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 №10

по дисциплине

«Программирование в системах информационной безопасности»

на тему

«Исключения»

Вариант 26

Выполнил:

студент группы БСУ1801

Чигрина А.С.

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 26**

|  |  |  |
| --- | --- | --- |
| 11 | Дек на основе массива | DequeBasedOnArray |

# Выполнение

Листинг 1 – программный код файла Deque.cpp

#include "Header.h"

Deque::Deque(int count)

{

this->count = 0;

this->size = count;

front = -1;

back = -1;

arr = new int[count];

}

Deque& Deque::operator=(Deque &&other)

{

if (this == &other)

{

return \*this;

}

delete[] arr;

arr = other.arr;

count = other.count;

other.arr = nullptr;

return \*this;

}

Deque::~Deque()

{

delete[] arr;

}

int Deque::GetSize()

{

return count;

}

void Deque::pushFront(const int a)

{

if (count)

{

if ((--front) < 0)

{

front = size - 1;

}

}

else

{

front = back = 0;

}

arr[front] = a;

count++;

}

void Deque::pushBack(const int a)

{

if (front == -1)

{

front++;

back++;

}

else

back = back + 1;

arr[back] = a;

++count;

}

void Deque::popFront(int& a)

{

if (front == -1)

{

throw NoElementsException("Deck is Empty!");

}

else

{

a = arr[front];

if (front == back)

{

front = back = -1;

}

else

{

front = front + 1;

}

--count;

}

}

void Deque::popBack(int& a)

{

if (front == -1)

{

throw NoElementsException("Deck is Empty!");

}

else

{

a = arr[back];

if (front == back)

{

front = back = -1;

}

else

{

back = back - 1;

}

}

}

void Deque::peekFront(int& a)

{

if (front == -1)

{

throw NoElementsException("Deck is Empty!");

}

else

{

a = arr[front];

}

}

void Deque::peekBack(int& a)

{

if (back == -1)

{

throw NoElementsException("Deck is Empty!");

}

else

{

a = arr[back];

}

Листинг 2 – программный код файла Header.h

#include <iostream>

using namespace std;

class NoElementsException :public exception

{

public:

NoElementsException(const char\* msg) :

exception(msg) {}

};

class OverflowException :public exception

{

public:

OverflowException(const char\* msg) :

exception(msg) {}

};

class Deque

{

public:

Deque(int count);

Deque& operator=(Deque&& other);

~Deque();

int GetSize();

void pushFront(const int a);

void pushBack(const int a);

void popFront(int& a);

void popBack(int& a);

void peekFront(int& a);

void peekBack(int& a);

private:

int\* arr;

int back;

int front;

int count;

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");

int b, dsize;

bool flag = true;

cout << "Введите размер дека\n";

cin >> dsize;

Deque\* a;

a = new Deque(dsize);

while (flag)

{

cout << "1. Добавить элемент в начало дека\n";

cout << "2. Получить элемент из начала дека\n";

cout << "3. Добавить элемент в конец дека\n";

cout << "4. Получить элемент из конца дека\n";

cout << "5. Посмотреть элемент в конец дека\n";

cout << "6. Посмотреть элемент из конца дека\n";

cout << "7. Выход\n";

switch (\_getch())

{

case '1':

cout << "Добавить элемент: ";

cin >> b;

a->pushFront(b);

cout << "Элемент добавлен в начало" << endl;

break;

case '2':

try

{

a->popFront(b);

cout << "Полученный элемент " << b << endl;

}

catch (NoElementsException ex)

{

cout << "Дек пуст\n";

}

break;

case '3':

cout << "Добавить элемент: ";

cin >> b;

a->pushBack(b);

cout << "Элемент добавлен в конец " << endl;

break;

case '4':

try

{

a->popBack(b);

cout << "Полученный элемент " << b << endl;

}

catch (NoElementsException ex)

{

cout << "Дек пуст\n";

}

break;

case '5':

try

{

a->peekFront(b);

cout << "Полученный элемент " << b << endl;

}

catch (NoElementsException ex)

{

cout << "Дек пуст\n";

}

break;

case '6':

try

{

a->peekBack(b);

cout << "Полученный элемент " << b << endl;

}

catch (NoElementsException ex)

{

cout << "Дек пуст\n";

}

break;

case '7':

flag = false;

break;

}

}

}

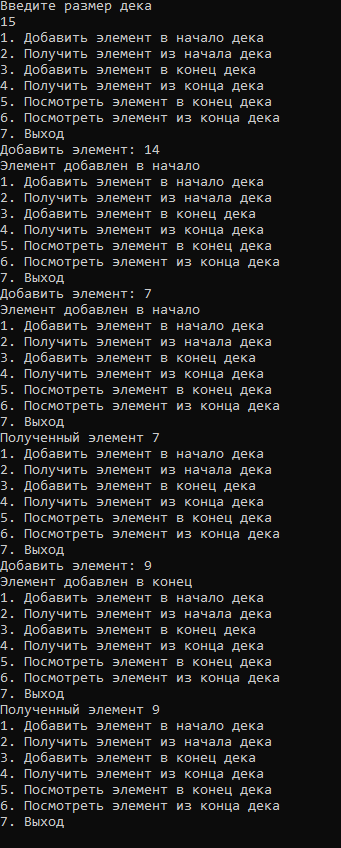


Рисунок 1 – Результат работы с программой.