Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

**по лабораторной работе №18.9**

Дисциплина: «информатика»

Тема: Обработка исключительных ситуаций

Вариант 5

Выполнил работу

студент группы РИС-20-1б

Зверев А.Д.

Проверила

Доцент кафедры ИТАС

Полякова О.А.

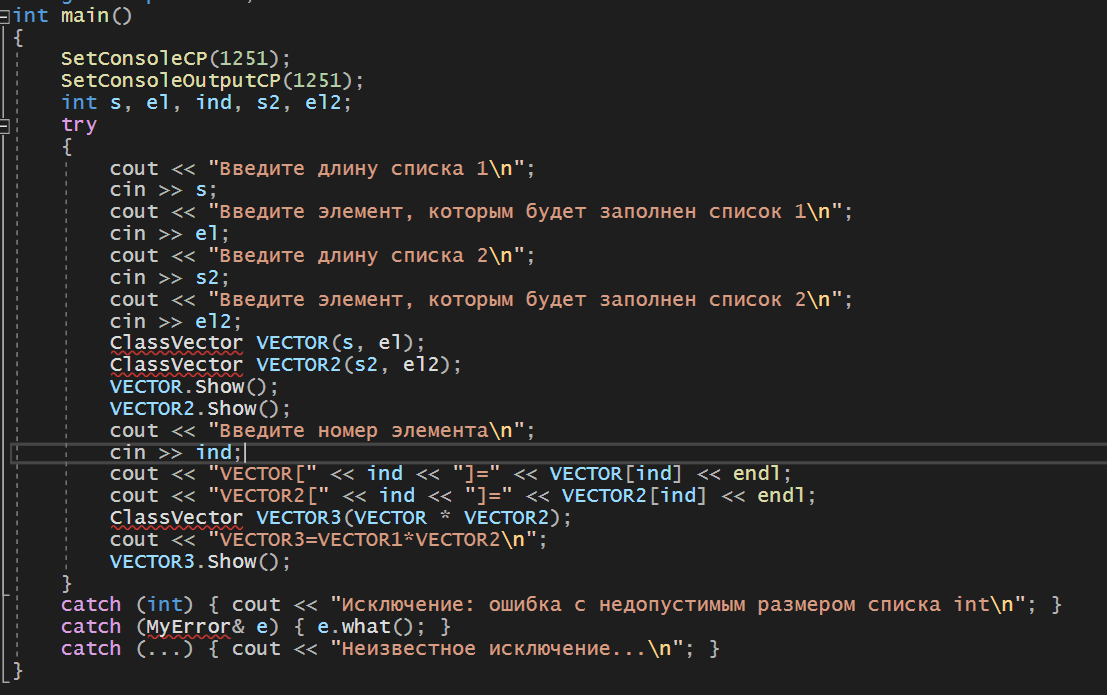
Пермь, 2021

**Постановка задачи**

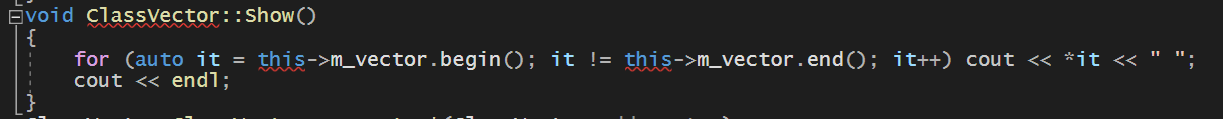
1. Реализовать класс, перегрузить для него операции, указанные в варианте.
2. Определить исключительные ситуации.
3. Предусмотреть генерации исключительных ситуаций.
4. Класс—контейнер ВЕКТОР с элементами типа int.
5. Реализовать операции:
   1. []—доступа по индексу;
   2. Int()-определение размера вектора;
   3. \* вектор—умножение элементов векторов a[i]\*b[i];
   4. + n—переход вправо к элементу с номером n.

**Анализ задачи**

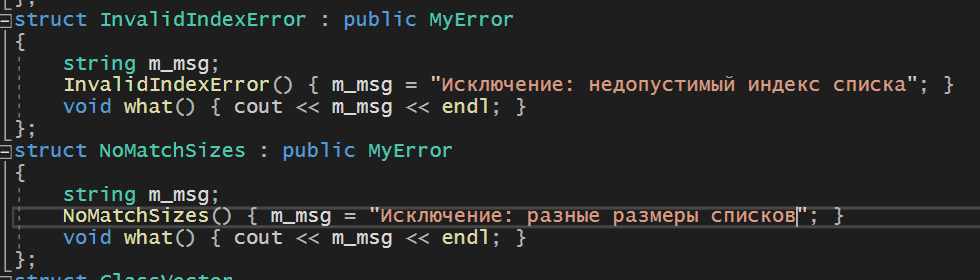
1. Для решения задачи необходимо…
   1. Написать функцию int main, в которой будут выполняться все необходимые операции;



