdlib>

#include "paquetes.h"

#include "furgonetas.h"

#include <iostream>

using namespace std;

void cargarPaquetes(tListaFurgonetas listaFurgonetas, tListaPaquetes& listaPaquetes);

int main()

{

tListaPaquetes lpaquetes;

tListaFurgonetas lfurgos;

if (cargarPaquetes(lpaquetes))

{

cout << " - LISTA DE PAQUETES AL COMIENZO -" << endl;

mostrarPaquetes(lpaquetes);

cout << "----------------------------------" << endl;

if (cargarCodigos(lfurgos))

{

cout << " - LISTA DE FURGONETAS AL COMIENZO -" << endl;

mostrarFurgonetas(lfurgos);

cout << "------------------------------------" << endl;

cargarPaquetes(lfurgos, lpaquetes);

cout << "- LISTA DE PAQUETES TRAS LA CARGA EN FURGONETAS -" << endl;

mostrarPaquetes(lpaquetes);

cout << "-------------------------------------------------" << endl;

cout << "- LISTA DE FURGONETAS TRAS LA CARGA DE PAQUETES -" << endl;

mostrarFurgonetas(lfurgos);

cout << "-------------------------------------------------" << endl;

recogerPaquete(lpaquetes);

cout << "- LISTA DE PAQUETES TRAS LA RECOGIDA EN MANO -" << endl;

mostrarPaquetes(lpaquetes);

cout << "----------------------------------------------" << endl;

liberarFurgonetas(lfurgos);

}

liberarPaquetes(lpaquetes);

}

\_CrtDumpMemoryLeaks(); // detección de fugas de memoria

return 0;

}

void cargarPaquetes(tListaFurgonetas listaFurgonetas, tListaPaquetes& listaPaquetes)

{

string id\_paquete;

int codigo\_postal;

int num\_paquetes = getNumPaquetes(listaPaquetes);

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

{

codigo\_postal = getCodigoPostalPaquete(listaPaquetes, i);

int pos = buscarFurgoneta(listaFurgonetas, codigo\_postal);

if (pos != -1 && getNumCargados(listaFurgonetas, pos) < MAX\_IDS)

{

setCargado(listaPaquetes, i);

id\_paquete= getIdPaquete(listaPaquetes, i);

setIdPaquete(listaFurgonetas, pos, id\_paquete);

listaFurgonetas[pos].cargados.cont++;

}

}

}

//**paquetes.cpp**

#include "paquetes.h"

#include <fstream>

#include <iostream>

using namespace std;

void mostrarPaquete(const tPaquete& paquete);

int buscarPaquete(const tListaPaquetes& listaPaquetes, string id);

void eliminarPaquete(tListaPaquetes& listaPaquetes, int posicion);

bool cargarPaquetes(tListaPaquetes& listaPaquetes)

{

ifstream fich;

bool ok = true;

tPaquete paq;

fich.open("paquetes.txt");

if (!fich.is\_open()) ok = false;

else

{

fich >> listaPaquetes.cont;

for (int i = 0; i < listaPaquetes.cont; i++) {

fich >> paq.id >> paq.codigo\_postal;

listaPaquetes.array\_paquetes[i] = new tPaquete(paq);

}

fich.close();

}

return ok;

}

void mostrarPaquetes(const tListaPaquetes& listaPaquetes)

{

for (int i = 0; i < listaPaquetes.cont; i++)

mostrarPaquete(\*listaPaquetes.array\_paquetes[i]);

}

void mostrarPaquete(const tPaquete& paquete)

{

cout << paquete.id << " va a " << paquete.codigo\_postal << " - cargado: ";

if (paquete.cargado) cout << "SI" << endl;

else cout << "NO" << endl;

}

void recogerPaquete(tListaPaquetes& listaPaquetes)

{

string idpaq;

int posicionPaq;

cout << "Introduzca el identificador del paquete a recoger: ";

cin >> idpaq;

posicionPaq = buscarPaquete(listaPaquetes, idpaq);

if (posicionPaq != -1 && !listaPaquetes.array\_paquetes[posicionPaq]->cargado)

eliminarPaquete(listaPaquetes, posicionPaq);

}

int buscarPaquete(const tListaPaquetes& listaPaquetes, string id)

{

bool encontrado = false;

int pos = 0;

while (pos < listaPaquetes.cont && !encontrado) {

encontrado = listaPaquetes.array\_paquetes[pos]->id == id;

if (!encontrado) pos++;

}

if (!encontrado) pos = -1;

return pos;

}

void eliminarPaquete(tListaPaquetes& listaPaquetes, int posicion)

