–Пермский филиал федерального государственного автономного образовательного учреждение высшего образования

Национальный исследовательский университет

«Высшая школа экономики»

Факультет экономики, менеджмента и бизнес-информатики

Пермяков Андрей Владимирович

**ЛАБОРАТОРНАЯ РАБОТА №2**

*Отчёт по лабораторной работе*

студента образовательной программы бакалавриата «Программная инженерия» по направлению подготовки *09.03.04 Программная инженерия*

Преподаватель

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

В.Д. Марквирер

Пермь, 2021

**Содержание**

[Задача №1 (14) 3](#_Toc85249035)

[1. Анализ 3](#_Toc85249036)

[2. Алгоритм 5](#_Toc85249037)

[3. Программа 6](#_Toc85249038)

[4. Тесты 8](#_Toc85249039)

[Задача №2 (21) 10](#_Toc85249040)

[1. Анализ 10](#_Toc85249041)

[2. Алгоритм 11](#_Toc85249042)

[3. Программа 12](#_Toc85249043)

[4. Тесты 13](#_Toc85249044)

[Задача №3 (48) 14](#_Toc85249045)

[1. Анализ 14](#_Toc85249046)

[2. Алгоритм 15](#_Toc85249047)

[3. Программа 16](#_Toc85249048)

[4. Тесты 17](#_Toc85249049)

**Задача №1 (14)**

**Задание:** дана последовательность из n целых чисел. Найти количество элементов этой последовательности, кратных числу К.

1. **Анализ**

Исходные данные*:*

* n – длина последовательности (целое число);
* K – кратное число (вещественное число);
* , …, – последовательности элементов (целое число).

Результат: multiplesNum – количество элементов последовательности, кратных числу K.

Анализ решения: используется цикл с параметром, т.к. известно количество элементов последовательности.

Составим таблицу анализа решения (таблица 1).

***Таблица 1 – Анализ решения***

| **Исх. данные** | **Классы** | **Ожидаемый результат** |
| --- | --- | --- |
| n, K | * n – целое число; * K – вещественное число. | multiplesNum |
| * n <= 0; * K = 0. | Невозможно определить число кратных чисел |
| * n – не целое число или не число * K – не вещественное число или не число. | Неверный ввод |
|  | Целое число | multiplesNum |
| Не целое число или не число | Неверный ввод |

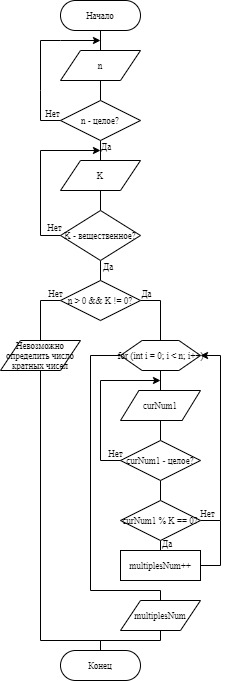
Классы входных данных:

1. n > 0 (целое). Результат: multiplesNum.
2. n <= 0 (целое). Результат: [!] Невозможно определить число кратных чисел.
3. n – не целое число или не число. Результат: [!] Неверный ввод.
4. K = 0. Результат: [!] Невозможно определить число кратных чисел.
5. K – вещественное число. Результат: multiplesNum.
6. К – не вещественное число или не число. Результат: [!] Неверный ввод.
7. – целое число. Результат: multiplesNum.
8. – не целое число или не число. Результат: [!] Неверный ввод.

Классы выходных данных:

1. multiplesNum.
2. Неверный ввод.
3. Невозможно определить число кратных чисел.
4. **Алгоритм**

Построим блок-схему для первого задания (Рисунок 1).



