Linked list source code

Robert Kelly

Contents

[Linked list source code 1](#_Toc384638248)

# Linked list source code

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LinkedLists

{

public class Node

{

/// <summary>

/// Node class this class holds objects and points to the next Node.

/// </summary>

public Node next;

public int data;

/// <summary>

/// This constructor takes a int. As the class in created it sets the data and points to null.

/// </summary>

/// <param name="i"></param>

public Node(int i)

{

data = i;

next = null;

}

/// <summary>

/// Prints out all linked items until next == null.

/// </summary>

public void print()

{

Console.Write("|"+data+"|->");

if(next !=null)

next.print();// uses the instance of "next" to call the node linked and print the data held in the node.

//Example node1 calls node2 calls node3 calls node4 and so on.

}

/// <summary>

/// Checks if the current "next" node is null if not calls the method again with the next linked node.

/// </summary>

/// <param name="data"></param>

public void AddToEnd(int data)

{

if (next == null)

next = new Node(data);

else

next.AddToEnd(data);

}

/// <summary>

/// Checks if current data is less than the next.data if yes than add before and adjust the links.

/// </summary>

/// <param name="data"></param>

public void AddSorted(int data)

{

if (next == null)

next = new Node(data);

else if (data < next.data)

{

Node temp = new Node(data);

temp.next = this.next;

this.next = temp;

}

else

next.AddSorted(data);

}

/// <summary>

/// Checks if the data held in the next node is equal to the data to be removed.

/// If true moves the pointer of the current node to the node that the next node points to.

/// This leaves the next node with no pointer to be cleaned up by garbage collection.

/// Calling Garbage collection manually is not recomended as it can make the program unstable.

/// </summary>

/// <param name="data"></param>

public void Delete(int data)

{

if (this.next.data == data)

this.next = next.next;

else

next.Delete(data);

}

}

public class List

{

/// <summary>

/// This is the list class.

/// This class has a pram called head node this is the first node in the list

/// used as a reference point to find the other nodes in the list

/// </summary>

public Node \_headNode;

/// <summary>

/// A new list with no pointers

/// </summary>

public List()

{

\_headNode = null;

}

/// <summary>

/// Finds the end of the list and adds a node

/// </summary>

/// <param name="data"></param>

public void AddToEnd(int data)

{

if (\_headNode == null)

\_headNode = new Node(data);

else

\_headNode.AddToEnd(data);

}

/// <summary>

/// Adds new node as the head points the new head node to the previous head node.

/// </summary>

/// <param name="data"></param>

public void AddToBeginning(int data)

{

if (\_headNode == null)

\_headNode = new Node(data);

else

{

Node tempNode = new Node(data);

tempNode.next = \_headNode;

\_headNode = tempNode;

}

}

public void AddSorted(int data)

{

if (\_headNode == null)

\_headNode = new Node(data);

else if (\_headNode.data > data)

AddToBeginning(data);

else

\_headNode.AddSorted(data);

}

/// <summary>

/// Calls the delete method in the node class

/// </summary>

/// <param name="data"></param>

public void Delete(int data)

{

if (\_headNode != null)

\_headNode.Delete(data);

}

public void print()

{

if (\_headNode != null)

\_headNode.print();

}

}

class Program

{

static void Main(string[] args)

{

/// <summary>

/// This block adds items to the end of a list and prints outthe results

/// </summary>

List myList = new List();

myList.AddToEnd(350);

myList.AddToEnd(32);

myList.AddToEnd(62);

myList.AddToEnd(25);

myList.AddToEnd(1);

myList.print();

Console.Read();

/// <summary>

/// This block add two records to the beginnig of the list and prints out the results

/// </summary>

myList.AddToBeginning(696969);

myList.AddToBeginning(440);

Console.WriteLine();

myList.print();

Console.ReadLine();

/// <summary>

/// Creates a new list and adds items in a sorted manner and prints out the results

/// </summary>

List mySortedList = new List();

mySortedList.AddSorted(350);

mySortedList.AddSorted(32);

mySortedList.AddSorted(62);

mySortedList.AddSorted(25);

mySortedList.AddSorted(1);

Console.WriteLine();

mySortedList.print();

Console.ReadLine();

/// <summary>

///Deletes the selected record from the list and prints out the results.

/// </summary>

mySortedList.Delete(350);

Console.ReadLine();

mySortedList.print();

Console.ReadLine();

}

}

}