# Лабораторная работа №6. Обработка исключений

**1 Цель работы**

Цель работы – ознакомиться с понятием исключительных ситуаций и способами их обработки, изучить методы создания классов исключений и их объектов, способов формирования исключений в программе.

**2 Порядок выполнения работы**

* ознакомиться с описанием лабораторной работы;
* получить задание у преподавателя по вариантам;
* разработать и отладить программу;
* составить и защитить отчет по лабораторной работе у преподавателя.

**3 Содержание отчета**

* титульный лист;
* краткое теоретическое описание;
* задание на лабораторную работу, включающее математическую формулировку задачи;
* результаты выполнения работы, включающие схему алгоритма, тексты программ, результаты вычислений;

**4 Краткая теория**

*Исключительная ситуация* (или исключение) – это ошибка, которая возникает во время выполнения программы. Типичные исключительные ситуации – это «деление на ноль», «достижение конца файла», «переполнение в арифметических операциях» и т.п.

Для реализации механизма обработки исключений в язык С++ введены следующие три ключевых слова: try (контролировать), catch (ловить), throw (генерировать, порождать, бросать, посылать, формировать).

Служебное слово try позволяет выделить в любом месте исполняемого текста так называемый *контролируемый блок*:

**try { операторы }**

Среди операторов, заключенных в фигурные скобки, могут быть: описания, определения, обычные операторы языка С++ и специальные операторы генерации (выброса) исключений:

**throw выражение\_генерации\_исключения;**

За этим блоком обязательно находятся один или несколько обработчиков исключений, каждый из которых идентифицируется служебным словом catch и имеет в общем случае следующий формат:

**catch (тип\_исключения имя) { операторы }**

Чтобы исключение было достаточно информативным, оно должно быть объектом класса, причем класс обычно определяется специально.

**5 Пример программы**

Создать массив произвольной длины, в который записать значения функции f(x) = √(1/x) при x∈[-2; 6], Δx = 0,4. Распечатать полученный массив. Предусмотреть обработку возникающих исключительных ситуаций.

**#include <iostream>**

**#include <math.h>**

**using namespace std;**

**//класс исключения "деление на ноль"**

**class ZeroDivideException**

**{**

**public:**

**int idx;**

**ZeroDivideException(int i)**

**{**

**idx = i;**

**}**

**};**

**//класс исключения "корень из отрицательного числа"**

**class SqrtException**

**{**

**public:**

**int idx;**

**SqrtException(int i)**

**{**

**idx = i;**

**}**

**};**

**//класс исключения "выход за пределы индексов массива"**

**class IndexException**

**{**

**public:**

**int idx;**

**IndexException(int i)**

**{**

**idx = i;**

**}**

**};**

**//функция установки очередного значения массива**

**void SetValue(double val,int i,int n,double\* ar)**

**{**

**if (val==0.0) //пытаемся делить на ноль**

**{**

**ar[i] = 0.0;**

**throw ZeroDivideException(i);**

**}**

**if (val<0) //пытаемся извлечь корень из отриц. числа**

**{**

**ar[i] = 0.0;**

**throw SqrtException(i);**

**}**

**if (i>=n) //недопустимый индекс массива**

**throw IndexException(i);**

**ar[i] = sqrt(1/val); //нормальная ситуация**

**}**

**//функция печати массива**

**void Print(int i,int n,double\* ar)**

**{**

**if (i>=n) //недопустимый индекс массива**

**throw IndexException(i);**

**cout<<ar[i]<<' ';**

**}**

**void main()**

**{**

**int n = 20;**

**double\* a = new double[n];**

**double x = -2.0;**

**for (int i=0;x<=6.0;i++,x+=0.4)**

**{**

**try**

**{**

**SetValue(x,i,n,a); //пытаемся сформировать массив**

**}**

**catch (ZeroDivideException ex) //перехват исключений**

**{**

**cout<<"Zero divide error in index "<<ex.idx<<endl;**

**}**

**catch (SqrtException ex)**

**{**

**cout<<"Sqrt error in index "<<ex.idx<<endl;**

**}**

**catch (IndexException ex)**

**{**

**cout<<"Index out of range in index "<<ex.idx<<endl;**

**}**

**}**

**for (int i=0;i<2\*n;i++) //генерация некорректного доступа**

**{**

**try**

**{**

**Print(i,n,a); //пытаемся распечатать массив**

**}**

**catch (IndexException ex) //перехват исключений**

**{**

**cout<<"Index out of range in index "<<ex.idx<<endl;**

**}**

**}**

**cout<<endl;**

**delete []a;**

**}**

**6 Контрольные вопросы**

1. Что такое исключительная ситуация? Приведите примеры программных исключений.
2. Как выделить в программе контролируемый блок?
3. С помощью какого ключевого слова осуществляется генерация исключительных ситуаций? Укажите существующие формы этого оператора.
4. Перечислите известные Вам формы описания обработчиков исключений.
5. Каким образом осуществляется обработка исключений во вложенных контролируемых блоках?

**7 Варианты заданий для самостоятельного решения**

Для каждого варианта необходимо создать три массива *a, b* и *с* размерами соответственно *n1, n2* и *n3* (*n1≠n2≠n3*). В массив *а* занести значения функции *f(x)* согласно варианту (при возникновении исключения заносить нули). Массив *b* заполнить случайными числами (среди них должны быть и отрицательные числа и нули). Массив *с* формируется согласно варианту. Предусмотреть и обработать возникающие при этом исключительные ситуации (деление на ноль, корень из отрицательного числа, арифметическое переполнение, выход за пределы диапазона индексов массива и т.п.). Варианты заданий приведены в таблице 6.1.

*Таблица 6.1 – Варианты заданий*

|  |  |  |
| --- | --- | --- |
| Вариант | Функция *f(x)* | Способ формирования массива *с* |
|  |  | *ci = ai + 1/bi* |
|  |  | *ci = ai – 1/bi* |
|  |  | *ci = ai /bi* |
|  |  | *ci = ai-1 + 1/bi+1* |
|  |  | *ci = ai+! – 1/bi-1* |
|  |  | *ci = ai /bi+1* |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  | *ci = ai+1 + 1/(bi+1-1)* |
|  |  | *ci = ai-! – 2/(bi-1+1)* |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |