#ifndef \_\_LISTAEQUIPOS\_H\_\_

#define \_\_LISTAEQUIPOS\_H\_\_

#include "Equipo.h"

#ifndef NULL

#define NULL 0

#endif

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Definiciones de Tipos de Datos \*/

/\*--------------------------------\*/

/\* tipo enumerado para realizar comparaciones \*/

#ifndef \_\_RESULTADOCOMPARACION\_

#define \_\_RESULTADOCOMPARACION\_

enum ResultadoComparacion {

MAYOR,

IGUAL,

MENOR

};

#endif

/\* Tipo de Informacion que esta contenida en los Nodos de la

Lista, identificada como equipo. \*/

typedef Equipo equipo;

/\* Tipo de Estructura de los Nodos de la Lista. \*/

struct NodoLista {

equipo dato; // dato almacenado

NodoLista\* sgte; // puntero al siguienteEquipo

};

/\* Tipo de Puntero a los Nodos de la Lista, el cual se usa para recorrer

la Lista y acceder a sus Datos. \*/

typedef NodoLista\* PtrNodoListaEquipo;

/\* Tipo de Estructura de la Lista \*/

struct ListaEquipo{

PtrNodoListaEquipo primero; // puntero al primer nodo de la lista

};

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Definicion de Primitivas \*/

/\*--------------------------\*/

/\*

pre : la lista no debe haber sido creada.

post: lista queda creada y preparada para ser usada.

lista : estructura de datos a ser creado.

\*/

void crearListaEquipo(ListaEquipo &lista);

/\*----------------------------------------------------------------------------\*/

/\*

pre : lista Creada con crearLista().

post: Devuelve true si lista esta vacia, sino devuelve false.

lista : lista sobre la cual se invoca la primitiva.

\*/

bool listaVacia(ListaEquipo &lista);

/\*----------------------------------------------------------------------------\*/

/\*

pre : lista Creada con crearLista().

post: devuelve la representacion de lo Siguiente al último Nodo de la lista,

o sea el valor Null, que en esta implementacion representa el final de

la lista.

return representación del fin de la lista.

\*/

PtrNodoListaEquipo finEquipo();

/\*----------------------------------------------------------------------------\*/

/\*

pre : lista Creada con crearLista().

post: devuelve el puntero al primer elemento de la lista, o devuelve fin() si

esta vacia

lista : lista sobre la cual se invoca la primitiva.

return puntero al primer nodo.

\*/

PtrNodoListaEquipo primeroEquipo(ListaEquipo &lista);

/\*----------------------------------------------------------------------------\*/

/\*

pre : lista Creada con crearLista().

post: devuelve el puntero al nodo proximo del apuntado, o devuelve fin() si

ptrNodo apuntaba a fin() o si lista esta vacia.

lista : lista sobre la cual se invoca la primitiva.

prtNodo : puntero al nodo a partir del cual se requiere el siguienteEquipo.

return puntero al nodo siguienteEquipo.

\*/

PtrNodoListaEquipo siguienteEquipo(ListaEquipo &lista, PtrNodoListaEquipo ptrNodo);

/\*----------------------------------------------------------------------------\*/

/\*

pre : lista Creada con crearLista().

ptrNodo es un puntero a un nodo de lista.

post: devuelve el puntero al nodo anterior del apuntado, o devuelve fin() si

ptrNodo apuntaba al primero o si lista esta vacia.

lista : lista sobre la cual se invoca la primitiva.

prtNodo : puntero al nodo a partir del cual se requiere el anterior.

return puntero al nodo anterior.

\*/

PtrNodoListaEquipo anterior(ListaEquipo &lista, PtrNodoListaEquipo ptrNodo);

/\*----------------------------------------------------------------------------\*/

/\*

pre : lista creada con crearLista().

post: devuelve el puntero al ultimo nodo de la lista, o devuelve fin() si

si lista esta vacia.

lista : lista sobre la cual se invoca la primitiva.

return puntero al último nodo.

\*/

PtrNodoListaEquipo ultimo(ListaEquipo &lista);

/\*----------------------------------------------------------------------------\*/

/\*

pre : lista creada con crearLista().

post: agrega un nodo nuevo al principio de la lista con el dato proporcionado

y devuelve un puntero a ese elemento.

lista : lista sobre la cual se invoca la primitiva.

dato : elemento a adicionar al principio de la lista.

return puntero al nodo adicionado.

\*/

PtrNodoListaEquipo adicionarPrincipio(ListaEquipo &lista, equipo dato);

/\*----------------------------------------------------------------------------\*/

/\*

pre : lista creada con crearLista().

post: agrega un nodo despues del apuntado por ptrNodo con el dato

proporcionado y devuelve un puntero apuntado al elemento insertado.

Si la lista esta vacía agrega un nodo al principio de esta y devuelve

un puntero al nodo insertado. Si ptrNodo apunta a fin() no inserta

nada y devuelve fin().

lista : lista sobre la cual se invoca la primitiva.

dato : elemento a adicionar.

ptrNodo : puntero al nodo después del cual se quiere adicionar el dato.

return puntero al nodo adicionado.

\*/

PtrNodoListaEquipo adicionarDespues(ListaEquipo &lista, equipo dato, PtrNodoListaEquipo ptrNodo);

/\*----------------------------------------------------------------------------\*/

/\*

pre : lista creada con crearLista().

post: agrega un nodo al final de la lista con el dato proporcionado y devuelve

un puntero al nodo insertado.

lista : lista sobre la cual se invoca la primitiva.

dato : elemento a adicionar al final de la lista.

return puntero al nodo adicionado.

