ФИЛИАЛ ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ

ВЫСШЕГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ «МЭИ»

в г. Смоленске

Кафедра вычислительной техники:

Отчет

По лабораторной работе № 4

Тема: «ТЕСТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ»

по курсу: «Технология программирования»

Студент: Иванов В.В.

Группа: ИВТ1-18

Преподаватель: Малашенкова И.В.

Вариант: 7

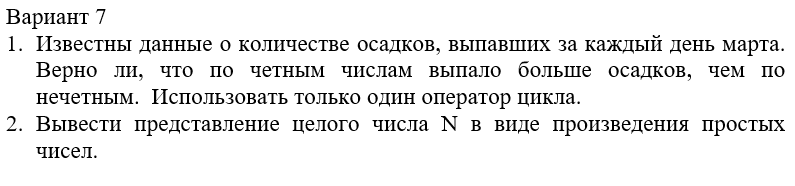
г. Смоленск ,2020

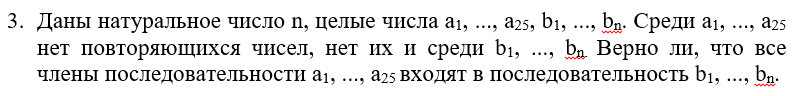
**Цель**:

Изучить методы тестирования программного обеспечения:

* Ручное тестирование программы;
* Структурное тестирование программы (тестирование базового пути, условий, циклов, потоков данных);
* Функциональное тестирование программы (анализ граничных значений, причинно-следственных связей, предположение об ошибке);

1. **Задание на выполнение:**





**2.** **Анализ задания**: в данной лабораторной работе необходимо предусмотреть проверку ввода данных (на вход принимаются только числа).

В 1-ом задании необходимо, используя один оператор цикла, узнать когда выпало больше осадков по чётным или нечётным числам.

Во 2-ом задании необходимо разложить число на произведение простых чисел.

В 3-ем задании необходимо узнать входит ли массив a в массив b учитывая, что в самих массивах нет повторяющихся элементов.

