Пермский Национальный Исследовательский  
Политехнический Университет

**Лабораторная работа № 9**

Основы алгоритмизации и программирования

за 2 семестр

Вариант № 1

Выполнил:

Студент группы РИС 20-1-бз

Курганов Н.В.

20-ЭТФ-631

Проверила:

Доцент кафедры ИТАС

Полякова О.А.

2022

Обработка исключительных ситуаций

1. **Цель задания**.

1.1 Создание консольного приложения, состоящего из нескольких файлов в системе программирования VisualStudio.

1.2. Разработка программы, обрабатывающей исключительные ситуации.

1. **Задание.**

Класс – контейнер ВЕКТОР с элементами типа int

Реализовать операции:

[] – доступ по индексу;

() – определение размера вектора;

+ число – добавляет константу ко всем элементам вектора;

-n – удаляет n элементов из конца вектора.

Вариант реализации 1,2.

Вариант реализации №1

1. Создаем:

**Файл Lab9\_main.cpp**

#include "Vector.h"

#include <iostream>

using namespace std;

void print\_error(int x);

int main()

{

//контролируемый блок

try

{

Vector x(2);//вектор из двух элементов

Vector y;//пустой вектор

cout << x;//печать вектора х

cout << "Nomer?";

int i;

cin >> i;

//вывод элемента с номером i, если номер больше 2 или меньше 0, то

//генерируется исключительная ситуация

cout << x[i] << endl;

//добавление элемента в вектор, если MAX\_SIZE=2, то генерируется

//исключительная ситуация

y = x + 3;

cout << y;

//удалить один элемент из вектора

x = x - 1;

cout << x;

//удалить один элемент из вектора

x = x - 1;

cout << x;//вектор пустой

//удалить один элемент из вектора

//генерируется исключительная ситуация

x = x - 1;

}

//обработчик исключения

catch (int x)

{

cout << "ERROR!!!" << endl;

print\_error(x);

}//сообщение об ошибке

std::cin.ignore();

std::cin.get();

return 0;

}

void print\_error(int x)

{

switch (x) {

case 1: std::cout << "Given size is greater than MAX\_SIZE\n"; break;

case 2: std::cout << "Index is less than 0\n"; break;

case 3: std::cout << "Index out of range\n"; break;

case 4: std::cout << "Can't delete values\n"; break;

case 5: std::cout << "Size is less than given for delete\n"; break;

}

}

**файл Vector.cpp**

#pragma once

#include "Vector.h"

using namespace std;

Vector::Vector(int s)

{

//если текущий размер больше максимального, то генерируется исключение

if (s > MAX\_SIZE)

throw 1;

size = s;

beg = new int[s];

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

beg[i] = 0;

}

Vector::Vector(const Vector& v)

{

size = v.size;

beg = new int[size];

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

beg[i] = v.beg[i];

}

Vector::~Vector()

{

if (beg != 0)

delete[] beg;

}

Vector::Vector(int s, int\* mas)

{

//если текущий размер больше максимального, то генерируется исключение

if (s > MAX\_SIZE)

throw 1;

size = s;

beg = new int[size];

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

beg[i] = mas[i];

}

const Vector& Vector::operator=(const Vector& v)

{

if (this == &v)

return \*this;

if (beg != 0)

delete[] beg;

size = v.size;

beg = new int[size];

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

beg[i] = v.beg[i];

return \*this;

}

ostream& operator<<(ostream& out, const Vector& v)

{

if (v.size == 0) out << "Empty\n";

else

{

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

out << v.beg[i] << " ";

out << endl;

}

return out;

}

istream& operator >>(istream& in, Vector& v)

{

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

{

cout << ">";

in >> v.beg[i];

}

return in;

}

int& Vector::operator[](int i)

{

if (i < 0) throw 2;//если индекс отрицательный, то генерируется исключение

//если индекс больше размер вектора, то генерируется исключение

if (i >= size) throw 3;

return beg[i];

}

int Vector::operator()()

{

return size;

}

const Vector& Vector::operator+(int a)

{

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

beg[i] += a;

return \*this;

}

const Vector& Vector::operator-(int n)

{

if (n < 0) throw 4;

if (size < n) throw 5;

if (size == n)//если в вектор один элемент

{

size = 0;

delete[] beg;

beg = 0;

} else {

int\* temp = new int[size - n];

size -= n;

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

temp[i] = beg[i];

delete[] beg;

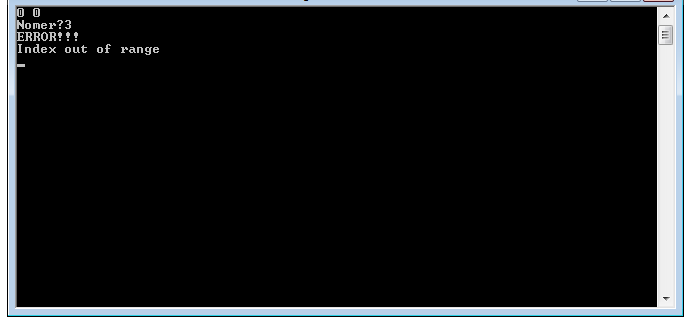
beg = temp;

