МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФГАОУ ВО «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ ИНФОРМАЦИОННЫХ

ТЕХНОЛОГИЙ И ТЕЛЕКОММУНИКАЦИЙ

КАФЕДРА ПРИКЛАДНОЙ ИНФОРМАТИКИ

**Лабораторная работа №3**

по дисциплине

«Объектно-ориентированное программирование»

**Выполнил:**

Ермолаев Кирилл Александрович

Студент 2 курса группы \_ПИН-б-о-22-1

Направления подготовки

09.03.03 Прикладная информатика

очной формы обучения

Ставрополь, 2023 г.

**Тема:** Основы объектно-ориентированного программирования на ЯП С++.

**Цель работы:** Шаблоны классов. Обработка исключительных ситуаций.

**Ход работы:**

**Вариант** – 7

**Целевой шаблонный класс** – Stack

**Реализация с применением** – std:list

// main.cpp

#include <iostream>

#include "stack.h"

#include "stack.cpp"

int main() {

try {

// Использование шаблонного класса stack с типами int, double, std::string

stack<int> intStack;

intStack.push(1);

intStack.push(2);

intStack.push(3);

std::cout << "Top element of intStack: " << intStack.top() << std::endl;

intStack.pop();

stack<double> doubleStack;

doubleStack.push(1.5);

doubleStack.push(2.5);

doubleStack.push(3.5);

std::cout << "Top element of doubleStack: " << doubleStack.top() << std::endl;

doubleStack.pop();

stack<std::string> stringStack;

stringStack.push("one");

stringStack.push("two");

stringStack.push("three");

std::cout << "Top element of stringStack: " << stringStack.top() << std::endl;

stringStack.pop();

// Попытка извлечь элемент из пустого стека

stack<int> emptyStack;

emptyStack.pop();

}

catch (const std::exception& e) {

std::cerr << "Exception: " << e.what() << std::endl;

}

return 0;

}

// stack.h

#pragma once

#ifndef STACK\_H

#define STACK\_H

#include <list>

#include <stdexcept>

template <typename T>

class stack {

private:

std::list<T> data;

public:

void push(const T& value);

void pop();

T top() const;

bool empty() const;

};

#endif

// stack.cpp

#include "stack.h"

template <typename T>

void stack<T>::push(const T& value) {

data.push\_back(value);

}

template <typename T>

void stack<T>::pop() {

if (empty()) {

throw std::out\_of\_range("Stack is empty. Cannot pop.");

}

data.pop\_back();

}

template <typename T>

T stack<T>::top() const {

if (empty()) {

throw std::out\_of\_range("Stack is empty. Cannot get top element.");

}

return data.back();

}

template <typename T>

bool stack<T>::empty() const {

return data.empty();

}

// UnitTest3.cpp

#include "pch.h"

#include "CppUnitTest.h"

#include "..\\ConsoleApplication1\stack.h"

#include "..\\ConsoleApplication1\stack.cpp"

using namespace Microsoft::VisualStudio::CppUnitTestFramework;

TEST\_CLASS(StackTests)

{

public:

TEST\_METHOD(TestPushPop)

{

stack<int> testStack;

testStack.push(5);

Assert::AreEqual(5, testStack.top());

testStack.pop();

Assert::IsTrue(testStack.empty());

}

TEST\_METHOD(TestTopException)

{

stack<int> emptyStack;

try {

emptyStack.top();

Assert::Fail(L"Expected exception not thrown");

}

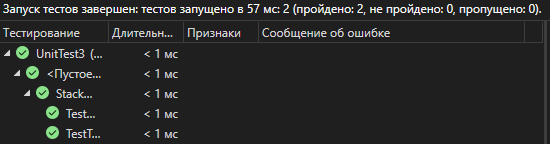
catch (const std::out\_of\_range& e) {

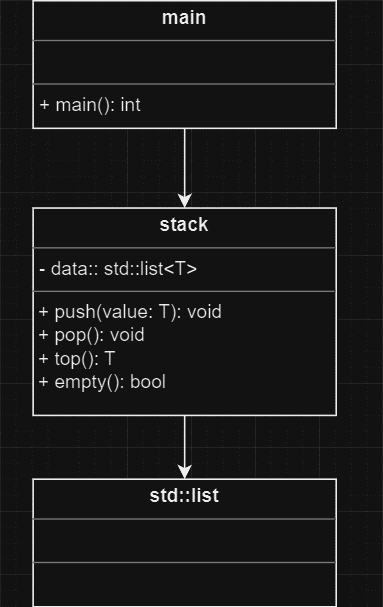
Assert::AreEqual("Stack is empty. Cannot get top element.", e.what());

}

}

};





<https://github.com/MoeTomatoki/OOP>

**Вывод:** изучил шаблоны классов, а также познакомился с обработкой исключительных ситуаций.