

Объектно-ориентированное программирование.

Лабораторная работа №5.

Обработка исключений

Макаров П. А.

Содержание

1	Задания для самостоятельной работы	1
2	Список источников	3

1 Задания для самостоятельной работы

1. Наберите текст программы, приведённой в Листинге 1, обращая внимание на вызов функции `scanf()`. Значение, возвращаемое этой функцией, позволяет отследить количество “правильно распознанных” фактических параметров.

```
1 #include <stdio.h>
2
3 int main(void) {
4     int x, y, z, status;
5
6     printf("Input 3 space-separated integers x, y, z: ");
7     status = scanf("%d %d %d", &x, &y, &z);
8
9     printf("Number of correctly processed arguments: %d\n",
10          status);
11    printf("The variables x, y, z took the values: %d, %d, %d\n", x, y, z);
12
13    return 0;
14 }
```

Листинг 1: Анализ ошибок, возможных при работе функции `scanf`

Скомпилируйте программу и запустите её несколько раз вводя данные корректно и некорректно:

```
$ gcc scanf_error.c -o scanf_error
$ ./scanf_error
Input 3 space-separated integers x, y, z: 1 2 3
Number of correctly processed arguments: 3
The variables x, y, z took the values : 1, 2, 3
$ ./scanf_error
Input 3 space-separated integers x, y, z: 1 2 a
...
```

```
$ ./scanf_error
Input 3 space-separated integers x, y, z: 1 3.4 7
...
```

Объясните поведение программы в приведённых примерах.

2. Изучите системную документацию, посвященную функциям `exit`, `strerror`, `perror` и `error`, глобальной переменной `errno` и стандартному потоку ошибок `stderr`.
3. Напишите программу, выполняющую операции сложения, умножения и транспонирования прямоугольных матриц. Обработайте с помощью генерации исключений ситуацию, когда невозможно выделить память под матрицу (переменные, задающие размеры имеют отрицательные или слишком большие размеры).
4. Добавьте к предыдущему заданию обработку исключения, связанного с ситуацией, когда невозможно осуществить сложение и умножение матриц из-за несоответствия их размеров.
5. Напишите функцию поиска местоположения заданного элемента в массиве. Функция должна возвращать номер найденного элемента. Обработайте с помощью генерации исключения ситуацию, когда заданный элемент в массиве не найден.
6. Напишите программу работы с текстовыми файлами. Имя файла должно вводиться пользователем. Обработайте исключительную ситуацию отсутствия требуемого файла.
7. Напишите, скомпилируйте и исследуйте принцип работы программы, считающей число строк в файлах, с применением простейшего механизма обработки исключений. Исходный текст программы приведён в Листинге 2.

```
1 #include <stdio>
2 #include <stdlib>
3 #include <ctime>
4
5 unsigned long lines_of_file(const char *file_name);
6
7 int main(int argc, char **argv) {
8     if(argc < 2) {
9         fprintf(stderr, "No file name!\n");
10        return 1;
11    }
12    srand(time(NULL));
13    try {
14        for(int i = 1; i < argc; i++) {
15            unsigned long res = lines_of_file(argv[i]);
16            printf("The file %s contains %ld lines\n", argv[
17                i], res);
18        }
19    } catch(const char *exception) {
20        fprintf(stderr, "Exception (string): %s\n",
21            exception);
22    }
23    return 1;
24 }
```

```

22     }
23     catch(int x) {
24         fprintf(stderr, "Exception (int): %d\n", x);
25         return 1;
26     }
27     catch(...) {
28         fprintf(stderr, "Something strange happened\n");
29         return 1;
30     }
31     return 0;
32 }
33
34 unsigned long lines_of_file(const char *file_name) {
35     FILE *fp = fopen(file_name, "r");
36
37     if(fp == NULL) {
38         int x = rand()%4;
39         switch(x) {
40             case 1: throw "file open error";
41             case 2: throw rand();
42             case 3: throw 2.5;
43             default: throw;
44         }
45     }
46
47     unsigned long n = 0;
48     int c = 0;
49
50     while((c = fgetc(fp)) != EOF)
51         if(c == '\n')
52             n++;
53
54     fclose(fp);
55
56     return n;
57 }

```

Листинг 2: Простейшая обработка исключений

8. Решите предыдущую задачу с помощью класса `FileException`, описанного в Лекции №5.

2 Список источников

1. <http://www.cplusplus.com>.
2. Подбельский В. В., Фомин С. С. Программирование на языке Си.
3. Столяров А. В. [Введение в язык Си++](#).
4. Андрианова А. А., Исмагилов Л. Н., Мухтарова Т. М. Объектно-ориентированное программирование на C++.
5. OpenNET: Интерактивная система просмотра системных руководств <https://www.opennet.ru/man.shtml>.