Министерство науки и высшего образования Российской Федерации  
Пензенский государственный университет  
Кафедра вычислительная техника

**ОТЧËТ**  
по лабораторной работе №5  
по курсу «Объектно-ориентированное программирование»  
на тему «Механизмы исключений C++»

Выполнили студенты группы 22ВВП1:  
Ипполитов И. Д.

Хоссейни Нежад С. А. С. М.

Приняли:  
Евсеева Ю.И.  
Гудков А. А.

Пенза 2024

**Название**

Механизмы исключений C++

**Цель работы**

Изучить механизмы исключений C++

**Лабораторное задание**

1. Доработать шаблонный класс из предыдущей лабораторной работы таким образом, чтобы в программе осуществлялась генерация и обработка исключительных ситуаций.

Исключительные ситуации генерируются:

1. В операции –
2. Создать программу, использующую механизм обработки исключительных ситуаций, позволяющую выполнять перевод вещественного числа из одной системы счисления в другую. Перевод числа из тринадцатеричной системы счисления в восьмеричную.

**Листинг**

#include <windows.h>

#include <string>

#include <sstream>

#include <cmath>

#include <iostream>

using namespace std;

unsigned int charToInt(char chr)

{

if (chr >= '0' && chr <= '9')

return chr - '0';

else if (chr >= 'A' && chr <= 'C')

return chr - 'A' + 10;

else if (chr >= 'a' && chr <= 'c')

return chr - 'a' + 10;

throw "Это не тринадцатеричное число!";

}

float thirteenToDec(const char\* hex, int length, int size)

{

int count = 0;

float dec = 0;

for (int j = 0, i = size - 1; j < length; ++j, --i) {

if (hex[j] == '.' || hex[j] == ',')

{

i++;

count++;

if (count >= 2)

throw "Число введено не верно!";

continue;

}

dec += charToInt(hex[j]) \* pow(13, i);

}

return dec;

}

string decToEight(float dec)

{

int eight = 0;

float number = 0;

float number2 = 0;

float intpart = 0;

float intpart2 = 0;

string strT;

int num = dec, t = 0, d = 1;

while (num)

{

t += (num % 8) \* d;

num = num / 8;

d = d \* 10;

}

eight = t;

number = modf(dec, &intpart);

if (number == 0)

return to\_string(eight);

t = 0, d = 1, intpart = 0;

for (size\_t i = 0; i < 3; i++)

{

number = number \* 8;

number = modf(number, &intpart);

if (i == 2)

{

number2 = number \* 8;

number2 = modf(number2, &intpart2);

if (intpart2 >= 5)

intpart++;

}

t += intpart \* d;

d = d \* 10;

}

strT = to\_string(t);

reverse(strT.begin(), strT.end());

string result = to\_string(eight) + '.' + strT;

return result;

}

template <class T = int>

class MySet

{

private:

int size = 0;

T\* Arr = NULL;

public:

MySet(T\* p)

{

Arr = p;

}

bool check(T item)

{

for (size\_t i = 0; i < size; i++)

{

if (Arr[i] == item)

return false;

}

return true;

}

void insert(T item)

{

if (size == 0)

{

size = 1;

Arr = new T[size];

Arr[0] = item;

}

else

{

if (check(item) == true)

{

size++;

T\* Arrey = new T[size];

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

{

Arrey[i] = Arr[i];

}

Arrey[size - 1] = item;

delete[] Arr;

Arr = Arrey;

}

}

}

void erase(T item)

{

if (check(item) == true)

return;

int flag = 0;

size--;

T\* Arrey = new T[size];

for (size\_t i = 0; i < size; i++)

{

if (Arr[i] == item || flag == 1)

{

Arrey[i] = Arr[i + 1];

flag = 1;

}

else

Arrey[i] = Arr[i];

}

delete[] Arr;

Arr = Arrey;

}

int getSize() { return size; }

bool count(T item) {

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

{

if (Arr[i] == item)

{

return true;

}

}

return false;

}

MySet& operator \* (MySet& s)

{

T\* Arr3 = NULL;

MySet copy{ Arr3 };

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

for (int c = 0; c < s.getSize(); c++) {

if (Arr[i] == s.Arr[c] && copy.count(Arr[i]) == false) {

copy.insert(Arr[i]);

}

}

}

return copy;

}

MySet& operator- (T s)

{

if (size == 0)

return \*this;

if (s == 0)

throw "Число ноль нельзя удалить";

erase(s);

return \*this;

}

bool operator < (MySet& s)

{

if (size < s.getSize())

return true;

return false;

}

void show() {

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

{

cout << Arr[i] << " ";

}

cout << endl;

};

};

int main()

{

try

{

setlocale(LC\_ALL, "rus");

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

int\* p1 = NULL;

int\* p2 = NULL;

MySet<int> arr1{ p1 };

MySet<int> arr2{ p2 };

for (int i = 0; i < 5; i++) {

arr1.insert(i);

};

for (int i = 0; i < 6; i++) {

arr2.insert(i);

