МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования

«МОСКОВСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ СВЯЗИ И ИНФОРМАТИКИ»

(МТУСИ)

Кафедра «Информационная безопасность»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №9

по дисциплине

«Программирование в системах информационной безопасности»

на тему

«Классы с динамическими структурами данных»

Вариант 9

Выполнил:

студент группы БСУ1801

Копылов М.А.

Проверил:

старший преподаватель кафедры ИБ

Барков В.В.

Москва, 2021

# Цель работы

Изучить динамические структуры данных, овладеть навыками создания конструкторов копирования, перемещения, деструкторов, перегрузки операций копирования и перемещения.

# Задание

Для типа динамической структуры данных, указанного в индивидуальном задании, разработать соответствующий класс, предусмотрев в нем конструкторы инициализации, копирования, перемещения, деструктор, функции вставки и удаления элемента, просмотра доступного элемента и функцию, проверяющую наличие элементов.

Перегрузить операции присваивания, присваивание с перемещением и потокового вывода для вывода содержимого динамической структуры на экран.

Обязательные функции-члены:

|  |  |  |
| --- | --- | --- |
| Стек | Очередь | Дек |
| int GetSize() const;  bool Push(const T &element);  bool Pop(T &element);  bool Peek(T &element); | int GetSize();  bool Push(const T &element);  bool Pop(T &element);  bool Peek(T &element); | int GetSize() const;  bool PushFront(const T &element);  bool PopFront(T &element);  bool PushBack(const T &element);  bool PopBack(T &element);  bool PeekFront(T &element) const;  bool PeekBack(T &element) const; |

Функции Push, PushFront и PushBack должны возвращать true в случае успешного добавления и false в случае нехватки места.

Функции Pop, PopBack, PopFront, Peek, PeekBack, PeekFront, должны возвращать true и копию элемента через параметр element, передаваемый по ссылке, в случае если в контейнере есть элементы и false, в случае если в контейнере нет элементов.

**Индивидуальное задание**

**Вариант 9**

|  |  |  |
| --- | --- | --- |
| 9 | Очередь на основе однонаправленного циклического списка | QueueBasedOnUnidirectionalCyclicLinkedList |

# Выполнение

Листинг 1 – программный код файла Queue.cpp

#include "Header.h"

Queue::Queue()

{

next = tail = nullptr;

}

Queue::Queue(int a)

{

next = new Node();

next->next = next;

tail = next;

next->a = a;

}

Queue& Queue::operator=(Queue &&other)

{

if (this == &other)

{

return \*this;

}

delete next;

next = other.next;

other.next = nullptr;

}

int Queue::GetSize() const

{

if (!HasElements()) { return 0; }

return size;

}

bool Queue::Push(const int a)

{

Node\* newnode = new Node();

newnode->a = a;

if (HasElements())

{

newnode->next = tail;

next->next = newnode;

tail = newnode;

}

else

{

newnode->next = next = tail = newnode;

}

size++;

return true;

}

bool Queue::Pop(int &a)

{

if (!HasElements()) { return false; };

a = next->a;

Node\* temp = new Node();

temp = tail;

if (next == tail)

{

delete next;

next = tail = nullptr;

size = 0;

return true;

}

while (temp->next != next)

{

temp = temp->next;

}

size--;

next = temp;

next->next = tail;

return true;

}

bool Queue::Peek(int& a)

{

if (!HasElements()) { return false; };

a = next->a;

return true;

}

bool Queue::HasElements() const

{

if (this->next != nullptr)

{

return true;

}

else

{

return false;

}

}

Листинг 2 – программный код файла Headher.h

{

int a;

Node\* next;

};

class Queue

{

public:

Queue();

Queue(int);

Queue& operator=(Queue&&);

int GetSize() const;

bool Push(const int);

bool Pop(int&);

bool HasElements() const;

bool Peek(int&);

private:

Node\* next;

Node\* tail;

int size;

};

Листинг 3 – программный код файла main.cpp

#include "Header.h"

#include <conio.h>

#include<iostream>

using namespace std;

int main()

