МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ЛИПЕЦКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра автоматизированных систем управления

Отчет по лабораторной работе №8.

По ОБЪЕКТО-ОРИЕНТИРОВАННОМУ ПРОГРАММИРОВАНИЮ

«Обработка исключений»

Выполнила:

студентка гр. ПМ-21-2 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Бутусова В.М.

Проверил:

доц., к.п.н. кафедры АСУ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Кургасов В. В.

Липецк 2022

**Цель работы:** научиться использовать встроенное в язык средство обработки исключительных ситуаций для предотвращения аварийного завершения программы в случае ошибок данных.

**Задание кафедры:** для класса, полученного в результате лабораторной работы №2, перегрузить еще как минимум три операции. Создать собственную иерархию класса обработки исключений. В ситуациях, при которых может возникать ошибка, генерировать собственное исключение и его обрабатывать. В программе должны быть использованы блоки обработки исключений: try, catch. В лабораторной работе описать все обрабатываемые в программе исключительные ситуации.

**Вариант №3:** Матрица вещественных чисел. Умножение двух матриц и присвоение результата третьей переменной-матрице осуществляется одной строкой C=A\*B.

Код программы:

#include <iostream>

using namespace std;

class Error\_B : public exception // ввод недопустимой размерности матрицы B

{

public:

Error\_B(const char\* msg) : exception(msg) {}

};

class Error\_Null : public exception // ввод размерности равной нулю

{

public:

Error\_Null(const char\* msg) : exception(msg) {}

};

class Error\_Number : public exception // ввод недопустимого числа

{

public:

Error\_Number(const char\* msg) : exception(msg) {}

};

class Error // клас обрабатывающий исключения

{

int a\_2;

public:

void getA(int& x, int& y) // функция для вода и проверки размерности матрицы A

{

while (1)// пока не будет введён правильный символ

{ //программа будет запрашиваеть его бесконечное число раз

try

{

cin >> x >> y;

a\_2 = y;

if (cin.fail())

throw Error\_Number("Вы ввели недопустимое значение.");

if (x == 0 || y == 0)

throw Error\_Null("Строки и столбцы не могут быть равны нулю");

cout << endl;

break;

}

catch (Error\_Number& ex)

{

cin.clear();

cin.ignore(INT\_MAX, '\n');

cout << ex.what() << endl;

cout << "Введите размерность матрицы A: ";

}

catch (Error\_Null& ex)

{

cout << ex.what() << endl;

cout << "Введите размерность матрицы A: ";

}

}

}

void getB(int& x, int& y)// функция для вода и проверки размерности матрицы B

{

while (1)// пока не будет введён правильный символ

{ //программа будет запрашиваеть его бесконечное число раз

try

{

cin >> x >> y;

if (cin.fail())

throw Error\_Number("Вы ввели недопустимое значение.");

if (x == 0 || y == 0)

throw Error\_Null("Строки и столбцы не могут быть равны нулю");

if (a\_2 != x)

throw Error\_B("Матрицу этой размерности нельзя умножить на матрицу A");

cout << endl;

break;

}

catch (Error\_Number& ex)

{

cin.clear();

cin.ignore(INT\_MAX, '\n');

cout << ex.what() << endl;

cout << "Введите размерность матрицы B: ";

}

catch (Error\_Null& ex)

{

cout << ex.what() << endl;

cout << "Введите размерность матрицы B: ";

}

catch (Error\_B& ex)

{

cout << ex.what() << endl;

cout << "Введите размерность матрицы B: ";

}

}

}

void getX(float& x)// функция для вода и проверки числа

{

while (1) // пока не будет введён правильный символ

{ //программа будет запрашиваеть его бесконечное число раз

try

{

cin >> x;

if (cin.fail())

throw Error\_Number("Вы ввели недопустимое значение.");

//cout << endl;

break;

}

catch (Error\_Number& ex)

{

cin.clear();

cin.ignore(INT\_MAX, '\n');

cout << ex.what() << endl;

//cout << "Введите число: ";

}

}

}

};

class Matrica

{

private:

int c1, c2;

float\*\* mas; //объявление массива и размерности

public:

Matrica()

{

c1 = 0;

c2 = 0;

mas = 0;

}

Matrica(float\*\* massiv, int n1, int n2)//конструктор с параметрами

{ //создаёт обрабатываемую матрицу

{

c1 = n1;

c2 = n2;

mas = new float\* [n1];

for (int i = 0; i < n1; i++)

{

mas[i] = new float[n2];

}

for (int i = 0; i < n1; i++)

{

for (int j = 0; j < n2; j++)

{

mas[i][j] = massiv[i][j];

