using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp14

{

public class Node<T>

{

public T value;

public Node<T> next;

public Node<T> prev;

public int id;

public void SetId(int id)

{

this.id = id;

}

public static int Count = 0;

public Node(T value)

{

this.value = value;

}

public Node()

{

}

}

class MyList<T>

{

public Node<T> list;

public Node<T> head;

public Node<T> tail;

public Node<T> prev;

public int id = 0;

public void Add(Node<T> NewNode)

{

if (list != null)

{

id++;

GetLast().next = NewNode;

GetLast().SetId(id);

GetLast().prev = prev;

tail = GetLast();

prev = tail;

return;

}

list = NewNode;

list.id = id;

id++;

head = list;

tail = list;

prev = list;

}

public Node<T> GetLast()

{

Node<T> current = list;

while (current.next != null)

{

current = current.next;

}

return current;

}

public void Print()

{

Node<T> Head = head;

while (Head != null)

{

Console.WriteLine(Head.value);

Head = Head.next;

}

}

public void PrintBack()

{

Node<T> Tail = tail;

while (Tail != null)

{

Console.WriteLine(Tail.value);

Tail = Tail.prev;

}

}

public void PushFront(Node<T> newNode)

{

Node<T> node = newNode;

node.next = head;

head.prev = node;

head = node;

}

public void InsertBefore(Node<T> newNode, int id)

{

Node<T> node = newNode;

Node<T> Head = head;

Node<T> IdedElement;

for (int i = 0; i < id - 1; i++)

{

Head = Head.next;

}

IdedElement = Head.next;

Head.next = node;

node.next = IdedElement;

}

public T Get(int id)

{

Node<T> Head = head;

for (int i = 0; i < id; i++)

{

Head = Head.next;

}

return Head.value;

}

public T Remove(int id)

{

T valueToReturn;

Node<T> Head = head;

Node<T> OneAhead;

for (int i = 0; i < id; i++)

{

Head = Head.next;

}

valueToReturn = Head.value;

OneAhead = Head.next;

Head = Head.prev;

Head.next = OneAhead;

return valueToReturn;

}

public void InsertAfter(Node<T> NewNode, int id)

{

Node<T> node = NewNode;

Node<T> Head = head;

Node<T> nextElement;

for (int i = 0; i < id + 1; i++)

{

Head = Head.next;

}

nextElement = Head;

Head = Head.prev;

Head.next = node;

node.next = nextElement;

}

public T PopFront()

{

T valueToReturn;

valueToReturn = head.value;

head = head.next;

head.prev = null;

return valueToReturn;

}

public T PopBack()

{

T valueToReturn;

valueToReturn = tail.value;

tail = tail.prev;

tail.next = null;

return valueToReturn;

}

}

class Program

{

static void Main(string[] args)

{

MyList<int> spisok = new MyList<int>();

spisok.Add(new Node<int>(10));

spisok.Add(new Node<int>(28));

spisok.Add(new Node<int>(32));

spisok.Add(new Node<int>(22));

spisok.Add(new Node<int>(12));

spisok.PushFront(new Node<int>(100));

Console.WriteLine();

spisok.Print();

Console.WriteLine();

Console.WriteLine();

Console.WriteLine();

Console.WriteLine(spisok.Get(1));

Console.WriteLine();

Console.WriteLine();

spisok.Print();

Console.WriteLine();

Console.WriteLine();

spisok.InsertAfter(new Node<int>(3), 2);

Console.WriteLine(spisok.Remove(1));

Console.WriteLine();

Console.WriteLine();

spisok.Print();

Console.WriteLine();

Console.WriteLine();

Console.WriteLine(spisok.PopFront());

Console.WriteLine();

Console.WriteLine(spisok.PopBack());

Console.WriteLine();

Console.WriteLine();

spisok.Print();

Console.ReadKey();

}

}

}