{

setlocale(LC\_ALL, "RUS");

Queue a;

int b;

bool flag = true;

while (flag)

{

cout << "1. Добавить элемент в очередь\n";

cout << "2. Получить элемент из очереди\n";

cout << "3. Посмотреть элемент из очереди\n";

cout << "4. Длина очереди\n";

cout << "5. Выход\n";

switch (\_getch())

{

case '1':

cout << "Добавить элемент: ";

cin >> b;

a.Push(b);

break;

case '2':

a.Pop(b) ? cout << "Полученный элемент " << b << endl : cout << "Очередь пуста\n";

break;

case '3':

a.Peek(b) ? cout << "Последний элемент " << b << endl : cout << "Очередь пуста\n";

break;

case '4':

b = a.GetSize();

cout << "Длина очереди: " << b << endl;

break;

case '5':

flag = false;

break;

}

}

}

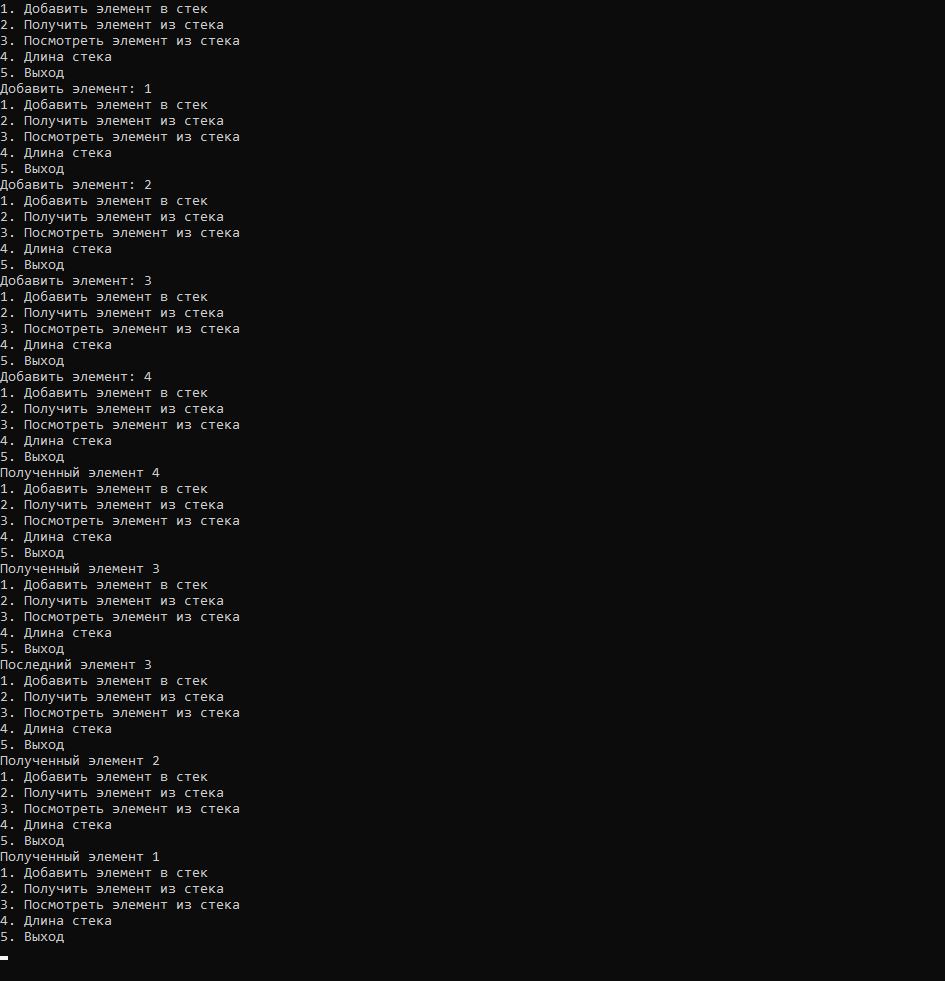


Рисунок 1 – Результат работы с программой.