}

}

}

}

Matrica(const Matrica& matricaA) // Конструктор копирования

{

c1 = matricaA.c1;

c2 = matricaA.c2;

mas = new float\* [c1];

for (int i = 0; i < c1; i++)

{

this->mas[i] = new float[c2];

}

for (int i = 0; i < c1; i++)

{

for (int j = 0; j < c2; j++)

{

this->mas[i][j] = matricaA.mas[i][j];

}

}

}

friend Matrica operator\* (Matrica& A, Matrica& B);

friend Matrica operator+ (Matrica& A, float& X);

friend Matrica operator- (Matrica& A, float& X);

friend Matrica operator\* (Matrica& A, float& X);

Matrica& operator=(const Matrica& ravno) //оператор присваивания

{ // присваевает матрице C результат перемножения матриц A и B

this->c1 = ravno.c1;

this->c2 = ravno.c2;

this->mas = new float\* [c1];

for (int i = 0; i < c1; i++)

{

this->mas[i] = new float[c2];

}

for (int i = 0; i < c1; i++)

{

for (int j = 0; j < c2; j++)

{

this->mas[i][j] = ravno.mas[i][j];

}

}

return \*this;

}

void vivod(string stroka)//функция выводит матрицу

{

cout << endl << stroka << endl;

for (int i = 0; i < c1; i++)

{

for (int j = 0; j < c2; j++)

{

cout << mas[i][j] << " ";

}

cout << endl;

}

}

~Matrica()//деструктор

{ // очищает массив

for (int i = 0; i < c2; i++)

{

delete[] mas[i];

}

delete[] mas;

}

};

Matrica operator-(Matrica& A, float& X)// Перегрузка оператора для вычитания из матрицы A вещественного числа

{

float\*\* N = new float\* [A.c1];

for (int i = 0; i < A.c1; i++)

{

N[i] = new float[A.c2];

for (int j = 0; j < A.c2; j++)

{

N[i][j] = 0;

N[i][j] += A.mas[i][j] - X;

}

}

Matrica new\_C(N, A.c1, A.c2); //для дальнейшего вывода матрицы на экран используем конструктор с параметрами

cout << endl << "Операция разности";

new\_C.vivod("Матрица C:"); // вызываем функцию вывода матрицы

for (int i = 0; i < A.c2; i++)

{

delete[] N[i];

}

delete[] N;

return new\_C;

}

Matrica operator\*(Matrica& A, float& X)// Перегрузка оператора для умножения матрицы A на вещественное число

{

float\*\* N = new float\* [A.c1];

for (int i = 0; i < A.c1; i++)

{

N[i] = new float[A.c2];

for (int j = 0; j < A.c2; j++)

{

N[i][j] = 0;

N[i][j] += A.mas[i][j] \* X;

}

}

Matrica new\_C(N, A.c1, A.c2); //для дальнейшего вывода матрицы на экран используем конструктор с параметрами

cout << endl << "Операция умножения на число";

new\_C.vivod("Матрица C:"); // вызываем функцию вывода матрицы

for (int i = 0; i < A.c2; i++)

{

delete[] N[i];

}

delete[] N;

return new\_C;

}

Matrica operator+ (Matrica& A, float& X)// Перегрузка оператора для сложения матрицы A и вещественного числа

{

float\*\* N = new float\* [A.c1];

for (int i = 0; i < A.c1; i++)

{

N[i] = new float[A.c2];

for (int j = 0; j < A.c2; j++)

{

N[i][j] = 0;

N[i][j] += A.mas[i][j] + X;

}

}

Matrica new\_C(N, A.c1, A.c2); //для дальнейшего вывода матрицы на экран используем конструктор с параметрами

cout << endl << "Операция сложения";

new\_C.vivod("Матрица C:"); // вызываем функцию вывода матрицы

for (int i = 0; i < A.c2; i++)

{

delete[] N[i];

}

delete[] N;

return new\_C;

}

Matrica operator\* (Matrica& A, Matrica& B) // Перегрузка оператора для умножения матрицы A на матрицу B

{

//производим умножение

float\*\* N = new float\* [A.c1];

for (int i = 0; i < A.c1; i++)

{

N[i] = new float[B.c2];

for (int j = 0; j < B.c2; j++)

{

N[i][j] = 0;

for (int k = 0; k < A.c2; k++)

{

N[i][j] += A.mas[i][k] \* B.mas[k][j];

}

}

}

Matrica new\_C(N, A.c1, B.c2); //для дальнейшего вывода матрицы на экран используем конструктор с параметрами

cout << endl << "Операция умножения";

new\_C.vivod("Матрица C:"); // вызываем функцию вывода матрицы

for (int i = 0; i < B.c2; i++)

{

delete[] N[i];

}

delete[] N;

return new\_C;

}

int main()