}

return \*this;

}

**Результат работы программы**

****

Вариант реализации №2

**Создаем файл Lab9\_main.cpp**

#include "Vector.h"

#include "error.h"

#include <iostream>

using namespace std;

void print\_error(int x);

int main()

{

//контролируемый блок

try

{

Vector x(2);//вектор из двух элементов

Vector y;//пустой вектор

cout << x;//печать вектора х

cout << "Nomer?";

int i;

cin >> i;

//вывод элемента с номером i, если номер больше 2 или меньше 0, то

//генерируется исключительная ситуация

cout << x[i] << endl;

//добавление элемента в вектор, если MAX\_SIZE=2, то генерируется

//исключительная ситуация

y = x + 3;

cout << y;

//удалить один элемент из вектора

x = x - 1;

cout << x;

//удалить один элемент из вектора

x = x - 1;

cout << x;//вектор пустой

//удалить один элемент из вектора

//генерируется исключительная ситуация

x = x - 1;

}

//обработчик исключения

catch (const error& err)

{

cout << "ERROR!!!" << endl;

print\_error(err.what());

}//сообщение об ошибке

std::cin.ignore();

std::cin.get();

return 0;

}

void print\_error(int x)

{

switch (x) {

case 1: std::cout << "Given size is greater than MAX\_SIZE\n"; break;

case 2: std::cout << "Index is less than 0\n"; break;

case 3: std::cout << "Index out of range\n"; break;

case 4: std::cout << "Can't delete values\n"; break;

case 5: std::cout << "Size is less than given for delete\n"; break;

}

}

**Создаем файл Vector.cpp**

#pragma once

#include "Vector.h"

#include "error.h"

using namespace std;

Vector::Vector(int s)

{

//если текущий размер больше максимального, то генерируется исключение

if (s > MAX\_SIZE)

throw error(1);

size = s;

beg = new int[s];

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

beg[i] = 0;

}

Vector::Vector(const Vector& v)

{

size = v.size;

beg = new int[size];

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

beg[i] = v.beg[i];

}

Vector::~Vector()

{

if (beg != 0)

delete[] beg;

}

Vector::Vector(int s, int\* mas)

{

//если текущий размер больше максимального, то генерируется исключение

if (s > MAX\_SIZE)

throw error(1);

size = s;

beg = new int[size];

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

beg[i] = mas[i];

}

const Vector& Vector::operator=(const Vector& v)

{

if (this == &v)

return \*this;

if (beg != 0)

delete[] beg;

size = v.size;

beg = new int[size];

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

beg[i] = v.beg[i];

return \*this;

}

ostream& operator<<(ostream& out, const Vector& v)

{

if (v.size == 0) out << "Empty\n";

else

{

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

out << v.beg[i] << " ";

out << endl;

}

return out;

}

istream& operator >>(istream& in, Vector& v)

{

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

{

cout << ">";

in >> v.beg[i];

}

return in;

}

int& Vector::operator[](int i)

{

if (i < 0) throw error(2);//если индекс отрицательный, то генерируется исключение

//если индекс больше размер вектора, то генерируется исключение

if (i >= size) throw error(3);

return beg[i];

}

int Vector::operator()()

{

return size;

}

const Vector& Vector::operator+(int a)

{

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

beg[i] += a;

return \*this;

}

const Vector& Vector::operator-(int n)

{

if (n < 0) throw error(4);

if (size < n) throw error(5);

if (size == n)//если в вектор один элемент

{

size = 0;

delete[] beg;

beg = 0;

} else {

int\* temp = new int[size - n];

size -= n;

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

temp[i] = beg[i];

delete[] beg;

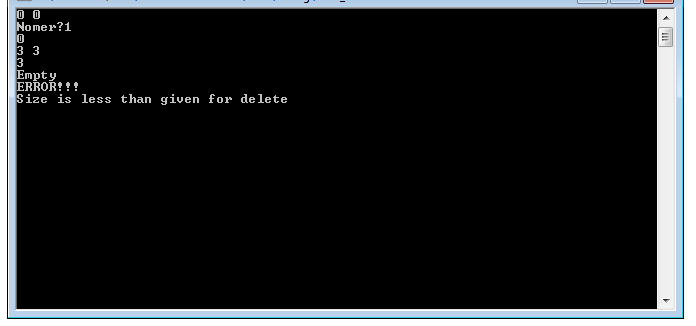
beg = temp;

}

return \*this;

}

**Результат работы программы**

****