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»

Кафедра «Информационная безопасность»

Лабораторная №5

Выполнила:

студентка группы БП31801

Зелик М.Б.

Вариант №10

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;  void Push(const T &element);  T Pop();  T Peek(); | int GetSize();  void Push(const T &element);  T Pop();  T Peek(); | int GetSize() const;  void PushFront(const T &element);  T PopFront();  void PushBack(const T &element);  T PopBack();  T PeekFront() const;  T PeekBack() const; |

**Индивидуальные варианты заданий**

|  |  |  |
| --- | --- | --- |
| 10 | Очередь на основе двунаправленного циклического списка | QueueBasedOnBidirectionalCyclicLinkedList |

**Выполнение**

Листинг 1 - Исходный код файла lab5.cpp

#include <iostream>

#pragma once

using namespace std;

template <typename T>

class Queue

{

public:

Queue(int size);

Queue(const Queue &other);

Queue(Queue &&other);

Queue &operator=(const Queue &other);

~Queue();

int GetSize() const;

void Push(T item);

T Pop()

{

if(front==-1)

{

cout<<"\n";

return 0;

}

else

{

cout<<"Udalenny element "<<arr[front];

if(front == back)

{

front = back = -1;

return 0;

}

else

front = front + 1;

}

}

T Peek()

{

if(front == -1)

{

cout<<"Pustaya ochered'\n";

}

else

cout<<"Pervy element "<<arr[front];

}

void show();

private:

T \*arr;

int front;

int back;

int size;

};

template <typename T>

Queue<T>::Queue(int size)

{

arr = new T[size];

front = -1;

back = -1;

}

template <typename T>

Queue<T>::Queue(const Queue &other)

{

arr = new T[other.size];

size = other.size;

for (int i = 0; i < size; ++i)

arr[i] = other.arr[i];

}

template <typename T>

Queue<T>::Queue(Queue &&other)

{

arr = other.arr;

size = other.count;

other.count = nullptr;

}

template <typename T>

Queue<T>::~Queue()

{

delete[] arr;

}

template <typename T>

void Queue<T>::Push(T item)

{

if (back >= size - 1)

{

cout<<"\n Perepolnenye";

}

else

{

if(front == -1)

{

front++;

back++;

}

else

back=back + 1;

arr[back]=item;

cout<<"\nElement dlya vvoda "<<arr[back];

}

}

template <typename T>

void Queue<T>::show()

{

if (front == -1)

{

cout<<"Pustaya ochered'\n";

}

else

{

for (int i = front; i <= back; i++)

{

cout<<arr[i]<<" ";

}

}

}

int main()

{

setlocale (0, "");

int i,c;

int size = 5;

Queue<int> D(size);

do

{

cout<<"\n1.Vvod"<<endl;

cout<<"2.Udalenie"<<endl;

cout<<"3.Pokazat' pervy element"<<endl;

cout<<"4.Pokazat' resul'tat"<<endl;

cin>>c;

switch(c)

{

case 1:

cout<<"Element dlya vvoda ";

cin>>i;

D.Push(i);

break;

case 2:

D.Pop();

break;

case 3:

D.Peek();

break;

case 4:

D.show();

break;

case 0: break;

default: cout<<endl<<"Error\n\n";

break;

}

} while (c != 0);

D.~Queue();

system("pause");

}

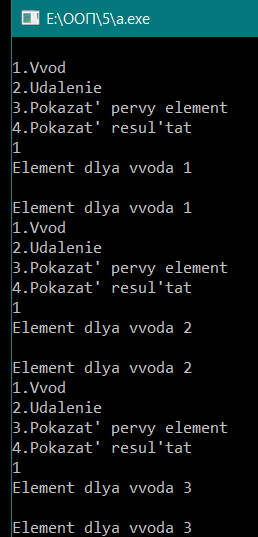


Рисунок 1 – Выполнение а.ехе

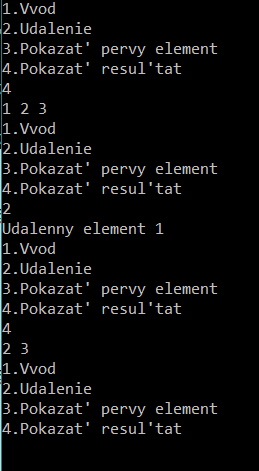


Рисунок 2 – Выполнение а.ехе

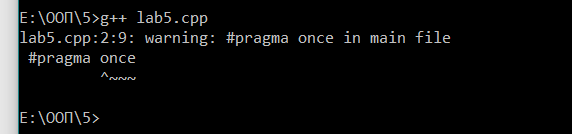


Рисунок 3 – компиляция программы