#include "agente.h"

**using namespace std**;

Agente::Agente() {

}

Agente::Agente(**const** Agente&a): miNombre(a.miNombre), horarioDeAtencion(a.horarioDeAtencion), miListaDeClientes(a.miListaDeClientes) {

}

**string** Agente::getNombre() {

**return** miNombre.toString();

}

**string** Agente::getHorarioDeAtencion() {

**return** horarioDeAtencion.toString();

}

**string** Agente::getListaDeClientes() {

**return** miListaDeClientes->toString();

}

**string** Agente::toString() {

**string** toString;

toString = miNombre.toString();

toString += "\n";

toString += horarioDeAtencion.toString();

toString += "\n";

toString += miListaDeClientes->toString();

**return** toString;

}

**void** Agente::setNombre(**const** Nombre&n) {

miNombre = n;

}

**void** Agente::setHorarioDeAtencion(**const** Horario&h) {

horarioDeAtencion = h;

}

**void** Agente::setListaDeClientes(ListaCliente\*l) {

miListaDeClientes = l;

}

Agente& Agente::**operator**=(**const** Agente&a) {

miNombre = a.miNombre;

horarioDeAtencion = a.horarioDeAtencion;

miListaDeClientes = a.miListaDeClientes;

**return** \***this**;

}

**bool** Agente::**operator**==(**const** Agente&a) {

**if**(miNombre == a.miNombre and horarioDeAtencion == a.horarioDeAtencion) {

**return true**;

} **else** {

**return false**;

}

}

**bool** Agente::**operator**<(**const** Agente&a) {

**return** miNombre < a.miNombre;

}

**bool** Agente::**operator**<=(**const** Agente&a) {

**return** miNombre <= a.miNombre;

}

**bool** Agente::**operator**>(**const** Agente&a) {

**return** miNombre > a.miNombre;

}

**bool** Agente::**operator**>=(**const** Agente&a) {

**return** miNombre >= a.miNombre;

}

#include "nodoagente.h"

**using namespace std**;

NodoAgente::NodoAgente() : siguiente(**nullptr**), anterior(**nullptr**)

{

}

Agente NodoAgente::getAgente() **const**

{

**return** agente;

}

NodoAgente\* NodoAgente::getSiguiente()

{

**return** siguiente;

}

NodoAgente\* NodoAgente::getAnterior()

{

**return** anterior;

}

**string** NodoAgente::toString()

{

**return** agente.toString();

}

**void** NodoAgente::setAgente(**const** Agente&a)

{

agente = a;

}

**void** NodoAgente::setSiguiente(NodoAgente\*s)

{

siguiente = s;

}

**void** NodoAgente::setAnterior(NodoAgente\*a)

{

anterior = a;

}

#include "listaagentes.h"

**using namespace std**;

**void** ListaAgentes::Intercambiar(NodoAgente\*a, NodoAgente\*b)

{

Agente aux(a->getAgente());

a->setAgente(b->getAgente());

b->setAgente(aux);

}

ListaAgentes::ListaAgentes() : ultimoInsertado(**nullptr**), primerInsertado(**nullptr**), auxiliar1(**nullptr**), auxiliar2(**nullptr**)

{

}

**bool** ListaAgentes::isEmpty()

{

**return** ultimoInsertado == **nullptr**;

}

**void** ListaAgentes::insertar(**const** Agente&a)

{

NodoAgente\* nuevo\_nodo = **new** NodoAgente();

**if** (primerInsertado == **nullptr**) {

primerInsertado = nuevo\_nodo;

nuevo\_nodo->setAgente(a);

nuevo\_nodo->setSiguiente(ultimoInsertado);

ultimoInsertado = nuevo\_nodo;

} **else** {

nuevo\_nodo->setAgente(a);

nuevo\_nodo->setSiguiente(ultimo);

ultimo->setAnterior(nuevo\_nodo);

ultimoInsertado = nuevo\_nodo;

}

}