***Рисунок 1– Блок-схема задания №1***

1. **Программа**

int n;

double K;

int multiplesNum = 0;

bool isRead;

Console.WriteLine("\nЗАДАЧА №1 (14)\n");

do

{

Console.WriteLine("Введите число n - длину последовательности (типа int): ");

string value = Console.ReadLine();

isRead = Int32.TryParse(value, out n);

if (!isRead)

Console.WriteLine("[!] Неверный ввод");

} while (!isRead);

do

{

Console.WriteLine("Введите число K (типа int, float или double): ");

string value = Console.ReadLine();

isRead = Double.TryParse(value, out K);

if (!isRead)

Console.WriteLine("[!] Неверный ввод");

} while (!isRead);

if (n > 0 && K != 0)

{

Console.WriteLine("Вводите числа последовательности (типа int, float или double).");

Console.WriteLine("После каждого ввода нажимайте Enter: ");

for (int i = 0; i < n; i++)

{

int curNum1;

do

{

string value = Console.ReadLine();

isRead = Int32.TryParse(value, out curNum1);

if (!isRead)

Console.WriteLine("[!] Неверный ввод");

} while (!isRead);

if (curNum1 % K == 0)

multiplesNum++;

}

Console.WriteLine($"Кол-во чисел n, кратных K равно {multiplesNum}");

}

else Console.WriteLine("[!] Невозможно определить число кратных чисел");

1. **Тесты**

Составим таблицу тестов (таблица 2), в том числе по методу черного ящика (таблица 3) и белого ящика (см. таблицу 4).

***Таблица 2 – Тесты***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **n** | **K** |  | **Полученный результат** | **Ожидаемый результат** | **Примечание** |
| Т1 | 2 | 2 | “a”; 1,1; 1; 3 | Неверный ввод;  Неверный ввод;  multiplesNum = 0 | Неверный ввод;  Неверный ввод;  multiplesNum = 0 |  |
| Т2 | “a”, 0 | – | – | Неверный ввод; Неверный ввод;  Неверный ввод;  Невозможно определить число кратных чисел | Неверный ввод; Неверный ввод;  Неверный ввод;  Невозможно определить число кратных чисел |  |
| Т3 | 2 | “a”, 0 | – | Неверный ввод;  Невозможно определить число кратных чисел | Неверный ввод;  Невозможно определить число кратных чисел |  |
| Т4 | 2 | 2 | 1, 2 | multiplesNum = 1 | multiplesNum = 1 |  |
| Т5 | 2 | 2 | 2, 4 | multiplesNum = 2 | multiplesNum = 2 |  |

***Таблица 3 – Тест по методу черного ящика***

| **Критерии** | **Т1** | **Т2** | **Т3** | **Т4** | **Т5** |
| --- | --- | --- | --- | --- | --- |
| n > 0 (целое) | + |  | + | + | + |
| n <= 0 (целое) |  | + |  |  |  |
| n – не целое число или не число |  | + |  |  |  |
| К = 0 |  |  | + |  |  |
| К – вещественное число | + |  | + | + | + |
| К – не вещественное число или не число |  |  | + |  |  |
| – целое число | + |  |  | + | + |
| – не целое число или не число | + |  |  |  |  |
| multiplesNum = 0 | + |  |  |  |  |
| multiplesNum = 1 |  |  |  | + |  |
| multiplesNum > 1 |  |  |  |  | + |
| Неверный ввод | + | + | + |  |  |
| Невозможно определить число кратных чисел |  | + | + |  |  |

***Таблица 4 – Тест по методу белого ящика***

|  |  | **Т1** | **Т2** | **Т3** | **Т4** | **Т5** |
| --- | --- | --- | --- | --- | --- | --- |
| do{}while (!isRead) // ввод n | 1 раз | + |  | + | + | + |
| > 1 раза |  | + |  |  |  |
| if (!isRead) | Да |  | + |  |  |  |
| Нет | + | + | + | + | + |
| do{}while (!isRead) // ввод K | 1 раз | + |  |  | + | + |
| > 1 раза |  |  | + |  |  |
| if (!isRead) | Да |  |  | + |  |  |
| Нет | + |  | + | + | + |
| if (n > 0 && K != 0) | Да | + |  |  | + | + |
| Нет |  | + | + |  |  |
| for (int i = 0; i < n; i++) | 0 раз |  | + | + |  |  |
| > 0 раз | + |  |  | + | + |
| do{}while (!isRead) // ввод | 1 раз |  |  |  | + | + |
| > 1 раза | + |  |  |  |  |
| if (!isRead) | Да | + |  |  |  |  |
| Нет | + |  |  | + | + |
| if (curNum1 % K == 0) | Да |  |  |  | + | + |
| Нет | + |  |  |  |  |

**Задача №2 (21)**

**Задание:** Дана последовательность целых чисел, за которой следует 0. Найти сумму нечетных элементов этой последовательности.

