Nodos

<https://www.youtube.com/watch?v=2Aw3Ga95UtI>

Tutoriales de programación explicada. “PEPE”

Muchacho crea clase nodo e introduce atributos privados:

Clase Nodo:

private int dato;

private Nodo siguiente;

//Es decir, cada nodo tiene un apuntador que apunta al nodo siguiente.

public int Dato

{

get { return dato;}

set { dato = value;}

}

public Nodo Siguiente

{

get { return siguiente; }

set { siguiente = value; }

}

Encapsulados los atributos con los métodos para llamarlos.

Clase Lista:

Declara 2 apuntadores tipo nodo. Uno indica cual es la cabeza de la lista (primero) y otro que indica cuál es el último (ultimo).

class Lista

{

private Nodo Primero = new Nodo();

//Ese va a ser el primer nodo de la lista y el de abajo el último.

private Nodo Ultimo = new Nodo();

public Lista()

{

Primero = null;

Ultimo = null;

//Los inicializa a ambos como null, en el método constructor.

}

public void InsertarNodo()

{

Nodo Nuevo = new Nodo();

//cada vez que el usuario quiera crear un nuevo nodo, por lo que:

//creará un objeto de la clase Nodo. Mas abajo (Nuevo.Dato) se ve como le da

//el valor al mismo.

Console.WriteLine("Ingrese el dato que contendrá el nuevo nodo:");

Nuevo.Dato = int.Parse(Console.ReadLine());

if (Primero==null)

//esto es true cuando se ingresa el primer Nodo

{

Primero = Nuevo;

Primero.Siguiente = null;

//apunta a null

Ultimo = Nuevo;

}

{

Ultimo.Siguiente = Nuevo;

//El último, es decir el que ingresamos al ppio, apuntará a este nuevo, segundo nodo.

Nuevo.Siguiente = null;

Ultimo = Nuevo;

//El último ahora, es el nuevo nodo ingresado.

}

//En esa pequeña estructura se ve que Primero y Primero.Siguiente se definen en el if y no se pisan con cada nuevo Nodo ingresado.

//El ùltimo, por el contrario si. Porque cada nodo ingresado es el ùltimo.

Console.WriteLine("\n Nodo Ingresado\n ");

}

public void DesplegarLista()

{

Nodo Actual = new Nodo();

Actual = Primero;

//O sea el primero ingresado, el que entra por el if.

if (Primero!=null)

//Si es diferente de Null es que hay nodos en la lista

{

while (Actual!=null)

//Del "otro null" por así decir, al que le apunta el último ingresado.

{

Console.WriteLine("{0}", Actual.Dato);

//Esto le imprime el valor que tenga Actual.Dato.

Actual = Actual.Siguiente;

}

}

else

{

Console.WriteLine("\n La lista se encuentra vacía \n");

}

//Si no entiendo mal Actual es un nodo auxiliar que facilita la impresión de la lista. Luego se pierde y no se lo utiliza mas.

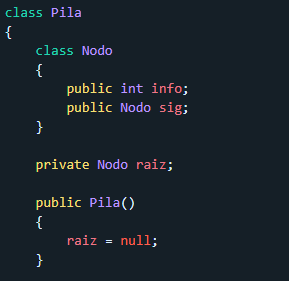
}

Hizo 2 métodos. Insertar nodo y printear.

Listas tipo Pila

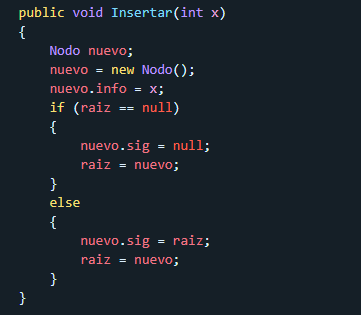
Quark

Si no me equivoco la clase Nodo es igual que la de lista:



Antes había hecho privados los 2 atributos y establecido sus setters y getters.

En este caso, a través del constructor se inicializa como null el Nodo Raiz. “PEPE” lo había inicializado en el constructor de la clase lista.



Acá se ve una diferencia en la forma en que le dan valor al Nodo o nuevo. Info, pasándoselo por parámetro, en tanto que PEPE mediante un console.readline.

Dentro del if: raíz=nuevo es igual a primero=nuevo;