**Схема алгоритма**

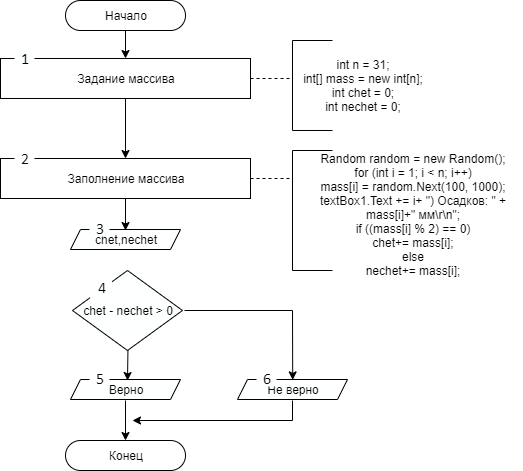


Рисунок 3.1 – Алгоритм для 1 задания

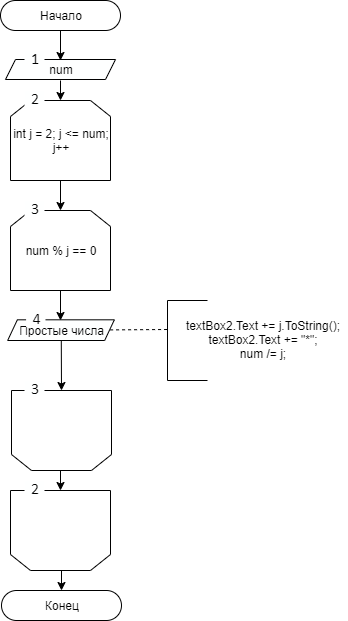


Рисунок 3.2 – Алгоритм для 2 задания

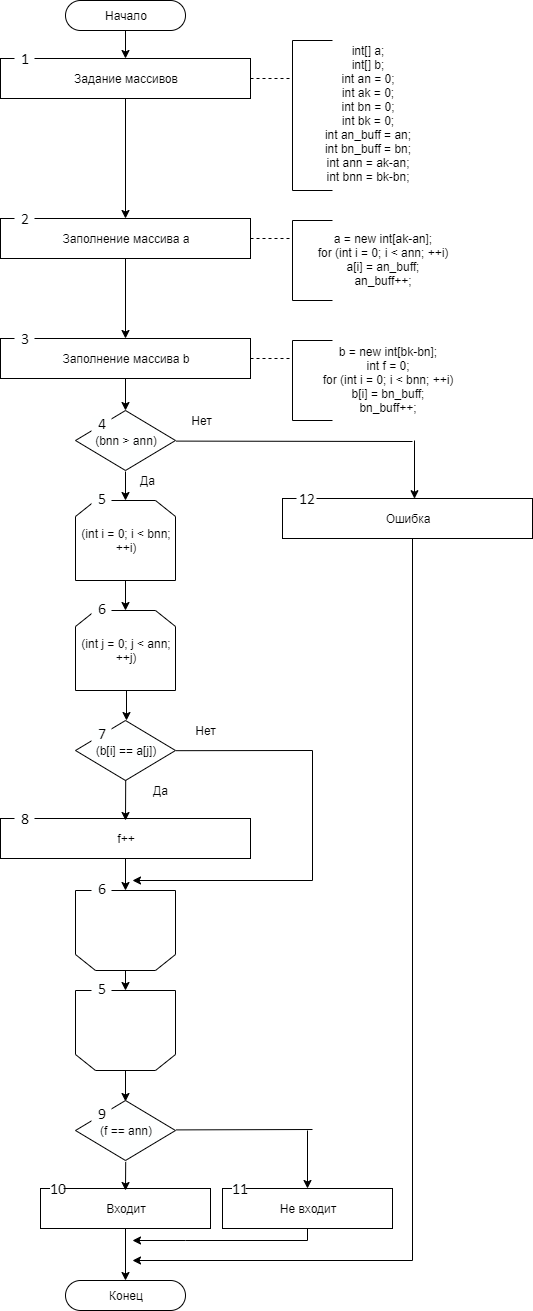


Рисунок 3.3 – Алгоритм для 3 задания

**4. Тестирование программы**

**4.1 Ручное тестирование**

Таблица 1 – Ручное тестирование

|  |  |  |
| --- | --- | --- |
| Набор данных | Вводимые данные | Результат |
| Корректный для 1 задания | ЫПтердоидя nsiongh v;iosergho; h кпшрпртвыршы jdfljslfjlОадшркп | .27 |
| Корректный для 2 задания | N=6; mas= { 5 6 4 3 2 7 } | 5 |
| Корректный для 3 задания | Нет входных данных | – |
| Некорректный для 1 задания | ПЕПЕРЕОПАЛИВоишврдашп  рыдщпомыжщупрвш. | 0 |
| Некорректный для 2 задания | b | Ошибка |
| Некорректный для 3 задания | Нет входных данных | – |

**4.2 Структурное тестирование**

1) Тестирование базового пути проводится по коду 2 задания



Рисунок 4.1 – Потоковый граф

Выполнение расчет циклической сложности:

1) E – кол-во дуг, а N – кол-во узлов;

2); p – кол-во предикатных узлов;

3) R– кол-во регионов. Достаточно 4 тестов.

Независимые Пути: 1) 1-2-3-4-5-8 (Сумма найдена)

2) 1-2-3-4-6-7-4-5-8

3) 1-8

4) 1-2-3-4-6-8(Сумма не найдена)

Таблица 2 – Тестирование базового пути

|  |  |  |
| --- | --- | --- |
| Тест | Ожидаемый результат | Полученный результат |
| 1 | Сумма найдена |  |
| 2 | Сумма найдена |  |
| 3 | Неверные данные |  |
| 4 | Сумма не найдена |  |

2) Тестирование условия. При проверке работы программы должна появиться надпись «нет результата» если сумма не была найдена, окно ошибки если введены некорректные данные, результат(наименьшая! сумма).

