using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ListaCircular1

{

public class ListaCircular

{

class Nodo

{

public int info;

public Nodo ant, sig;

}

private Nodo raiz;

public ListaCircular()

{

raiz = null;

}

public void InsertarPrimero(int x)

{

Nodo nuevo = new Nodo();

nuevo.info = x;

if (raiz == null)

{

nuevo.sig = nuevo;

nuevo.ant = nuevo;

raiz = nuevo;

}

else

{

Nodo ultimo = raiz.ant;

nuevo.sig = raiz;

nuevo.ant = ultimo;

raiz.ant = nuevo;

ultimo.sig = nuevo;

raiz = nuevo;

}

}

public void InsertarUltimo(int x)

{

Nodo nuevo = new Nodo();

nuevo.info = x;

if (raiz == null)

{

nuevo.sig = nuevo;

nuevo.ant = nuevo;

raiz = nuevo;

}

else

{

Nodo ultimo = raiz.ant;

nuevo.sig = raiz;

nuevo.ant = ultimo;

raiz.ant = nuevo;

ultimo.sig = nuevo;

}

}

public bool Vacia()

{

if (raiz == null)

return true;

else

return false;

}

public void Imprimir()

{

if (!Vacia())

{

Nodo reco = raiz;

do

{

Console.Write(reco.info + "-");

reco = reco.sig;

} while (reco != raiz);

Console.WriteLine();

}

}

public int Cantidad()

{

int cant = 0;

if (!Vacia())

{

Nodo reco = raiz;

do

{

cant++;

reco = reco.sig;

} while (reco != raiz);

}

return cant;

}

public void Borrar(int pos)

{

if (pos <= Cantidad())

{

if (pos == 1)

{

if (Cantidad() == 1)

{

raiz = null;

}

else

{

Nodo ultimo = raiz.ant;

raiz = raiz.sig;

ultimo.sig = raiz;

raiz.ant = ultimo;

}

}

else

{

Nodo reco = raiz;

for (int f = 1; f <= pos - 1; f++)

reco = reco.sig;

Nodo anterior = reco.ant;

reco = reco.sig;

anterior.sig = reco;

reco.ant = anterior;

}

}

}

static void Main(string[] args)

{

ListaCircular lc=new ListaCircular();

lc.InsertarPrimero(40);

lc.InsertarPrimero(30);

lc.InsertarPrimero(20);

lc.InsertarPrimero(10);

Console.WriteLine("Paso 1: Se insertaron 4 nodos al principio");

lc.Imprimir();

Console.WriteLine("");

//Insertado de los 4 primeros nodos

lc.InsertarUltimo(100);

lc.InsertarUltimo(200);

Console.WriteLine("Paso 2: Se le agregaron 2 nodos al final");

lc.Imprimir();

Console.WriteLine("La Cantidad de nodos en total son:"+ lc.Cantidad());

Console.WriteLine("");

//Agregado de nodos en la posicion final y listado del total de nodos

Console.WriteLine("Paso 3: Se borra el nodo de la primera posición");

lc.Borrar(1);

lc.Imprimir();

Console.WriteLine("La Cantidad de nodos en total ahora son:"+ lc.Cantidad());

Console.WriteLine("");

//Eliminado de nodo en la posicion inicial y listado del total de nodos

Console.WriteLine("Paso 4: Se borra el nodo de la cuarta posición");

lc.Borrar(4);

lc.Imprimir();

Console.WriteLine("La Cantidad de nodos en total ahora son:"+ lc.Cantidad());

Console.WriteLine("");

//Eliminado de nodo en la posicion cuatro y listado del total de nodos

Console.ReadKey();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ListaGenericaDoble1

{

class ListaGenericaDoble

{

class Nodo {

public int info;

public Nodo ant,sig;

}

private Nodo raiz;

public ListaGenericaDoble ()

{

raiz=null;

}

void Insertar (int pos, int x)

{

if (pos <= Cantidad () + 1)

{

Nodo nuevo = new Nodo ();

nuevo.info = x;

if (pos == 1)

{

nuevo.sig = raiz;

if (raiz!=null)

raiz.ant=nuevo;

raiz = nuevo;

}

else

if (pos == Cantidad () + 1)

{

Nodo reco = raiz;

while (reco.sig != null)

{

reco = reco.sig;

}

reco.sig = nuevo;

nuevo.ant=reco;

nuevo.sig = null;

}

else

{

Nodo reco = raiz;

for (int f = 1 ; f <= pos - 2 ; f++)

reco = reco.sig;

