Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЁТ**

**по лабораторной работе №18.9**

Дисциплина: «Информатика»

Тема: Обработка исключительных ситуаций

Вариант 7

Выполнил:

Студент группы ИВТ 20-2Б

Копытов А.А.

Проверил:

Доцент кафедры ИТАС

Полякова О. А.

**Пермь, 2021**

**Цель задачи**

1) Создание консольного приложения, состоящего из нескольких файлов в системе программирования Visual Studio.

2) Разработка программы, обрабатывающей исключительные ситуации.

Для достижения поставленной цели необходимо решить следующие задачи:

* Проанализировать теоретические сведения о ООП
* Абстрактные типы данных. Контейнеры
* Реализовать алгоритм поставленной задачи
* Учесть все исключительные ситуации на ввод элементов
* Создать отдельные файлы: .cpp для описания методов класса, .cpp для описания главной функции, .h для описания класса

Постановка задачи

Класс- контейнер МНОЖЕСТВО с элементами типа int. Реализовать операции:

[] – доступа по индексу;

int() – определение размера вектора;

\* – пересечение множеств;

-- - удаление элемента из множества.

Анализ задачи

1. Определить какие операции должны быть выполнены по заданию:

* Создание класса

class Vector

* Создание множества и заполнение класса
* Создание методов получения размера множества, пересечений и доступа по индексу
* Отчистка памяти

1. Для решения задачи используются переменные:

Несколько переменных целочисленного значения для ввода чисел и реализации меню

int s, in, menu = 4;

Указатель для работы с динамической памятью

1. Ввод данных осуществляется посредством функции

cin >> rub;

Вывод данных осуществляется посредством функции

cout << "Искомый элемент = " << a.give(in);

1. Создание метода класса , который ищет пересечения класса:

void Vector::cross()

{

int k = 0;

for (int i = 0; i < size - 1; i++)

{

for (int j = i + 1; j < size; j++)

{

if (data[i] == data[j])

k++;

}

if (k != 0)

cout << "\nЭлемент " << data[i] << " встречается " << k + 1 << " раз";

k = 0;

}

1. Создание методов класса для нахождения размера и доступа по индексу

int give(int index);

int givesize();

Создание контейнера

Vector::Vector(int s)

{

int k;

size = s;

data = new int[size];

cout << "Введи элементы множества" << endl;

for (int i = 0; i < size; i++)

{

cin >> data[i];

}

}

1. Создание инструмента обработки исключительных ситуаций. Для этого была использована логика try:

try

{

cout << "Искомый элемент: " << a.give(in);

}

catch (const int in)

{

cout << "Ошибка ввода";

}

}

Код

Main.cpp:

#include <iostream>

#include "MNOG.h"

using namespace std;

int main()

{

setlocale(LC\_ALL, "Russian");

int s=-1, in, menu = 4;

while (s <= 0)

{

cout << "Введи количество элементов множества: "; cin >> s;

}

Vector a(s);

while (menu == 1 || menu == 2 || menu == 3 || menu == 4 || menu == 5)

{

cout << endl << endl << "Меню: \n\t1. Получить элемент по индексу \n\t2. Получить размер множества \n\t3. Получить пересечения множества \n\t4. Удалить элемент по индексу\n\t5. Вывести множество \n\n\t0. Выход";

cout << endl << "Выбери пункт меню: "; cin >> menu;

if (menu == 1)

{

cout << "Введи индекс элемента, который хочешь получить: "; cin >> in;

try

{

cout << "Искомый элемент: " << a.give(in);

}

catch (const int in)

{

cout << "Ошибка ввода";

}

}

if (menu == 2)

cout << "Размер = " << a.givesize();

if (menu == 3)

a.cross();

if (menu == 4)

{

cout << "Введи индекс элемента: "; cin >> in;

try

{

a.del(in);

cout << "Элемент удален " << endl;

}

catch (const int e)

{

cout << "Ошибка ввода";

}

}

if (menu == 5)

{

a.print();

}

if (menu == 0)

{

a.end();

break;

}

}

}

MNOG.cpp:

#include "MNOG.h"

#include <iostream>

using namespace std;

Vector::Vector(int s)

{

int k;

size = s;

data = new int[size];

cout << "Введи элементы множества" << endl;

for (int i = 0; i < size; i++)

{

cin >> data[i];

}

}

void Vector::print()

{

cout << "Элементы множества: ";

for (int i = 0; i < size; i++)

{

cout << data[i] << " ";

}

cout << endl;

}

int Vector::give(int index)

{

if (index < 0 || index > size)

throw index;

return data[index];

}

int Vector::givesize()

{

return size;

}

void Vector::cross()

{

int k = 0;

for (int i = 0; i < size - 1; i++)

{

for (int j = i + 1; j < size; j++)

{

if (data[i] == data[j])

k++;

}

if (k != 0)

cout << "\nЭлемент " << data[i] << " встречается " << k + 1 << " раз";

k = 0;

}

}

void Vector::end()

{

delete [] data;

data = 0;

}

void Vector::del(int in)

{

if (in < 0 || in > size)

throw in;

data1 = new int[size - 1];

for (int i = 0; i < in; i++)

{

data1[i] = data[i];

}

for (int i = in + 1; i < size; i++)

{

data1[i - 1] = data[i];

}

delete[] data;

data = data1;

size--;

}

MNOG.h:

class Vector

{

public:

Vector(int s);

int give(int index);

int givesize();

void cross();

void end();

void del(int i);

void print();

private:

int size;

int\* data;

int\* data1;

};

Блок-схема:

Main.cpp:



Pair.cpp:



Pair.h:



Работа кода