{

setlocale(LC\_ALL, "rus");

float\*\* mas1;

float\*\* mas2;

Error\* error = new Error;

int a1, a2, b1, b2;

float x;

int a;

cout << "Выберите операцию:" << endl;

cout << "1. Произведение матриц" << endl;

cout << "2. Умножение матрицы на число" << endl;

cout << "3. Разность матрицы и числа" << endl;

cout << "4. Сложение матрицы и числа" << endl;

cin >> a;

if (a == 1)

{

cout << "Введите размерность матрицы A: ";

error->getA(a1, a2);

mas1 = new float\* [a1];

for (int i = 0; i < a1; i++) //создание первого массива

{

mas1[i] = new float[a2];

}

cout << "Введите элементы матрицы A: " << endl;

for (int i = 0; i < a1; i++)

{

for (int j = 0; j < a2; j++)

{

float k;

error->getX(k);// проверка элемента массива

mas1[i][j] = k;

}

}

cout << "Введите размерность матрицы B: ";

error->getB(b1, b2);

mas2 = new float\* [b1];

for (int i = 0; i < b1; i++) //создание второго массива

{

mas2[i] = new float[b2];

}

cout << "Введите элементы матрицы B: " << endl;

for (int i = 0; i < b1; i++)

{

for (int j = 0; j < b2; j++)

{

float k;

error->getX(k);// проверка элемента массива

mas2[i][j] = k;

}

}

Matrica C;

Matrica A(mas1, a1, a2);

Matrica B(mas2, b1, b2);

A.vivod("Матрица A");

B.vivod("Матрица B");

C = A \* B;

}

if (a == 2)

{

cout << "Введите размерность матрицы A: ";

error->getA(a1, a2);

mas1 = new float\* [a1];

for (int i = 0; i < a1; i++) //создание первого массива

{

mas1[i] = new float[a2];

}

cout << "Введите элементы матрицы A: " << endl;

for (int i = 0; i < a1; i++)

{

for (int j = 0; j < a2; j++)

{

float k;

error->getX(k);// проверка элемента массива

mas1[i][j] = k;

}

}

cout << "Введите число: ";

error->getX(x);// проверка числа

Matrica C;

Matrica A(mas1, a1, a2);

A.vivod("Матрица A");

cout << "Число: " << x << endl;

C = A \* x;

}

if (a == 3)

{

cout << "Введите размерность матрицы A: ";

error->getA(a1, a2);

mas1 = new float\* [a1];

for (int i = 0; i < a1; i++) //создание первого массива

{

mas1[i] = new float[a2];

}

cout << "Введите элементы матрицы A: " << endl;

for (int i = 0; i < a1; i++)

{

for (int j = 0; j < a2; j++)

{

float k;

error->getX(k);// проверка элемента массива

mas1[i][j] = k;

}

}

cout << "Введите число: ";

error->getX(x);// проверка числа

Matrica C;

Matrica A(mas1, a1, a2);

A.vivod("Матрица A");

cout << "Число: " << x << endl;

C = A - x;

}

if (a == 4)

{

cout << "Введите размерность матрицы A: ";

error->getA(a1, a2);

mas1 = new float\* [a1];

for (int i = 0; i < a1; i++) //создание массива

{

mas1[i] = new float[a2];

}

cout << "Введите элементы матрицы A: " << endl;

for (int i = 0; i < a1; i++)

{

for (int j = 0; j < a2; j++)

{

float k;

error->getX(k); // проверка элемента массива

mas1[i][j] = k;

}

}

cout << "Введите число: ";

error->getX(x); // проверка числа

Matrica C;

Matrica A(mas1, a1, a2);

A.vivod("Матрица A");

cout << "Число: " << x << endl;

C = A + x;

}

}

**Пример работы программы.**

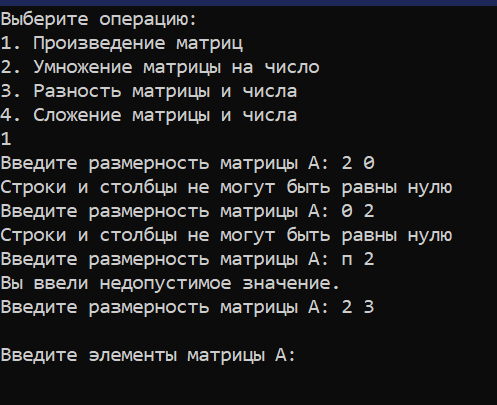


Рисунок 1. Если при вводе размерности количество строк или столбцов будет равно нулю или недопустимому значению, программа попросит ввести параметры заного.

