**“jugadoras.h”**

#ifndef JUGADORAS\_H\_

#define JUGADORAS\_H\_

#include <string>

#include <iostream>

#include <fstream>

using namespace std;

const int MAX\_JUG = 100;

typedef struct tJugadora

{

int id;

string nombre;

string apellido;

int goles = 0;

};

typedef tJugadora\* tJugadoraPtr;

typedef tJugadoraPtr tArrayJug[MAX\_JUG];

typedef struct tJugadoras

{

tArrayJug array\_jugadoras;

int cont = 0;

} ;

void cargarJugadoras(ifstream& archivo, tJugadoras& lista\_jug);

void mostrarJugadoras(const tJugadoras& lista\_jug);

bool buscarJugadora(int id, const tJugadoras& lista\_jug, tJugadora &jug);

string getApellido(const tJugadora& jug);

void liberar\_memoria(tJugadoras& lista\_jugadoras);

#endif

**“liga.h”**

#ifndef LIGA\_H\_

#define LIGA\_H\_

#include <string>

#include<fstream>

#include "jugadoras.h"

using namespace std;

typedef int\* tArrayJugEquipo; //array dinámico de identificadores de jugadoras

typedef struct tEquipo

{

string nombre;

int puntos = 0;

int presupuesto;

int num\_jugadoras;

tArrayJugEquipo jugadoras;

};

typedef struct tLiga

{

int num\_equipos=0;

tEquipo equipos[10];

};

void cargarEquipos(ifstream& archivo, tLiga& liga);

void mostrarEquipos(const tLiga& liga,const tJugadoras& jugadoras);

tEquipo campeonLiga(ifstream& archivo, tLiga& liga);

void aumentarPresupuesto(tLiga& liga, string nombre);

bool ficharNuevaJugadora(tJugadoras& lista\_jugadoras, tLiga& liga, string equipo,

int id\_jug, string nombre\_jug, string apellido\_jug, int goles\_jug);

void descensoEquipo(string eq, tLiga& liga, tJugadoras& jugadoras);

string getNombre(const tEquipo& equipo);

int getPuntos(const tEquipo& equipo);

void liberar\_memoria(tLiga& liga);

#endif

**“main.cpp”**

#include <iostream>

#include <fstream>

#include <string>

#include "jugadoras.h"

#include "liga.h"

using namespace std;

void fichajes(ifstream& archivo, tJugadoras& lista\_jugadoras, tLiga liga);

int main()

{

tJugadoras lista\_jugadoras;

tLiga liga;

ifstream archivo;

tEquipo equipo\_campeon;

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

archivo.open("datos.txt");

bool abierto = archivo.is\_open();

if (abierto)

{

cargarJugadoras(archivo, lista\_jugadoras);

mostrarJugadoras(lista\_jugadoras);

cargarEquipos(archivo, liga);

mostrarEquipos(liga, lista\_jugadoras);

//Equipo campeón

equipo\_campeon = campeonLiga(archivo, liga);

cout << "El campeón de la liga es: " << getNombre(equipo\_campeon)

<< " con " << getPuntos(equipo\_campeon) << " puntos" <<

endl <<endl;

//Aumentar presupuesto

string equipo;

archivo >> equipo;

aumentarPresupuesto(liga, equipo);

mostrarEquipos(liga, lista\_jugadoras);

//Fichajes

fichajes(archivo, lista\_jugadoras, liga);

mostrarJugadoras(lista\_jugadoras);

mostrarEquipos(liga, lista\_jugadoras);

//Descenso equipo

string descenso;

archivo >> descenso;

descensoEquipo(descenso, liga, lista\_jugadoras);

cout << "Desciende el equipo " << descenso << endl << endl;

mostrarJugadoras(lista\_jugadoras);

mostrarEquipos(liga, lista\_jugadoras);

cout << "Se libera memoria dinámica" << endl;

liberar\_memoria(liga);

liberar\_memoria(lista\_jugadoras);

archivo.close();

}

else

cout << "Archivo no abierto" << endl;

return 0;

}

void fichajes(ifstream& archivo, tJugadoras& lista\_jugadoras, tLiga liga)

{

int num\_fichajes;

int id\_jug, goles\_jug;

string equipo, nombre\_jug, apellido\_jug;

tJugadoraPtr jug;

archivo >> num\_fichajes;

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

{

archivo >> equipo;

archivo >> id\_jug >> nombre\_jug >> apellido\_jug >> goles\_jug;

bool fichada = ficharNuevaJugadora(lista\_jugadoras, liga, equipo,

id\_jug, nombre\_jug, apellido\_jug, goles\_jug);

if (!fichada)

cout << equipo << " no puede fichar más jugadoras" << endl;

}

cout << endl;

}

**“liga.cpp”**

#include "liga.h"

#include "jugadoras.h"

void cargarEquipo(ifstream& archivo, tEquipo& equipo);

void mostrarEquiposAux(const tLiga& liga, const tJugadoras &jugadoras, int num);

void mostrarEquipo(const tEquipo& equipo, const tJugadoras &jugadoras);

int sumarPuntos(string eq, tLiga& liga, int puntos);

int posicionEquipo(tLiga& liga, string nombre);

void actualizarCampeon(string eq, string& campeon, int& max\_puntos, int p);

void cargarEquipos(ifstream& archivo, tLiga& liga)

{

int num\_equipos;

archivo >> num\_equipos;

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

{

cargarEquipo(archivo, liga.equipos[i]);

liga.num\_equipos++;

}

}

void cargarEquipo(ifstream& archivo, tEquipo& equipo)

{

archivo >> equipo.nombre;

archivo >> equipo.presupuesto;

archivo >> equipo.num\_jugadoras;

equipo.jugadoras = new int[equipo.presupuesto \* 3 / 1000];

for (int i = 0; i < equipo.num\_jugadoras; i++)

archivo >> equipo.jugadoras[i];

}

void mostrarEquipos(const tLiga& liga, const tJugadoras &jugadoras)

{

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

cout << "EQUIPOS" << endl;

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

mostrarEquiposAux(liga, jugadoras, 0);

cout << endl;

}

void mostrarEquiposAux(const tLiga& liga, const tJugadoras & jugadoras, int pos)

{

if (pos < liga.num\_equipos)

{

mostrarEquipo(liga.equipos[pos], jugadoras);

mostrarEquiposAux(liga, jugadoras, pos + 1);

}

}

void mostrarEquipo(const tEquipo& equipo, const tJugadoras &jugadoras)

{

tJugadora jug;

cout << "Nombre: " << equipo.nombre << ","

<< " Presupuesto: " << equipo.presupuesto << ","

<< " Puntos: " << equipo.puntos << ","

<< " Jugadoras: " << equipo.num\_jugadoras << endl;

if (equipo.num\_jugadoras > 0)

cout << "Plantilla: ";

for (int i = 0; i < equipo.num\_jugadoras; i++)

if (buscarJugadora(equipo.jugadoras[i], jugadoras, jug))

cout << getApellido(jug) << ", ";

cout << endl << endl;

}

tEquipo campeonLiga(ifstream& archivo, tLiga& liga)

{

string eq1, eq2;

int pt1, pt2, p;

int num\_enfrentamientos;

int max\_puntos = 0;

string campeon = "";

archivo >> num\_enfrentamientos;

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

{

archivo >> eq1 >> pt1 >> eq2 >> pt2;

if (pt1 == pt2)

{

p = sumarPuntos(eq1, liga, 1);

actualizarCampeon(eq1, campeon, max\_puntos, p);

p = sumarPuntos(eq2, liga, 1);

actualizarCampeon(eq2, campeon, max\_puntos, p);

}

else if (pt1 > pt2)

{

p = sumarPuntos(eq1, liga, 3);

actualizarCampeon(eq1, campeon, max\_puntos, p);

}

else

{

p = sumarPuntos(eq2, liga, 3);

actualizarCampeon(eq2, campeon, max\_puntos, p);

}

}

int pos = posicionEquipo(liga, campeon);

return liga.equipos[pos];

}

int sumarPuntos(string eq, tLiga& liga, int puntos)

{

int pos = posicionEquipo(liga, eq);

liga.equipos[pos].puntos += puntos;

return liga.equipos[pos].puntos;

}

int posicionEquipo(tLiga& liga, string nombre)

{

int pos = -1;

bool encontrado = false;

int i = 0;

while (i < liga.num\_equipos && liga.equipos[i].nombre != nombre)

i++;

if (i < liga.num\_equipos)

pos = i;

return pos;

}

void actualizarCampeon(string eq, string &campeon, int &max\_puntos, int p)

{

if (p > max\_puntos)

{

max\_puntos = p;

campeon = eq;

}

}

void aumentarPresupuesto(tLiga& liga, string id)

{

int pos = posicionEquipo(liga, id);

if (pos != -1)

