## Практическая работа 9

**Тема:** Тестирование программного продукта методом «черного ящика»

### 1. Цель

Научиться проводить тестирование «черного ящика» разными способами на примере программы, вычисляющей средний балл группы студентов.

### 2. Инструктаж

Проводится согласно инструкции по охране труда при работе в лаборатории Технологии разработки БД, Автоматизированного проектирования технологических процессов и программирования №319 ИОТ – 029 2016.

### 3. Перечень средств обучения

Персональный компьютер с установленной средой разработки (например, Visual Studio, VS Code).

## 4. Практическая и экспериментальная часть

### 4.1 Задание

Массив a1, a2, …, an содержит экзаменационные оценки студентов группы по высшей математике. Необходимо определить **средний балл группы**.

### 4.2 Анализ задания

**Предусловия:**

Количество студентов n ≥ 1.

Все элементы массива – целые числа в диапазоне 0–100.

**Постусловия:**

Вычисляется средний балл avg = (a1 + a2 + … + an) / n.

Если массив пустой (n = 0), программа должна выдавать сообщение об ошибке.

### 4.3 Построение дерева разбиений

**Условия ввода:**

Количество элементов массива (n):

n < 1 → недопустимый класс

n ≥ 1 → допустимый класс

Значения элементов массива (ai):

ai < 0 → недопустимый класс

0 ≤ ai ≤ 100 → допустимый класс

ai > 100 → недопустимый класс

**Дерево разбиений**

| **Уровень** | **Разбиение** | **Классы эквивалентности** |
| --- | --- | --- |
| 1 | Количество студентов | n < 1, n ≥ 1 |
| 2 | Значения оценок | ai < 0, 0 ≤ ai ≤ 100, ai > 100 |

**Итог:** Получаем тестовые классы:

n ≥ 1, 0 ≤ ai ≤ 100 → нормальный случай

n < 1 → ошибка: массив пустой

n ≥ 1, ai < 0 → ошибка: оценка меньше 0

n ≥ 1, ai > 100 → ошибка: оценка больше 100

### 4.4 Тестовые варианты

| **Номер теста** | **Назначение теста** | **Значения исходных данных** | **Ожидаемый результат** | **Реакция программы** | **Вывод** |
| --- | --- | --- | --- | --- | --- |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | Проверка корректного ввода числа | 5 | Программа принимает число | Программа приняла число | Тест пройден |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 2 | Проверка отрицательного числа | -3 | Сообщение об ошибке | Сообщение об ошибке выведено | Тест пройден |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 3 | Проверка ввода нуля | 0 | Сообщение об ошибке | Сообщение об ошибке выведено | Тест пройден |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 4 | Проверка нечислового ввода | "abc" | Сообщение об ошибке | Сообщение об ошибке выведено | Тест пройден |

4.5 Программа на C# для тестирования

using System;

class Program

{

static void Main()

{

int[] grades1 = { 90, 80, 70, 100 };

int[] grades2 = { 100 };

int[] grades3 = { };

int[] grades4 = { 50, -10, 80 };

int[] grades5 = { 70, 105, 80 };

int[] grades6 = { 0, 100 };

Test(grades1);

Test(grades2);

Test(grades3);

Test(grades4);

Test(grades5);

Test(grades6);

}

static void Test(int[] grades)

{

try

{

if (grades.Length == 0)

{

Console.WriteLine("Ошибка: массив пустой");

return;

}

foreach (int grade in grades)

{

if (grade < 0 || grade > 100)

{

Console.WriteLine("Ошибка: недопустимая оценка " + grade);

return;

}

}

double avg = 0;

foreach (int grade in grades)

avg += grade;

avg /= grades.Length;

Console.WriteLine("Средний балл: " + avg);

}

catch (Exception ex)

{

Console.WriteLine("Ошибка: " + ex.Message);

}

}

}

### 4.6 Вывод

Метод «черного ящика» позволяет выявлять ошибки на основе спецификации без знания внутренней структуры программы.

Построение классов эквивалентности и анализ граничных значений позволяет минимизировать количество тестов и проверить все критические сценарии.

На примере вычисления среднего балла были протестированы нормальные и граничные случаи, а также ошибки ввода.

### ****Контрольные вопросы и ответы****

**Охарактеризуйте понятие «тестирование «черным ящиком». К какому виду тестирования он относится?**  
Тестирование «черным ящиком» — это метод проверки программного продукта без знания внутренней структуры кода, основываясь только на входных данных и ожидаемых результатах. Относится к **функциональному тестированию**.

**Дайте определение классу эквивалентности.**  
Класс эквивалентности — это множество входных данных, для которых программа должна вести себя одинаково. Тестирование одного значения из класса считается достаточным для проверки всей группы.

**Расскажите правила формирования классов эквивалентности.**

Разделяют все входные данные на группы: **корректные** и **некорректные**.

В каждой группе выбирается одно типичное значение для тестирования.

Все значения внутри класса считаются взаимозаменяемыми для теста.

**Опишите порядок проведения анализа граничных значений.**

Определяются минимальные и максимальные допустимые значения.

Проверяются значения **на границах**: минимальные, максимальные, чуть меньше минимального и чуть больше максимального.

Результаты анализа помогают выявить ошибки при экстремальных значениях входных данных.