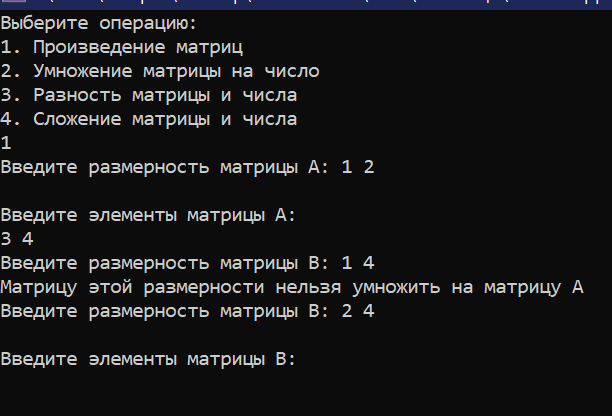


Рисунок 2. Если при вводе размерности матрицы B, её количество строк не совпадёт с количеством столбцов матрицы A, программа выдаст ошибку и попросит ввести параметры заново.

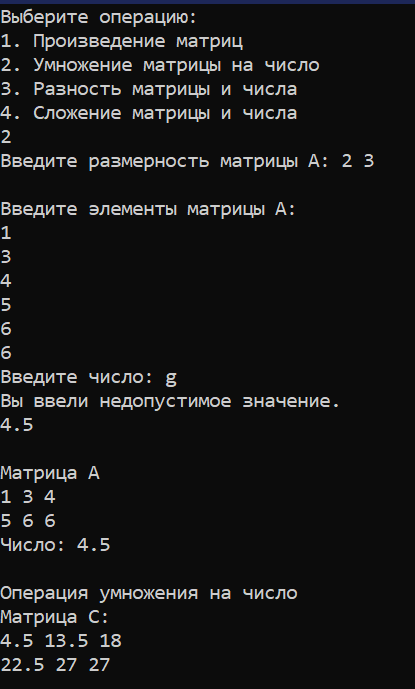


Рисунок 3. Если входное число для операции будет равно недопустимому значению, программа выдаст ошибку. Число будет необходимо ввести заново.

**Вывод:** в данной лабораторной работе я научилась использовать встроенное в язык средство обработки исключительных ситуаций для предотвращения аварийного завершения программы в случае ошибок данных.

**Контрольные вопросы.**

1. **Зачем нужен блок «try»?**

В блоке try **выполняются операции, которые могут привести к исключительной ситуации**.

1. **Зачем нужен блок «catch»?**

Блок catch **содержит инструкции, которые будут выполнены, если в блоке try произошла ошибка**. Если любая инструкция в блоке try выбрасывает исключение, то управление сразу же переходит в блок catch. Если в блок try не было выброшено исключение, то блок catch не выполняется.

1. **Какие исключения может обрабатывать каждый из блоков «catch»?**

Параметры catch работают также, как и параметры функции, причем параметры одного блока catch могут быть доступны и в другом блоке catch (который находится за ним). Исключения фундаментальных типов данных могут быть пойманы по значению (параметром блока catch является значение), но исключения не фундаментальных типов данных должны быть пойманы по константной ссылке (параметром блока catch является константная ссылка), чтобы избежать ненужного копирования.

1. **Как задать блок «catch», чтобы он мог обрабатывать любые исключительные ситуации?**

Если инструкция **catch** задает многоточие (...) вместо типа, **catch** блок обрабатывает каждый тип исключения.

1. **Для чего используется блок «finally»?**

Блок «finally» выполняется всегда, независимо от того, произошло исключение в процессе работы блока try или нет.

С помощью блока finally **можно выполнить очистку всех ресурсов, выделенных в блоке try.**

1. **Что будет, если в программе не задан блок «catch» для возникшей исключительной ситуации?**

Если в блоке **try** возникнет исключительная ситуация, которая не предусмотрена блоком **catch**, то вызывается стандартная функция **terminate()**, которая по умолчанию вызовет функцию **abort()**. Эта стандартная функция останавливает выполнение программы.

1. **Как работает обработка исключительных ситуаций, если заданы несколько вложенных блоков «try»?**

Блок try-catch может содержать вложенные блоки try-catch и если не будет определено соответствующего оператора catch на текущем уровне вложения, исключение будет поймано на более высоком уровне.

**8. Когда необходима обработка исключительных ситуаций? Приведите примеры.**

Обработка исключительных ситуаций необходима тогда, когда во время выполнения программы могут возникать ситуации, когда состояние внешних данных, устройств ввода-вывода или компьютерной системы в целом делает дальнейшие вычисления в соответствии с базовым алгоритмом невозможными или бессмысленными.

Пример: Целочисленное деление на ноль. Конечного результата у данной операции быть не может, поэтому ни дальнейшие вычисления, ни попытка использования результата деления не приведут к решению задачи.