Таблица 3 – Тестирование условий

|  |  |  |
| --- | --- | --- |
| Исходные данные | Ожидаемый результат | Полученный результат |
| 5 1 1 8 8 8 | Нет результата | Нет результата |
| 8 1 2 7 2 4 | 4 | 4 |
| two | Ошибка | Ошибка |

Программа работает корректно.

3) Тестирование циклов. Проверим работу вложенных циклов, считающих суммы элементов. Тест 1 – проверка на ввод некорректных данных. Тест 2 – проверка расчета сумм по 2элемента. Тест 3 –проверка счета элементов больше 2.

Таблица 4 – Тестирование циклов

|  |  |  |  |
| --- | --- | --- | --- |
| Тест | Цель | Ожидаемый результат | Полученный результат |
| 1 | Проверка на корректность | Ошибка |  |
| 2 | Проверка вложенного цикла | Вывод суммы 2х элементов |  |
| 3 | Проверка вложенного цикла | Вывод суммы элементов больше 2 |  |

**4.3 Функциональное тестирование**

1) Разбиение на классы эквивалентности. Вводимы данные можно разделить на 2 класса эквивалентности: класс допустимых значений(неотрицательные целые числа); класс недопустимых значений(отрицательные числа, специальные символы, любые буквы).

Таблица 5 – Разбиение на классы эквивалентности

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Тест | Цель | Исходные данные | Ожидаемый результат | Полученный результат |
| 1 | Допустимые значения | 6 | Успешное выполнение |  |
| 2 | Недопустимые значения | -2 | Ошибка |  |

2) Анализ граничных значений. Граничные значения для переменных типа int, в условиях данной задачи, значения будут от 3 до 2147483647.

Таблица 6 – Анализ граничных значений

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Тест | Цель | Исходные данные | Ожидаемый результат | Полученный результат |
| 1 | Допустимые значения | 5 | Успешное выполнение |  |
| 2 | Недопустимые значения | -7 | Ошибка |  |

3) Анализ причинно-следственных связей. Причинами будут классы введенных данных, вторичные причины – результат выполнения условия. Результаты занесены в таблицу7.



Рисунок 4.1 –граф причинно-следственных связей

1 – esp1;

2 – exp2;

11 – все условия выполняются;

12 – не все условия выполнены;

111 – вывод результат;

112 –нет результата.

Таблица 7 – Анализ причинно-следственных связей

|  |  |  |  |
| --- | --- | --- | --- |
| Причины | 1 | 1 | 0 |
| 2 | 0 | 1 |
| Вторичные причины | 11 | 1 | 0 |
| 12 | 0 | 1 |
| Следствие | 111 | 1 | 0 |
| 112 | 0 | 1 |

Таблица 8 – Тестирование причинно-следственных связей

|  |  |  |
| --- | --- | --- |
| Тест | Ожидаемый результат | Полученный результат |
| 1 | Сумма найдена |  |
| 2 | Неверные данные |  |
| 3 | Сумма не найдена |  |