* 1. Написать функцию ClassVector::Show типа void, которая выводит элементы вектора;



* 1. Написать несколько структур, которые отвечают за исключения в программе;



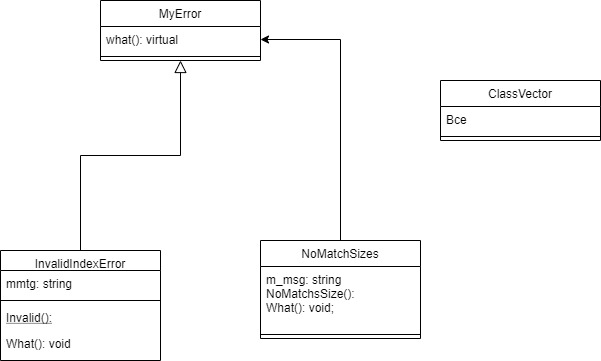
1. В программе были использованы следующие типы данных:
   1. Тип данных int для хранения различных данных;



* 1. Тип данных void для различных функций;



**Диаграмма классов**



**Решение**

**Lab18.cpp**

#include "lab18.h"

#include <string>

#include <iostream>

#include <fstream>

#include <math.h>

using namespace std;

int ClassVector::operator[](int i)

{

if (i < 0 || i >= this->m\_vector.size()) { throw InvalidIndexError(); }

auto it = this->m\_vector.begin();

advance(it, i);

return \*it;

}

void ClassVector::Show()

{

for (auto it = this->m\_vector.begin(); it != this->m\_vector.end(); it++) cout << \*it << " ";

cout << endl;

}

ClassVector ClassVector::operator\*(ClassVector add\_vector)

{

if (this->m\_vector.size() != add\_vector.m\_vector.size()) throw NoMatchSizes();

ClassVector new\_vector(this->m\_vector.size(), add\_vector[0]);

auto it1 = this->m\_vector.begin();

auto it2 = add\_vector.m\_vector.begin();

auto it3 = new\_vector.m\_vector.begin();

while (it1 != this->m\_vector.end())

{

\*it3 = (\*it2) \* (\*it1);

it1++; it2++; it3++;

}

\*it3 = (\*it2) \* (\*it1);

return new\_vector;

}

int ClassVector::operator()() { return this->m\_vector.size(); }

**main.cpp**

#include <iostream>

#include <fstream>

#include "lab18.h"

#include <string>

#include <windows.h>

#include <vector>

#include <list>

#include <queue>

#include <stack>

#include <set>

#include <map>

#include <algorithm>

#include <numeric>

using namespace std;

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int s, el, ind, s2, el2;

try

{

cout << "Введите длину списка 1\n";

cin >> s;

cout << "Введите элемент, которым будет заполнен список 1\n";

cin >> el;

cout << "Введите длину списка 2\n";

cin >> s2;

cout << "Введите элемент, которым будет заполнен список 2\n";

cin >> el2;

ClassVector VECTOR(s, el);

ClassVector VECTOR2(s2, el2);

VECTOR.Show();

VECTOR2.Show();

cout << "Введите номер элемента\n";

cin >> ind;

cout << "VECTOR[" << ind << "]=" << VECTOR[ind] << endl;

cout << "VECTOR2[" << ind << "]=" << VECTOR2[ind] << endl;

ClassVector VECTOR3(VECTOR \* VECTOR2);

cout << "VECTOR3=VECTOR1\*VECTOR2\n";

VECTOR3.Show();

}

catch (int) { cout << "Исключение: ошибка с недопустимым размером списка int\n"; }

catch (MyError& e) { e.what(); }

catch (...) { cout << "Неизвестное исключение...\n"; }

}

**Lab18.h**

#pragma once

#include <string>

#include <iostream>

#include <list>

#include <vector>

#include <fstream>

using namespace std;

//собственные классы исключений

struct MyError

{

virtual void what() = 0;

};

struct InvalidIndexError : public MyError

{

string m\_msg;

InvalidIndexError() { m\_msg = "Исключение: недопустимый индекс списка"; }

void what() { cout << m\_msg << endl; }

};

struct NoMatchSizes : public MyError

{

string m\_msg;

NoMatchSizes() { m\_msg = "Исключение: разные размеры списков"; }

void what() { cout << m\_msg << endl; }

};

struct ClassVector

{

ClassVector(int s, int el)

{

if (s <= 0) throw 1;

for (int i = 0; i < s; i++) m\_vector.push\_back(el);

}

~ClassVector() {}

vector <int> m\_vector;

int operator[](int i);

int operator()();

ClassVector operator\*(ClassVector add\_vector);

void Show();

void Change(int index, int el)

{

auto it = m\_vector.begin();

advance(it, index);

\*it = el;

}

};

**Скриншоты**

