3.3 Doubly Linked Lists



This section will guide you to:

* Create a Windows Console project in Visual Studio to demonstrate the use of a doubly linked list
* Create a class, Node, that will store data for the linked list
* Create a class, DoubleLinkedList, that will implement a doubly linked list
* Create a method, runApp(), that will show the use of a doubly linked list

**Development Environment**

* Visual Studio 2019 Community Version

This guide has seven subsections, namely:

* + 1. Creating a Windows Console project in Visual Studio to demonstrate the use of a doubly linked list
    2. Creating a class, Node, that will store data for the linked list
    3. Creating a class, DoubleLinkedList, that will implement a doubly linked list
    4. Adding a method, runApp(), that will show the use of a doubly linked list
    5. Building the project
    6. Publishing and running the project
    7. Pushing the code to your GitHub repositories

**Step** **3.3.1:** Creating a Windows Console project in Visual Studio to demonstrate the use of a doubly linked list

* Open Visual Studio.
* From the top menu, select **File->New->Project.**
* In **Create A New Project** screen, select **Console app (.NET Core)** from the list of available project types and click on **Next.**
* Enter **Project Name** as **Phase1Section4.7c** and click on **Create.**
* This will create the files for a Windows Console project.

**Step 3.3.2:** Creating a class, Node, that will store data for the linked list

* In the **Solution Explorer** window, right click **Phase1Section4.7c** and choose **Add->Class.**
* Enter Class name as **Node.cs** and click **Add.**
* Enter the following code:

**using** System;

**using** System.Collections.Generic;

**using** System.Text;

**namespace** Phase1Section4.\_7c

{

**public** **class** Node

{

**private** **string** data;

**private** Node next = **null**;

**private** Node prev = **null**;

**public** **string** Data

{

**get** { **return** **this**.data; }

**set** { **this**.data = value; }

}

**public** Node Next

{

**get** { **return** **this**.next; }

**set** { **this**.next = value; }

}

**public** Node Prev

{

**get** { **return** **this**.prev; }

**set** { **this**.prev = value; }

}

}

}

**Step 3.3.3:** Creating a class, DoubleLinkedList, that will implement a doubly linked list

* In the **Solution Explorer** window, right click **Phase1Section4.7c** and choose **Add->Class.**
* Enter Class name as **DoubleLinkedList.cs** and click **Add.**
* Enter the following code:

**using** System;

**using** System.Collections.Generic;

**using** System.Text;

**namespace** Phase1Section4.\_7c

{

**public** **class** DoubleLinkedList

{

**private** Node root = **null**;

**private** Node current = **null**;

**private** **int** length;

**public** Node Root

{

**get** { **return** **this**.root; }

**set** { **this**.root = value; }

}

**public** **int** Length

{

**get** { **return** **this**.length; }

**set** { **this**.length = value; }

}

**public** **void** add(Node node)

{

node.Next = **null**;

**if** (root == **null**)

{

root = node;

length = 1;

current = root;

root.Prev = **null**;

}

**else**

{

current.Next = node;

node.Prev = current;

current = node;

length++;

}

}

**public** Node getCurrent()

{

**return** **this**.current;

}

**public** Node getRoot()

{

current = root;

**return** **this**.root;

}

**public** Node getNext()

{

**if** (current.Next != **null**)

{

current = current.Next;

**return** current;

}

**else**

**return** **null**;

}

**public** Node getPrev()

{

**if** (current.Prev != **null**)

{

current = current.Prev;

**return** current;

}

**else**

**return** **null**;

}

**public** **int** getLengthOfList()

{

**return** **this**.length;

}

}

}

**Step 3.3.4:** Adding a method, runApp(), that will show the use of a doubly linked list

* Select **Program.cs** as the current Code tab.
* Enter the following code:

**using** System;

**namespace** Phase1Section4.\_7c

{

**class** Program

{

**static** **void** Main(**string**[] args)

{

runApp();

}

**public** **static** **void** runApp()

{

DoubleLinkedList list = **new** DoubleLinkedList();

Node n = **new** Node();

n.Data = "root";

list.add(n);

**for** (**int** i = 0; i < 10; i++)

{

Node node = **new** Node();

node.Data = Convert.ToString(i);

list.add(node);

}

Console.WriteLine("length of singly linked list=" + list.getLengthOfList());

Console.WriteLine("Traversing forward..");

n = list.getRoot();

**while** (n != **null**)

{

Console.WriteLine(n.Data);

n = list.getNext();

}

Console.WriteLine("Traversing backwards..");

n = list.getPrev();

**while** (n != **null**)

{

Console.WriteLine(n.Data);

n = list.getPrev();

}

}

}

}

**Step 3.3.5:** Building the project

* From the top menu, choose **Build->Build Solution.**
* If any compile errors are shown, fix them as required.

**Step 3.3.6:** Publishing and running the project

* From the top menu, select **Debug->Start Without Debugging.**
* This will execute the program in a console window.

**Step 3.3.7:** Pushing the code to your GitHub repositories

Open your command prompt and navigate to the folder where you have created your files.

cd <folder path>

Initialize your repository using the following command:

git init

Add all the files to your git repository using the following command:

git add .

Commit the changes using the following command:

git commit -m “Changes have been committed.”

Push the files to the folder you created initially using the following command:

git push -u origin master