1. **Анализ**

Исходные данные:

* , …, – последовательности элементов (целое число).

Результат: oddSum – сумма нечетных элементов последовательности (целое число).

Анализ решения: используется цикл с постусловием, т.к. неизвестно количество чисел в последовательности, а известно то, что последовательностью являются все введенные числа до первого нуля.

Составим таблицу анализа решения (таблица 5).

***Таблица 5 – Анализ решения***

| **Исходные данные** | **Классы** | **Ожидаемый результат** |
| --- | --- | --- |
|  | Целое число | oddSum |
| Не целое число или не число | Неверный ввод |

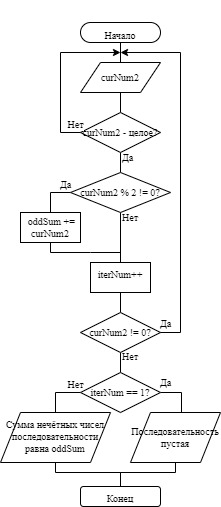
Классы входных данных:

1. – целое число. Результат: oddSum.
2. – не целое число или не число. Результат: [!] Неверный ввод.

Классы выходных данных:

1. Неверный ввод.
2. oddSum.
3. Пустая последовательность.
4. **Алгоритм**

Построим блок-схему для второго задания (Рисунок 2).



***Рисунок 2 – Блок-схема задания №2***

1. **Программа**

int oddSum = 0;

int curNum2 = 1; // Ввожу любое ненулевое число

int iterNum = 0;

bool isRead;

Console.WriteLine("\nЗАДАЧА №2 (21)\n");

Console.WriteLine("Вводите числа последовательности (типа int). Конец последовательности - число 0.");

Console.WriteLine("После каждого ввода нажимайте Enter: ");

do

{

do

{

string value = Console.ReadLine();

isRead = Int32.TryParse(value, out curNum2);

if (!isRead)

Console.WriteLine("[!] Неверный ввод");

} while (!isRead);

if (curNum2 % 2 != 0)

oddSum += curNum2;

iterNum++;

} while (curNum2 != 0);

if (iterNum == 1)

Console.WriteLine("[!] Последовательность пустая");

else

Console.WriteLine($"Сумма нечётных чисел последовательности равна {oddSum}");

1. **Тесты**

Составим таблицу тестов (таблица 6), в том числе по методу черного ящика (таблица 7) и белого ящика (таблица 8).

***Таблица 6 – Тесты***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** |  | **Полученный результат** | **Ожидаемый результат** | **Примечание** |
| Т1 | 2,5; “a”; 0 | Неверный ввод; Неверный ввод; Последовательность пустая | Неверный ввод; Неверный ввод; Последовательность пустая |  |
| Т2 | 1, 0 | oddSum = 1 | oddSum = 1 |  |
| Т3 | 1, 2, 0 | oddSum = 1 | oddSum = 1 |  |
| Т4 | -1, -2, -3, 0 | oddSum = -4 | oddSum = -4 |  |

***Таблица 7 – Тест по методу черного ящика***

| **Критерии** | **Т1** | **Т2** | **Т3** | **Т4** |
| --- | --- | --- | --- | --- |
| = 0 | **+** |  |  |  |
| – целое число (кроме 0) |  | + | + | + |
| – не целое число или не число | + |  |  |  |
| Неверный ввод | + |  |  |  |
| Последовательность пустая | + |  |  |  |

***Таблица 8 – Тест по методу белого ящика***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | **Т1** | **Т2** | **Т3** | **Т4** |
| do{}while (curNum2 != 0) | 1 раз | + |  |  |  |
| > 1 раза |  | + | + | + |
| do{}while (!isRead) // ввод для | 1 раз |  | + | + | + |
| > 1 раза | + |  |  |  |
| if (!isRead) | Да | + |  |  |  |
| Нет | + | + | + | + |
| if (iterNum == 1) | Да | + |  |  |  |
| Нет |  | + | + | + |

**Задача №3 (48)**

**Задание:** Найти сумму цифр в десятичном числе k.

1. **Анализ**

Исходные данные: k – десятичная дробь.

