МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

ИНСТИТУТ НЕПРЕРЫВНОГО И ДИСТАНЦИОННОГО ОБРАЗОВАНИЯ

|  |
| --- |
| КАФЕДРА КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И ПРОГРАММНОЙ ИНЖЕНЕРИИ |

ОЦЕНКА

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| старший преподаватель |  |  |  | Е.О. Шумова |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №5 |
| Обработка исключительных ситуаций |
| по дисциплине: Объектно-ориентированное программирование |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ гр. № | Z1431 |  |  |  | М.Д. Быстров |
|  | номер группы |  | подпись, дата |  | инициалы, фамилия |
| Студенческий билет № | 2021/3572 | |  |  |  |

Санкт-Петербург 2023

**Условие**

Цель работы: изучить принципы построения консольных приложений, применив на практике знания базовых синтаксических конструкций языка C++ и объектно-ориентированного программирования. Закрепить знания по теме: классы, наследование классов, варианты доступа.

Общая часть задания:

Создать класс Массив, содержащий:

- одномерный целочисленный массив

- размерность массива определяется суммой двух последних цифр шифра студента (если сумма получается меньше 12, то размерность массива будет равна 12)

- поле, для хранения длины массива

- конструктор, в котором осуществляется ввод данных в массив

- расчетные функции (согласно варианту)

- функцию вывода исходного массива

- задействовать механизм обработки исключительных ситуаций (не менее 2-3 ситуаций)

- одну из ситуаций обработать с использованием собственного класса исключений

Индивидуальное задание (вариант 2):

В одномерном массиве, состоящем из n вещественных элементов, вычислить:

• сумму положительных элементов массива;

• произведение элементов массива, расположенных между максимальным по модулю и минимальным по модулю элементами.

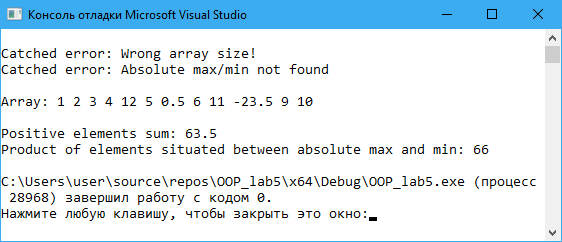
• упорядочить элементы массива по убыванию.

**Полный текст (листинг) программы**

1. Файл «main.cpp»
2. #include <iostream>
3. #include <string>
4. #include "MyArray.h"
5. #include "ArraySizeException.h"
7. int main()
8. {
9. cout << "LR#5 VAR 2" << endl << endl;
11. **try**
12. {
13. MyArray\* arr = new MyArray(new double[10] {1, 2, 3, 4, 5, 6, 7, 8, 9, 0}, 10);
14. }
15. catch (ArraySizeException e)
16. **{**
17. cout << "Catched error: " << e.what() << endl;
18. }
20. try
21. **{**
22. MyArray\* arr = new MyArray(new double[12] {1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1}, 12);
24. arr->getBetweenMaxMinProduct();
25. }
26. **catch (std::string e)**
27. {
28. cout << "Catched error: " << e << endl;
29. }
31. **cout << endl;**
33. try
34. {
35. MyArray\* arr = new MyArray(new double[12] {1, 2, 3, 4, 12, 5, 0.5, 6, 11, -23.5, 9, 10}, 12);
37. cout << "Array: " << arr->getArrayStr() << endl;
38. cout << "Positive elements sum: " << arr->getPositiveElementsSum() << endl;
39. cout << "Product of elements situated between absolute max and min: " << arr->getBetweenMaxMinProduct() << endl;
40. }
41. **catch (...)**
42. {
43. cout << "Catched error" << endl;
44. }
46. **return 0;**
47. }
48. Файл «MyArray.h»
49. #include <string>
51. #pragma once
52. class MyArray
53. **{**
54. private:
56. double\* arr;
57. int arraySize;
59. public:
60. MyArray(double\* const source, int sourceSize);
61. ~MyArray();
63. **std::string getArrayStr() const;**
65. double getPositiveElementsSum() const;
66. double getBetweenMaxMinProduct() const;
67. };

70. Файл «MyArray.cpp»
71. #include <sstream>
72. #include <limits>
73. #include "MyArray.h"
74. #include "ArraySizeException.h"
76. #define ARRAY\_SIZE 12
78. MyArray::MyArray(double\* const source, int sourceSize)
79. {
80. **this->arraySize = ARRAY\_SIZE;**
81. this->arr = new double[this->arraySize];
83. if (sourceSize != this->arraySize)
84. {
85. **throw ArraySizeException();**
86. }
88. for (int i = 0; i < this->arraySize; i++)
89. {
90. **this->arr[i] = source[i];**
91. }
92. }
94. MyArray::~MyArray()
95. **{**
96. delete[] this->arr;
97. }
99. std::string MyArray::getArrayStr() const
100. **{**
101. std::stringstream ss;
103. for (int i = 0; i < this->arraySize; i++)
104. {
105. **if (i != 0)**
106. {
107. ss << " ";
108. }
110. **ss << this->arr[i];**
111. }
113. ss << std::endl;
115. **return ss.str();**
116. }
118. double MyArray::getPositiveElementsSum() const
119. {
120. **double sum = 0;**
122. for (int i = 0; i < this->arraySize; i++)
123. {
124. if (this->arr[i] > 0)
125. **{**
126. sum += this->arr[i];
127. }
128. }
130. **return sum;**
131. }
133. double MyArray::getBetweenMaxMinProduct() const
134. {
135. **int maxIdx = -1;**
136. int minIdx = -1;
138. double max = std::numeric\_limits<double>::min();
139. double min = std::numeric\_limits<double>::max();
141. for (int i = 0; i < this->arraySize; i++)
142. {
143. if (abs(this->arr[i]) > max)
144. {
145. **max = abs(this->arr[i]);**
147. maxIdx = i;
148. }
150. **if (abs(this->arr[i]) < min)**
151. {
152. min = abs(this->arr[i]);
154. minIdx = i;
155. **}**
156. }
158. if (maxIdx == minIdx)
159. {
160. **throw std::string("Absolute max/min not found");**
161. }
163. int direction = maxIdx > minIdx ? 1 : -1;
165. **double product = 1;**
167. for (int i = minIdx + direction; i != maxIdx; i += direction)
168. {
169. product \*= this->arr[i];
170. **}**
172. return product;
173. }
175. Файл «ArraySizeException.h»
176. #include <iostream>
177. using namespace std;
179. #pragma once
180. **class ArraySizeException : public std::exception**
181. {
182. public:
183. const char\* what();
185. **};**
186. Файл «ArraySizeException.cpp»
187. #include "ArraySizeException.h"
189. const char\* ArraySizeException::what()
190. {
191. **return "Wrong array size!";**
192. }

**Работа программы**



*Рисунок 1 Работа программы*

**Выводы**

В ходе выполнения лабораторной работы №5 получены навыки работы с исключениями в языке C++.

Создан класс исключения. При работе программы предусмотрены ситуации, при которых происходит выброс исключения. Продемонстрирован перехват ошибок различных типов, в том числе созданных самостоятельно.

Написанное приложение показывает, каким образом можно организовать обработку ошибок в ходе выполнения программы при помощи перехвата исключений.