{

liga.equipos[pos].presupuesto += 1000;

int\* aux = new int[(liga.equipos[pos].presupuesto \* 3 / 1000)];

for (int i = 0; i < liga.equipos[pos].num\_jugadoras; i++)

aux[i] = liga.equipos[pos].jugadoras[i];

delete[] liga.equipos[pos].jugadoras;

liga.equipos[pos].jugadoras = aux;

cout << "Se aumenta el presupuesto de " << id << endl;

}

}

bool ficharNuevaJugadora(tJugadoras& lista\_jugadoras, tLiga& liga, string equipo,

int id\_jug, string nombre\_jug, string apellido\_jug, int goles\_jug)

{

bool insertada = false;

int pos = posicionEquipo(liga, equipo);

if (pos != -1 && liga.equipos[pos].num\_jugadoras <

liga.equipos[pos].presupuesto \* 3 / 1000)

{

insertada = true;

liga.equipos[pos].jugadoras[liga.equipos[pos].num\_jugadoras] = id\_jug;

liga.equipos[pos].num\_jugadoras++;

tJugadoraPtr jug;

crearJugadora(jug, id\_jug, nombre\_jug, apellido\_jug, goles\_jug);

insertarJugadora(lista\_jugadoras, jug);

}

return insertada;

}

void descensoEquipo(string eq, tLiga& liga, tJugadoras& jugadoras)

{

int pos = posicionEquipo(liga, eq);

if (pos != -1)

{

tEquipo equipo = liga.equipos[pos];

for (int j = pos; j < liga.num\_equipos; j++)

liga.equipos[j] = liga.equipos[j + 1];

liga.num\_equipos--;

for (int i = 0; i < equipo.num\_jugadoras; i++)

eliminarJugadora(jugadoras, equipo.jugadoras[i]);

delete[] equipo.jugadoras;

equipo.jugadoras = nullptr;

}

}

string getNombre(const tEquipo& equipo)

{

return equipo.nombre;

}

int getPuntos(const tEquipo& equipo)

{

return equipo.puntos;

}

void liberar\_memoria(tLiga& liga)

{

for (int i = 0; i < liga.num\_equipos; i++)

{

delete[] liga.equipos[i].jugadoras;

liga.equipos[i].jugadoras = nullptr;

}

}

**“jugadoras.cpp”**

#include "jugadoras.h"

#include <fstream>

using namespace std;

void cargarJugadora(ifstream& archivo, tJugadora& lista\_jug);

void mostrarJugadoras\_aux(const tJugadoras& lista, int pos);

void mostrarJugadora(const tJugadora& jug);

void cargarJugadoras(ifstream& archivo, tJugadoras& lista\_jug)

{

int num;

archivo >> num;

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

{

tJugadora\* ptr = new tJugadora;

cargarJugadora(archivo, \*ptr);

lista\_jug.array\_jugadoras[lista\_jug.cont] = ptr;

lista\_jug.cont++;

}

}

void cargarJugadora(ifstream & archivo, tJugadora& jug)

{

archivo >> jug.id >> jug.nombre >> jug.apellido >> jug.goles;

}

void mostrarJugadoras(const tJugadoras& lista\_jug)

{

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

cout <<"JUGADORAS"<<endl;

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

mostrarJugadoras\_aux(lista\_jug,0);

}

void mostrarJugadoras\_aux(const tJugadoras& lista\_jug, int pos)

{

if (pos < lista\_jug.cont)

{

mostrarJugadora(\*lista\_jug.array\_jugadoras[pos]);

cout << endl;

mostrarJugadoras\_aux(lista\_jug, pos + 1);

}

}

void mostrarJugadora(const tJugadora& jug)

{

cout << "Nombre: " << jug.nombre << " " << jug.apellido << ", Goles: " << jug.goles;

}

bool buscarJugadora(int id, const tJugadoras& lista\_jug, tJugadora& jug)

{

int i = 0;

bool encont;

while (i < lista\_jug.cont && lista\_jug.array\_jugadoras[i]->id != id)

i++;

encont = i < lista\_jug.cont;

if (encont)

jug = \*lista\_jug.array\_jugadoras[i];

return encont;

}

string getApellido(const tJugadora& jug)

{

return jug.apellido;

}

void liberar\_memoria(tJugadoras& lista\_jugadoras)

{

for (int i = 0; i < lista\_jugadoras.cont; i++)

{

delete lista\_jugadoras.array\_jugadoras[i];

lista\_jugadoras.array\_jugadoras[i] = nullptr;

}

}