\*/

PtrNodoListaEquipo adicionarFinal(ListaEquipo &lista, equipo dato);

/\*----------------------------------------------------------------------------\*/

/\*

pre : lista creada con crearLista().

post: agrega un nodo con el dato proporcionado antes del apuntado por ptrNodo

y devuelve un puntero al nodo insertado. Si la lista esta vacia no

inserta nada y devuelve fin(). Si ptrNodo apunta al primero, el nodo

insertado sera el nuevo primero.

lista : lista sobre la cual se invoca la primitiva.

dato : elemento a adicionar.

ptrNodo : puntero al nodo antes del cual se quiere adicionar el dato.

return puntero al nodo adicionado.

\*/

PtrNodoListaEquipo adicionarAntes(ListaEquipo &lista, equipo dato, PtrNodoListaEquipo ptrNodo);

/\*----------------------------------------------------------------------------\*/

/\*

pre : lista creada con crearLista(), no vacia. ptrNodo es distinto de fin().

post: coloca el dato proporcionado en el nodo apuntado por ptrNodo.

lista : lista sobre la cual se invoca la primitiva.

dato : elemento a colocar.

ptrNodo : puntero al nodo del cual se quiere colocar el dato.

\*/

void colocarDato(ListaEquipo &lista, equipo &dato, PtrNodoListaEquipo ptrNodo);

/\*----------------------------------------------------------------------------\*/

/\*

pre : lista creada con crearLista(), no vacia. ptrNodo es distinto de fin().

post: devuelve el dato del nodo apuntado por ptrNodo.

lista : lista sobre la cual se invoca la primitiva.

dato : elemento obtenido.

ptrNodo : puntero al nodo del cual se quiere obtener el dato.

\*/

void obtenerDato(ListaEquipo &lista, equipo &dato, PtrNodoListaEquipo ptrNodo);

/\*----------------------------------------------------------------------------\*/

/\*

pre : lista creada con crearLista().

post: elimina el nodo apuntado por ptrNodo. No realiza accion si la lista

esta vacia o si ptrNodo apunta a fin().

lista : lista sobre la cual se invoca la primitiva.

ptrNodo : puntero al nodo que se desea eliminar.

\*/

void eliminarNodoEquipo(ListaEquipo &lista, PtrNodoListaEquipo ptrNodo);

/\*----------------------------------------------------------------------------\*/

/\*

pre : lista creada con crearLista().

post: si la lista no esta vacia, elimina su nodo primero, sino no realiza

accion alguna.

lista : lista sobre la cual se invoca la primitiva.

\*/

void eliminarNodoPrimero(ListaEquipo &lista);

/\*----------------------------------------------------------------------------\*/

/\*

pre : lista creada con crearLista().

post: si la lista no esta vacia elimina su nodo ultimo,

sino no realiza accion.

lista : lista sobre la cual se invoca la primitiva.

\*/

void eliminarNodoUltimo(ListaEquipo &lista);

/\*----------------------------------------------------------------------------\*/

/\*

pre : lista creada con crearLista().

post: elimina todos los Nodos de la lista quedando destruida e inhabilitada

para su uso.

lista : lista sobre la cual se invoca la primitiva.

\*/

void eliminarLista(ListaEquipo &lista);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Definición de Operaciones Adicionales \*/

/\*---------------------------------------\*/

/\*

pre : lista fue creada con crearLista().

post: si el dato se encuentra en la lista, devuelve el puntero al primer nodo

que lo contiene. Si el dato no se encuentra en la lista devuelve fin().

lista : lista sobre la cual se invoca la primitiva.

dato : elemento a localizar.

return puntero al nodo localizado o fin().

\*/

PtrNodoListaEquipo localizarDato(ListaEquipo &lista , equipo dato);

/\*----------------------------------------------------------------------------\*/

/\*

pre : lista fue creada con crearLista() y cargada con datos ordenados de

menor a mayor respecto del sentido progresivo.

post: agrega a la lista el dato manteniendo el orden pero con multiples

valores iguales y devuelve un puntero al nodo insertado.

lista : lista sobre la cual se invoca la primitiva.

dato : elemento a insertar.

return puntero al nodo insertado.

\*/

PtrNodoListaEquipo insertarDato(ListaEquipo &lista, equipo dato);

/\*----------------------------------------------------------------------------\*/

/\*

pre : la lista fue creada con crearLista().

post : elimina el dato de la lista, si el mismo se encuentra.

lista : lista sobre la cual se invoca la primitiva.

dato : elemento a eliminar.

\*/

void eliminarDato(ListaEquipo &lista, equipo dato);

/\*----------------------------------------------------------------------------\*/

/\*

pre : la lista fue creada con crearLista().

post : reordena la lista.

lista : lista sobre la cual se invoca la primitiva.

\*/

void reordenar(ListaEquipo &lista);

/\*----------------------------------------------------------------------------\*/

/\*

pre : la lista fue creada con crearLista().

post : devuelve la cantidad de datos que tiene la lista.

lista : lista sobre la cual se invoca la primitiva.

\*/

int longitud(ListaEquipo &lista);

#endif // LISTAEQUIPOS\_H\_INCLUDED