}

cout << "Множество 1:" << endl;

arr1.show();

cout << "Множество 2:" << endl;

arr2.show();

cout << "Множество 1 - '2':" << endl;

arr1 = arr1 - (2);

arr1.show();

cout << "Множество 1 - '0':" << endl;

arr1 = arr1 - (0);

arr1.show();

cout << "Множество 2 - '4':" << endl;

arr2 = arr2 - (4);

arr2.show();

if (arr1 < arr2)

cout << "Текущее множество меньше" << endl;

else if (arr1.getSize() == arr2.getSize())

cout << "Множества одинаковы" << endl;

else

cout << "Текущее множество больше" << endl;

cout << "Множество 1 \* 2:" << endl;

MySet<int> arr3{ arr1 \* arr2 };

arr3.show();

string str;

cout << "В 13 сс: ";

cin >> str;

int size;

if (str.find(".") == string::npos)

{

if(str.find(",") == string::npos)

size = str.length();

else

size = str.find(",");

}

else

size = str.find(".");

float dec = thirteenToDec(str.c\_str(), str.length(), size);

cout << "В 10 сс: " << dec << endl;

string eight = decToEight(dec);

cout << "В 8 сс: " << eight << endl;

}

catch (const char\* message)

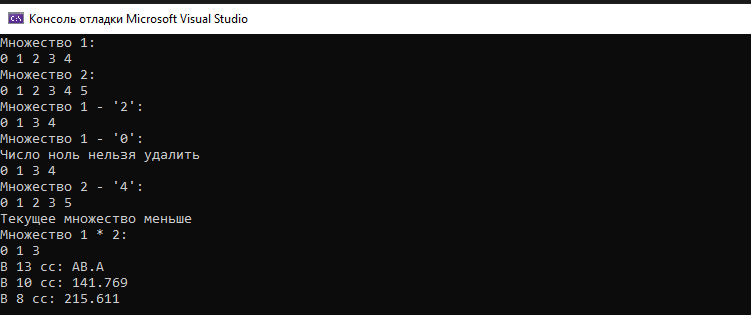
{

cout << message << endl;

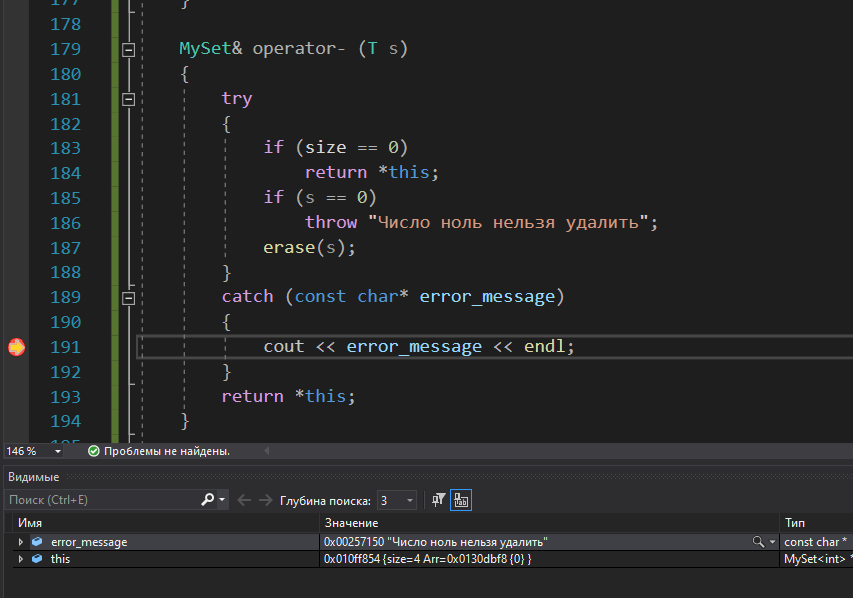
}

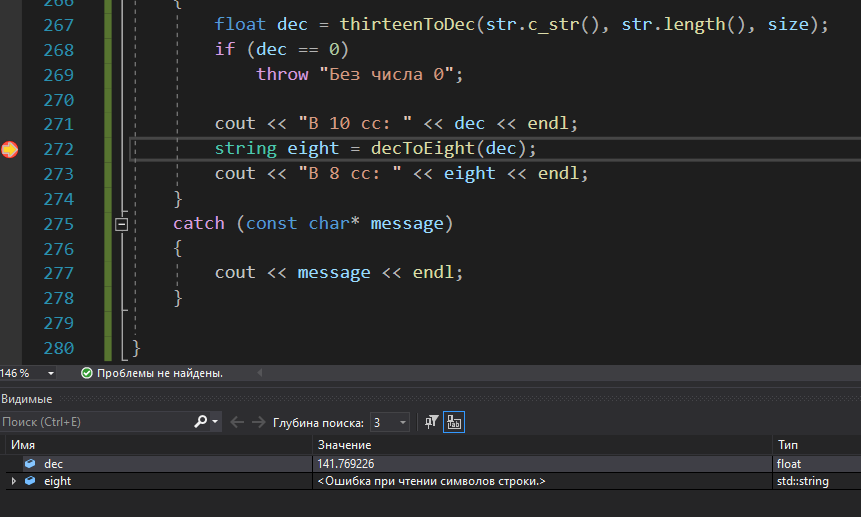
}

**Результат работы программы**



**Протокол трассировки программы**





**Вывод**

Мы изучили механизмы исключений C++