Результат: numSum – сумма чисел десятичной дроби.

Составим таблицу анализа решения (таблица 9).

***Таблица 9 – Анализ решения***

| **Исходные данные** | **Классы** | **Ожидаемый результат** |
| --- | --- | --- |
| k | k – десятичная дробь | numSum |
| k – не десятичная дробь или не число | Неверный ввод |

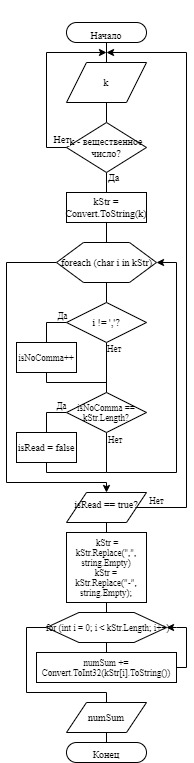
Классы входных данных:

1. k – десятичная дробь. Результат: numSum.
2. k – не десятичная дробь или не число. Результат: [!] Неверный ввод.

Классы выходных данных:

1. numSum.
2. Неверный ввод.
3. **Алгоритм**

Построим блок-схему для третьего задания (Рисунок 3).



***Рисунок 3 – Блок схема задания №3***

1. **Программа**

Console.WriteLine("\nЗАДАЧА №3 (48)\n");

Console.WriteLine("Введите десятичную дробь k:");

double k;

double numSum = 0;

string kStr = "";

bool isRead;

do

{

string value = Console.ReadLine();

isRead = Double.TryParse(value, out k);

if (!isRead)

{

Console.WriteLine("[!] Неверный ввод");

continue;

}

kStr = Convert.ToString(k);

int isNoComma = 0;

foreach (char i in kStr)

{

if (i != ',')

isNoComma++;

if (isNoComma == kStr.Length)

{

Console.WriteLine("[!] Введена не десятичная дробь");

isRead = false;

}

}

} while (!isRead) ;

kStr = kStr.Replace(",", string.Empty);

kStr = kStr.Replace("-", string.Empty);

for (int i = 0; i < kStr.Length; i++)

{

numSum += Convert.ToInt32(kStr[i].ToString());

}

Console.WriteLine($"Сумма десятичного числа k равна {numSum}");

1. **Тесты**

Составим таблицу тестов (таблица 10), в том числе по методу черного ящика (таблица 11) и белого ящика (таблица 12).

***Таблица 10 – Тесты***

| **№** | **k** | **Полученный результат** | **Ожидаемый результат** | **Примечание** |
| --- | --- | --- | --- | --- |
| Т1 | “a”; 3; 0,11 | Неверный ввод; Неверный ввод; numSum = 2 | Неверный ввод; Неверный ввод; numSum = 2 |  |
| Т2 | -0,12 | numSum = 3 | numSum = 3 |  |

***Таблица 11 – Тест по методу черного ящика***

|  |  |  |
| --- | --- | --- |
| **Критерии** | **Т1** | **Т2** |
| k – десятичная дробь | + | + |
| k – не десятичная дробь или не число | + |  |
| Неверный ввод | + |  |

***Таблица 12 – Тест по методу белого ящика***

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **Т1** | **Т2** |
| do{}while (!isRead) // ввод k | 1 раз |  | + |
| > 1 раза | + |  |
| if (!isRead) | Да | + |  |
| Нет | + | + |
| foreach (char i in kStr) | – | + | + |
| if (i != ',') | Да | + | + |
| Нет | + | + |
| if (isNoComma == kStr.Length) | Да | + |  |
| Нет | + | + |
| for (int i = 0; i < kStr.Length; i++) | – | + | + |