МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Тихоокеанский государственный университет»

Кафедра «Программное обеспечение вычислительной техники и автоматизированных систем»

Перегрузка операторов

Лабораторная работа №4

по дисциплине «Объектно ориентированное программирование»

Выполнил студент Пшеничный Д.О.

Факультет, группа ФКФН, ПО(аб)-81

Проверил Федосеев А.А.

Хабаровск – 2020г.

Задание: На основе лабораторной работы № 3, перегрузить операторы для улучшения работы с созданной структурой данных. Согласно варианта реализовать набор перегруженных операторов.

Задание для Вариант 2:

дек с использованием динамического линейного списка. Операторы:

“<< ” - добавление нового элемента в конец;

“< ” - добавление нового элемента в начало;

“>> ” - извлечение элемента из конца;

“> ” - извлечение элемента из начала;

**Описание класса контейнера**

using System;

using System.Collections.Generic;

using System.Text;

namespace Prog

{

class ListContainer<T>

{

int Count = 0;

ListNode<T> head;

ListNode<T> tail;

public ListContainer()

{

}

public void Insert(T Value)

{

ListNode<T> newNode = new ListNode<T>(Value, this);

if (Count == 0)

{

head = newNode;

tail = newNode;

}

else

{

head.prev = newNode;

newNode.next = head;

head = newNode;

}

Count++;

}

public void Append(T Value)

{

ListNode<T> newNode = new ListNode<T>(Value, this);

if (Count == 0)

{

head = newNode;

tail = newNode;

}

else

{

tail.next = newNode;

newNode.prev = head;

tail = newNode;

}

Count++;

}

public ListNode<T> PopBack()

{

ListNode<T> popNode;

if(Count != 0)

{

if (Count == 1)

{

Count--;

popNode = head;

head = null;

tail = null;

return popNode;

}

else

{

Count--;

popNode = tail;

tail = tail.prev;

return popNode;

}

}

return null;

}

public ListNode<T> PopForward()

{

ListNode<T> popNode;

if (Count != 0)

{

if (Count == 1)

{

Count--;

popNode = head;

head = null;

tail = null;

return popNode;

}

else

{

Count--;

popNode = head;

head = head.next;

return popNode;

}

}

return null;

}

public override string ToString()

{

string res = "";

ListNode<T> iter = head;

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

{

res += iter.ToString() + "\n";

iter = iter.next;

}

return res;

}

public static ListNode<T> operator >= ( ListContainer<T> container, int node) // Выталкивание с конца

{

return container.PopBack();

}

public static ListNode<T> operator <=(ListContainer<T> container, int node)

{

return null;

}

public static ListNode<T> operator >(ListContainer<T> container, int node) // Выталкивание с начала

{

return container.PopForward();

}

public static ListNode<T> operator <(ListContainer<T> container, int node)

{

return null;

}

public static ListNode<T> operator <=(ListContainer<T> container, T node) // Вставка в конец

{

//return container.PopBack();

container.Append(node);

return null;

}

public static ListNode<T> operator>=(ListContainer<T> container, T node)

{

//return container.PopBack();

container.Append(node);

return null;

}

public static ListNode<T> operator >(ListContainer<T> container, T node) // Вставка в начало

{

container.Insert(node);

return null;

}

public static ListNode<T> operator <(ListContainer<T> container, T node)

{

container.Insert(node);

return null;

}

}

}

**Описание класса элемента**

class ListNode<T>

{

ListContainer<T> container;

public ListNode<T> prev;

public ListNode<T> next;

public T Value { get; set; }

public ListNode(T \_value, ListContainer<T> \_container)

{

Value = \_value;

container = \_container;

}

public override string ToString()

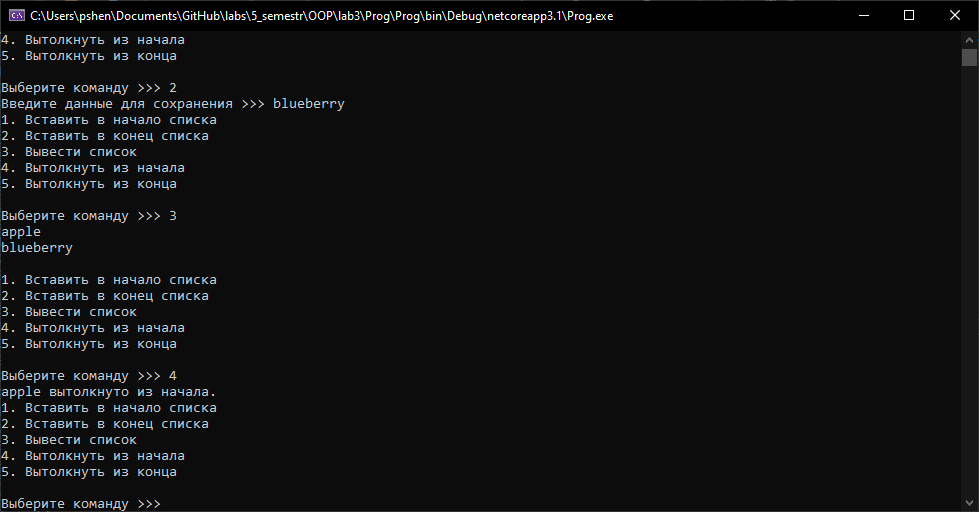
{

return Value.ToString();

}

}

**Пример работы**

****

**Код программы**

class Program

{

static void Main(string[] args)

{

int ans = 0;

ListContainer<string> list = new ListContainer<string>();

while(true)

{

Console.WriteLine("1. Вставить в начало списка\n2. Вставить в конец списка\n3. Вывести список\n4. Вытолкнуть из начала\n5. Вытолкнуть из конца\n");

Console.Write("Выберите команду >>> ");

ans = int.Parse(Console.ReadLine());

switch (ans)

{

case 1:

Console.Write("Введите данные для сохранения >>> ");

var i = list < Console.ReadLine();

break;

case 2:

Console.Write("Введите данные для сохранения >>> ");

var j = list <= Console.ReadLine();

break;

case 3:

Console.WriteLine(list.ToString());

break;

case 4:

Console.WriteLine("{0} вытолкнуто из начала.", list > 0);

break;

case 5:

Console.WriteLine("{0} вытолкнуто из конца.", list >= 0);

break;

default:

break;

}

}

}

}