**5. Экранные формы**

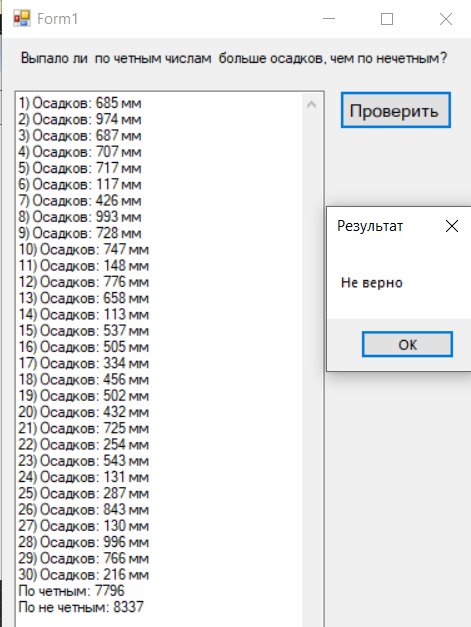


Рисунок 5.1 – Экранная форма для 1 задания

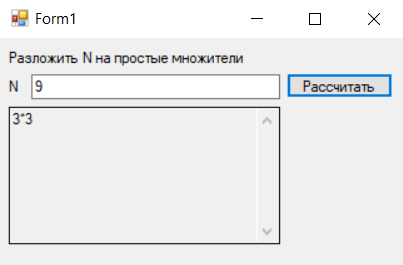


Рисунок 5.2 – Экранная форма для задания 2

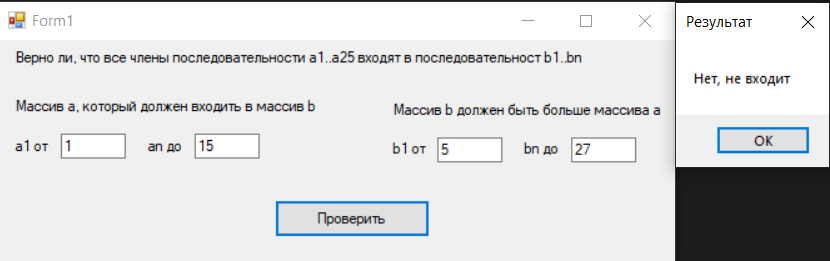


Рисунок 5.3 – Экранная форма для задания 3

**Вывод:** изучены методы ручного, структурного и функционального тестирования программного обеспечения.

**Приложение А. Программный код**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Лр4.\_1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Button1\_Click(object sender, EventArgs e)

{

textBox1.Clear();

int n = 31;

int[] mass = new int[n];

int chet = 0;

int nechet = 0;

Random random = new Random();

for (int i = 1; i < n; i++)

{

mass[i] = random.Next(100, 1000);

textBox1.Text += i+ ") Осадков: " + mass[i]+" мм\r\n";

if ((mass[i] % 2) == 0)

chet+= mass[i];

else

nechet+= mass[i];

}

textBox1.Text += "По четным: "+ chet + "\r\n";

textBox1.Text += "По не четным: "+ nechet;

if (chet - nechet > 0)

MessageBox.Show("Верно", "Результат");

else

MessageBox.Show("Не верно", "Результат");

}

}

}

**Приложение Б.**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Лр4.\_2

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Button1\_Click(object sender, EventArgs e)

{

textBox2.Clear();

int num = int.Parse(textBox1.Text);

for (int j = 2; j <= num; j++)

{

while (num % j == 0)

{

textBox2.Text += j.ToString();

textBox2.Text += "\*";

num /= j;

}

}

textBox2.Text = textBox2.Text.Substring(0, textBox2.Text.Length - 1);

}

}

}

**Приложение В.**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Лр4.\_3

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Button1\_Click(object sender, EventArgs e)

{

int[] a;

int[] b;

int an = 0;

int ak = 0;

int bn = 0;

int bk = 0;

bn = Convert.ToInt32(tbBn.Text);

bk = Convert.ToInt32(tbBk.Text);

an = Convert.ToInt32(tbAn.Text);

ak = Convert.ToInt32(tbAk.Text);

int an\_buff = an;

int bn\_buff = bn;

int ann = ak-an;

int bnn = bk-bn;

a = new int[ak-an];

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

{

a[i] = an\_buff;

an\_buff++;

}

b = new int[bk-bn];

int f = 0;

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

{

b[i] = bn\_buff;

bn\_buff++;

}

if (bnn > ann)

{

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

{

for (int j = 0; j < ann; ++j)

{

if (b[i] == a[j])

f++;

}

}

if (f == ann)

MessageBox.Show(" Да, входит", "Результат");

else

MessageBox.Show(" Нет, не входит", "Результат");

}

else

MessageBox.Show(" b должно быть больше a");

}

}

}