{

delete listaPaquetes.array\_paquetes[posicion];

for (int i = posicion; i < listaPaquetes.cont - 1; i++)

listaPaquetes.array\_paquetes[i] = listaPaquetes.array\_paquetes[i + 1];

listaPaquetes.cont--;

}

void liberarPaquetes(tListaPaquetes& listaPaquetes)

{

for (int i = 0; i < listaPaquetes.cont; i++)

{

delete listaPaquetes.array\_paquetes[i];

listaPaquetes.array\_paquetes[i] = nullptr;

}

listaPaquetes.cont = 0;

}

int getNumPaquetes(const tListaPaquetes& listaPaquetes)

{

return listaPaquetes.cont;

}

int getCodigoPostalPaquete(const tListaPaquetes& listaPaquetes, int i)

{

return listaPaquetes.array\_paquetes[i]->codigo\_postal;

}

string getIdPaquete(const tListaPaquetes& listaPaquetes, int i)

{

return listaPaquetes.array\_paquetes[i]->id;

}

void setCargado(tListaPaquetes& listaPaquetes, int i)

{

listaPaquetes.array\_paquetes[i]->cargado = true;

}

//**furgonetas.cpp**

#include "furgonetas.h"

#include <fstream>

#include <iostream>

using namespace std;

int buscarFurgoneta(const tListaFurgonetas listaFurgonetas, int codigo\_postal);

int buscarFurgonetaAux(const tListaFurgonetas listaFurgonetas, int codigo\_postal,

int ini, int fin);

bool cargarCodigos(tListaFurgonetas listaFurgonetas) //Es array estático (no lista)

{

ifstream fich;

bool ok = true;

fich.open("codigos.txt");

if (!fich.is\_open())

ok = false;

else

{

for (int i = 0; i < NUM\_FURGONETAS; i++)

{

fich >> listaFurgonetas[i].codigo\_postal;

listaFurgonetas[i].cargados.array = new string[MAX\_IDS]; //inic array din

listaFurgonetas[i].cargados.cont = 0;

}

fich.close();

}

return ok;

}

void mostrarFurgonetas(const tListaFurgonetas listaFurgonetas)

{

for (int f = 0; f < NUM\_FURGONETAS; f++)

{

cout << "Furgoneta " << f + 1 << " reparte en " << listaFurgonetas[f].codigo\_postal;

if (listaFurgonetas[f].cargados.cont == 0)

cout << " - Sin paquetes asignados" << endl;

else

{

cout << " - Paquetes asignados: ";

for (int c = 0; c < listaFurgonetas[f].cargados.cont; c++)

cout << listaFurgonetas[f].cargados.array[c] << " ";

cout << endl;

}

}

}

void cargarPaquetes(tListaFurgonetas listaFurgonetas, tListaPaquetes& listaPaquetes)

{

int codigo\_postal;

int num\_paquetes = getNumPaquetes(listaPaquetes);

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

{

codigo\_postal = getCodigoPostalPaquete(listaPaquetes, i);

int pos = buscarFurgoneta(listaFurgonetas, codigo\_postal);

if (pos != -1 && listaFurgonetas[pos].cargados.cont < MAX\_IDS)

{

setCargado(listaPaquetes, i);

listaFurgonetas[pos].cargados.array[listaFurgonetas[pos].cargados.cont] =

getIdPaquete(listaPaquetes, i);

listaFurgonetas[pos].cargados.cont++;

}

}

}

int buscarFurgoneta(const tListaFurgonetas listaFurgonetas, int codigo\_postal)

{

return buscarFurgonetaAux(listaFurgonetas, codigo\_postal, 0, NUM\_FURGONETAS - 1);

}

int buscarFurgonetaAux(const tListaFurgonetas listaFurgonetas, int codigo\_postal,

int ini, int fin)

{

int pos = -1;

if (ini <= fin)

{

int mitad = (ini + fin) / 2;

if (codigo\_postal == listaFurgonetas[mitad].codigo\_postal)

pos = mitad;

else if (codigo\_postal < listaFurgonetas[mitad].codigo\_postal)

pos = buscarFurgonetaAux(listaFurgonetas, codigo\_postal, ini, mitad - 1);

else

pos = buscarFurgonetaAux(listaFurgonetas, codigo\_postal, mitad + 1, fin);

}

return pos;

}

void liberarFurgonetas(tListaFurgonetas listaFurgonetas) {

for (int i = 0; i < NUM\_FURGONETAS; i++) {

delete []listaFurgonetas[i].cargados.array;

listaFurgonetas[i].cargados.array = nullptr;

listaFurgonetas[i].cargados.cont = 0;

}

}