Nodo siguiente = reco.sig;

reco.sig = nuevo;

nuevo.ant=reco;

nuevo.sig = siguiente;

siguiente.ant=nuevo;

}

}

}

public int Extraer (int pos)

{

if (pos <= Cantidad ())

{

int informacion;

if (pos == 1)

{

informacion = raiz.info;

raiz = raiz.sig;

if (raiz!=null)

raiz.ant=null;

}

else

{

Nodo reco;

reco = raiz;

for (int f = 1 ; f <= pos - 2 ; f++)

reco = reco.sig;

Nodo prox = reco.sig;

reco.sig = prox.sig;

Nodo siguiente=prox.sig;

if (siguiente!=null)

siguiente.ant=reco;

informacion = prox.info;

}

return informacion;

}

else

return int.MaxValue;

}

public void Borrar (int pos)

{

if (pos <= Cantidad ())

{

if (pos == 1)

{

raiz = raiz.sig;

if (raiz!=null)

raiz.ant=null;

}

else

{

Nodo reco;

reco = raiz;

for (int f = 1 ; f <= pos - 2 ; f++)

reco = reco.sig;

Nodo prox = reco.sig;

prox=prox.sig;

reco.sig = prox;

if (prox!=null)

prox.ant=reco;

}

}

}

public void Intercambiar (int pos1, int pos2)

{

if (pos1 <= Cantidad () && pos2 <= Cantidad ())

{

Nodo reco1 = raiz;

for (int f = 1 ; f < pos1 ; f++)

reco1 = reco1.sig;

Nodo reco2 = raiz;

for (int f = 1 ; f < pos2 ; f++)

reco2 = reco2.sig;

int aux = reco1.info;

reco1.info = reco2.info;

reco2.info = aux;

}

}

public int Mayor ()

{

if (!Vacia ())

{

int may = raiz.info;

Nodo reco = raiz.sig;

while (reco != null)

{

if (reco.info > may)

may = reco.info;

reco = reco.sig;

}

return may;

}

else

return int.MaxValue;

}

public int PosMayor()

{

if (!Vacia ())

{

int may = raiz.info;

int x=1;

int pos=x;

Nodo reco = raiz.sig;

while (reco != null)

{

if (reco.info > may)

{

may = reco.info;

pos=x;

}

reco = reco.sig;

x++;

}

return pos;

}

else

return int.MaxValue;

}

public int Cantidad ()

{

int cant = 0;

Nodo reco = raiz;

while (reco != null)

{

reco = reco.sig;

cant++;

}

return cant;

}

public bool Ordenada()

{

if (Cantidad() > 1)

{

Nodo reco1=raiz;

Nodo reco2=raiz.sig;

while (reco2!=null)

{

if (reco2.info < reco1.info)

{

return false;

}

reco2=reco2.sig;

reco1=reco1.sig;

}

}

return true;

}

public bool Existe(int x)

{

Nodo reco=raiz;

while (reco!=null)

{

if (reco.info==x)

return true;

reco=reco.sig;

}

return false;

}

public bool Vacia ()

{

if (raiz == null)

return true;

else

return false;

}

public void Imprimir ()

{

Nodo reco = raiz;

while (reco != null)

{

Console.Write (reco.info + "-");

reco = reco.sig;

}

Console.WriteLine();

}

static void Main(string[] args)

{

ListaGenericaDoble lg=new ListaGenericaDoble();

lg.Insertar (1, 10);

lg.Insertar (2, 20);

lg.Insertar (3, 30);

lg.Insertar (2, 15);

lg.Insertar (1, 115);

lg.Imprimir ();

Console.WriteLine("Luego de Borrar el primero");

lg.Borrar (1);

lg.Imprimir ();

Console.WriteLine("Luego de Extraer el segundo");

lg.Extraer (2);

lg.Imprimir ();

Console.WriteLine("Luego de Intercambiar el primero con el tercero");

lg.Intercambiar (1, 3);

lg.Imprimir ();

if (lg.Existe(10))

Console.WriteLine("Se encuentra el 20 en la lista");

else

Console.WriteLine("No se encuentra el 20 en la lista");

Console.WriteLine("La posición del mayor es:"+lg.PosMayor());

if (lg.Ordenada())

Console.WriteLine("La lista está ordenada de menor a mayor");

else

Console.WriteLine("La lista no está ordenada de menor a mayor");

Console.ReadKey();

}

}

}