**Task: Write a program to create a doubly linked list. Display the linked list**

**SOLUTION:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace doublylinkedlist

{

class Program

{

internal class Dnode

{

internal int data;

internal Dnode prev;

internal Dnode next;

public Dnode(int d)

{

data = d;

prev = null;

next = null;

}

public Dnode()

{

data = 0;

prev = null;

next = null;

}

}

internal class Doublelinkedlist

{

internal Dnode head;

internal Dnode start;

internal void insertatfront(Doublelinkedlist doublelinked, int data)

{

Dnode newnode = new Dnode(data);

newnode.next = doublelinked.head;

newnode.prev = null;

if (doublelinked.head != null)

{

doublelinked.head.prev = newnode;

}

doublelinked.head = newnode;

}

internal void insertatend(Doublelinkedlist doublelinked, int data)

{

Dnode newnode = new Dnode(data);

if (doublelinked.head == null)

{

newnode.prev = null;

doublelinked.head = newnode;

return;

}

Dnode lastnode = getlastnode(doublelinked);

lastnode.next = newnode;

newnode.prev = lastnode;

}

internal Dnode getlastnode(Doublelinkedlist doublelinked)

{

Dnode temp = doublelinked.head;

while (temp.next != null)

{

temp = temp.next;

}

return temp;

}

public void InsertAfter(int data, int x)

{

Dnode newnode = new Dnode(data);

Dnode p = start;

while (p != null)

{

if (p.data == x)

break;

p = p.next;

}

if (p == null)

{

Console.WriteLine(x + "not present in the list ");

}

else

{

newnode.prev = p;

newnode.next = p.next;

if (p.next != null)

p.next.prev = newnode; //should not be done with p refers to last node

p.next = newnode;

}

}

public void display()

{

Dnode temp = head;

if (head==null)

{

Console.WriteLine("Linked List Is Empty");

}

else

{

Console.WriteLine("Nodes Of Doubly Linked List");

while (temp!=null)

{

Console.WriteLine(temp.data);

temp=temp.next;

}

}

}

}

static void Main(string[] args)

{

Doublelinkedlist list = new Doublelinkedlist();

Console.WriteLine("Insert at front");

list.insertatfront(list, 2);

list.insertatfront(list, 3);

list.insertatfront(list, 5);

list.display();

Console.WriteLine("Insert at end");

list.insertatend(list, 10);

list.insertatend(list, 18);

list.display();

Console.WriteLine("");

list.InsertAfter(100, 6);

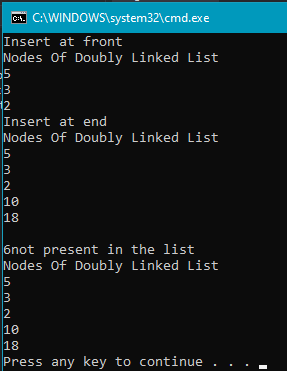
list.display();

}

}

}

**OUTPUT:**

****