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.

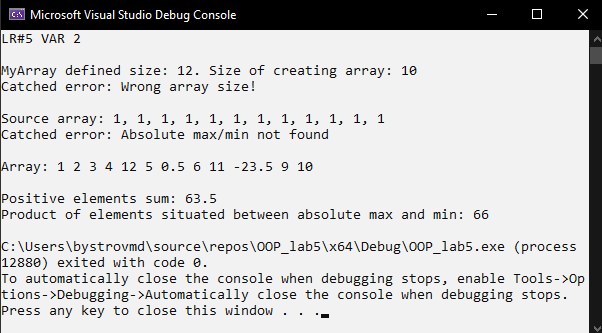
• упорядочить элементы массива по убыванию.

**Полный текст (листинг) программы**

1. Файл «main.cpp»
2. #include <iostream>
3. #include <string>
4. #include "MyArray.h"
5. #include "ArraySizeException.h"
7. #define ARRAY\_SIZE 12
9. int main()
10. {
11. **cout << "LR#5 VAR 2" << endl << endl;**
13. try
14. {
15. cout << "MyArray defined size: " << ARRAY\_SIZE << ". Size of creating array: " << 10 << endl;
17. MyArray\* arr = new MyArray(new double[10] {1, 2, 3, 4, 5, 6, 7, 8, 9, 0}, 10);
19. delete arr;
20. }
21. **catch (ArraySizeException e)**
22. {
23. cout << "Catched error: " << e.what() << endl;
24. }
26. **cout << endl;**
28. try
29. {
30. const int length = 12;
32. double\* sourceArray = new double[12] {1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1};
34. cout << "Source array: ";
36. **for (int i = 0; i < length; i++)**
37. {
38. if (i > 0)
39. {
40. cout << ", ";
41. **}**
43. cout << sourceArray[i];
44. }
46. **cout << endl;**
48. MyArray\* arr = new MyArray(sourceArray, length);
50. arr->getBetweenMaxMinProduct();
52. delete arr;
53. }
54. catch (std::string e)
55. {
56. **cout << "Catched error: " << e << endl;**
57. }
59. cout << endl;
61. **try**
62. {
63. MyArray\* arr = new MyArray(new double[12] {1, 2, 3, 4, 12, 5, 0.5, 6, 11, -23.5, 9, 10}, 12);
65. cout << "Array: " << arr->getArrayStr() << endl;
66. **cout << "Positive elements sum: " << arr->getPositiveElementsSum() << endl;**
67. cout << "Product of elements situated between absolute max and min: " << arr->getBetweenMaxMinProduct() << endl;
69. delete arr;
70. }
71. **catch (...)**
72. {
73. cout << "Catched error" << endl;
74. }
76. **return 0;**
77. }
78. Файл «MyArray.h»
79. #include <string>
81. #pragma once
82. class MyArray
83. **{**
84. private:
86. double\* arr;
87. int arraySize;
89. public:
90. MyArray(double\* const source, int sourceSize);
91. ~MyArray();
93. **std::string getArrayStr() const;**
95. double getPositiveElementsSum() const;
96. double getBetweenMaxMinProduct() const;
97. };

100. Файл «MyArray.cpp»
101. #include <sstream>
102. #include <limits>
103. #include "MyArray.h"
104. #include "ArraySizeException.h"
106. #define ARRAY\_SIZE 12
108. MyArray::MyArray(double\* const source, int sourceSize)
109. {
110. **this->arraySize = ARRAY\_SIZE;**
111. this->arr = new double[this->arraySize];
113. if (sourceSize != this->arraySize)
114. {
115. **throw ArraySizeException();**
116. }
118. for (int i = 0; i < this->arraySize; i++)
119. {
120. **this->arr[i] = source[i];**
121. }
122. }
124. MyArray::~MyArray()
125. **{**
126. delete[] this->arr;
127. }
129. std::string MyArray::getArrayStr() const
130. **{**
131. std::stringstream ss;
133. for (int i = 0; i < this->arraySize; i++)
134. {
135. **if (i != 0)**
136. {
137. ss << " ";
138. }
140. **ss << this->arr[i];**
141. }
143. ss << std::endl;
145. **return ss.str();**
146. }
148. double MyArray::getPositiveElementsSum() const
149. {
150. **double sum = 0;**
152. for (int i = 0; i < this->arraySize; i++)
153. {
154. if (this->arr[i] > 0)
155. **{**
156. sum += this->arr[i];
157. }
158. }
160. **return sum;**
161. }
163. double MyArray::getBetweenMaxMinProduct() const
164. {
165. **int maxIdx = -1;**
166. int minIdx = -1;
168. double max = std::numeric\_limits<double>::min();
169. double min = std::numeric\_limits<double>::max();
171. for (int i = 0; i < this->arraySize; i++)
172. {
173. if (abs(this->arr[i]) > max)
174. {
175. **max = abs(this->arr[i]);**
177. maxIdx = i;
178. }
180. **if (abs(this->arr[i]) < min)**
181. {
182. min = abs(this->arr[i]);
184. minIdx = i;
185. **}**
186. }
188. if (maxIdx == minIdx)
189. {
190. **throw std::string("Absolute max/min not found");**
191. }
193. int direction = maxIdx > minIdx ? 1 : -1;
195. **double product = 1;**
197. for (int i = minIdx + direction; i != maxIdx; i += direction)
198. {
199. product \*= this->arr[i];
200. **}**
202. return product;
203. }
205. Файл «ArraySizeException.h»
206. #include <iostream>
207. using namespace std;
209. #pragma once
210. **class ArraySizeException : public std::exception**
211. {
212. public:
213. const char\* what();
215. **};**
216. Файл «ArraySizeException.cpp»
217. #include "ArraySizeException.h"
219. const char\* ArraySizeException::what()
220. {
221. **return "Wrong array size!";**
222. }

**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.