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 20

Выполнил:

студент группы БСУ1801

Пышкина А.И.

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 20**

|  |  |  |
| --- | --- | --- |
| 5 | Стек на основе двунаправленного циклического списка | StackBasedOnBidirectionalCyclicLinkedList |

# Выполнение

Листинг 1 – программный код файла Stack.cpp

#include "Header.h"

Stack::Stack() {}

Stack::Stack(int a)

{

tail = new Node();

tail->prev = tail;

tail->next = tail;

tail->a = a;

count = 0;

}

Stack& Stack::operator=(Stack&& other)

{

if (this == &other)

{

return \*this;

}

delete tail;

delete next;

tail = other.tail;

other.tail = nullptr;

}

int Stack::GetSize() const

{

return count;

}

void Stack::Push(const int a)

{

Node\* newnode = new Node();

Node\* checking = next;

if (HasElements())

{

while (checking->next != tail)

{

checking = checking->next;

}

checking->next = newnode;

tail->prev = next;

}

else

{

tail = newnode;

next = newnode;

}

newnode->a = a;

newnode->prev = checking;

newnode->next = tail;

count = count + 1;

}

void Stack::Pop(int& a)

{

if (!HasElements()) { throw NoElementsException("Deck is Empty!"); };

Node\* temp = tail;

while (temp->next != tail)

{

temp = temp->next;

}

a = temp->a;

next = temp;

if (next != tail)

{

next->prev->next = tail;

}

else

{

next = tail = nullptr;

}

count = count - 1;

}

void Stack::Peek(int& a)

{

if (!HasElements()) { throw NoElementsException("Deck is Empty!"); };

Node\* temp = next;

while (temp->next != tail)

{

temp = temp->next;

}

a = temp->a;

}

bool Stack::HasElements() const

{

if (this->tail != 0)

{

return true;

}

else

{

return false;

}

}

Листинг 2 – программный код файла Header.h

#include <iostream>

using namespace std;

class NoElementsException :public exception

{

public:

NoElementsException(const char\* msg) :

exception(msg) {}

};

struct Node

{

int a;

Node\* prev;

Node\* next;

};

class Stack

{

public:

Stack();

Stack(int);

Stack& operator=(Stack&&);

int GetSize() const;

void Push(const int);

void Pop(int&);

bool HasElements() const;

void Peek(int&);

private:

Node\* tail;

Node\* next;

int count;

};

Листинг 3 – программный код файла main.cpp

#include "Header.h"

#include <conio.h>

#include<iostream>

using namespace std;

int main()

{

setlocale(LC\_ALL, "RUS");

Stack a;

int b;

bool flag = true;

while (flag)

{

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

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

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

cout << "4. Длина стека\n";

cout << "5. Выход\n";

switch (\_getch())

{

case '1':

cout << "Добавить элемент: ";

cin >> b;

a.Push(b);

break;

case '2':

try

{

a.Pop(b);

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

}

catch (NoElementsException ex)

{

cout << "Cтек пуст\n";

}

break;

case '3':

try

{

a.Peek(b);

cout << "Последний элемент " << b << endl;

}

catch (NoElementsException ex)

{

cout << "Cтек пуст\n";

}

break;

case '4':

b = a.GetSize();

cout << "Длина стека: " << b << endl;

break;

case '5':

flag = false;

break;

}

}

}

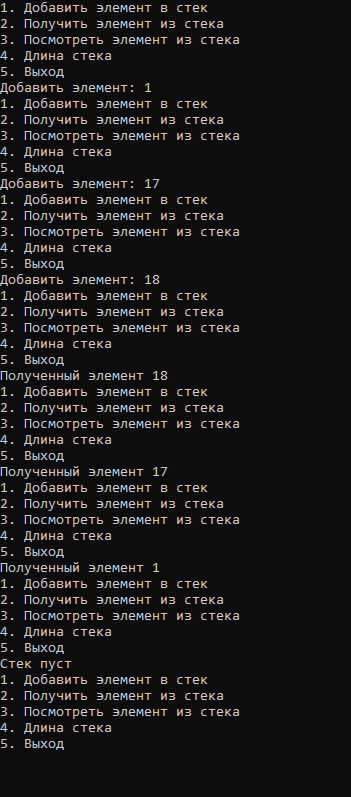


Рисунок 1 – Результат работы с программой.