МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 9

# по дисциплине «Объектно-ориентированное проектирование и программирование»

на тему: «Обработка исключительных ситуаций»

Выполнил студент гр. ИТИ-21:  
Буякевич Д.О.  
 Принял преподаватель:  
 Чернявская А.Э.

Гомель 2022

**Цель:** научиться обрабатывать ошибки в программе, чтобы программа не имела мест для выброса из программы.

На рисунке 1 показаны требования к лабораторной работе.

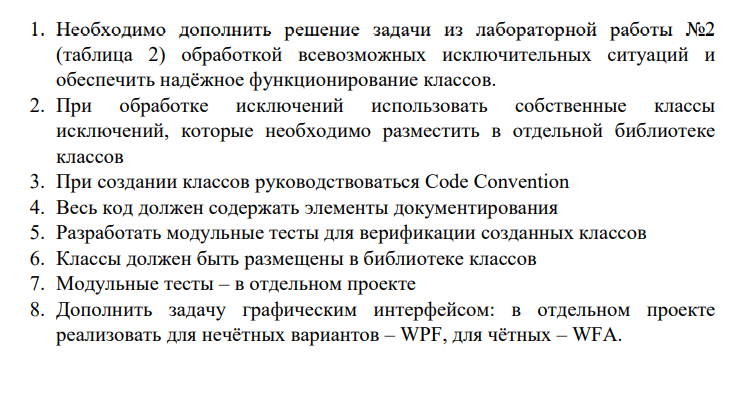


Рисунок 1 – Требования

На рисунках 2-4 показана работа программы в WFA приложении.

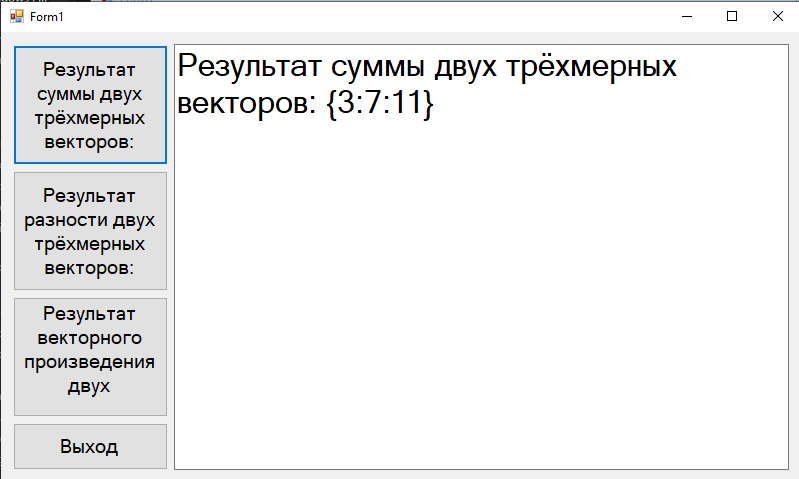


Рисунок 2 – Пункт 1

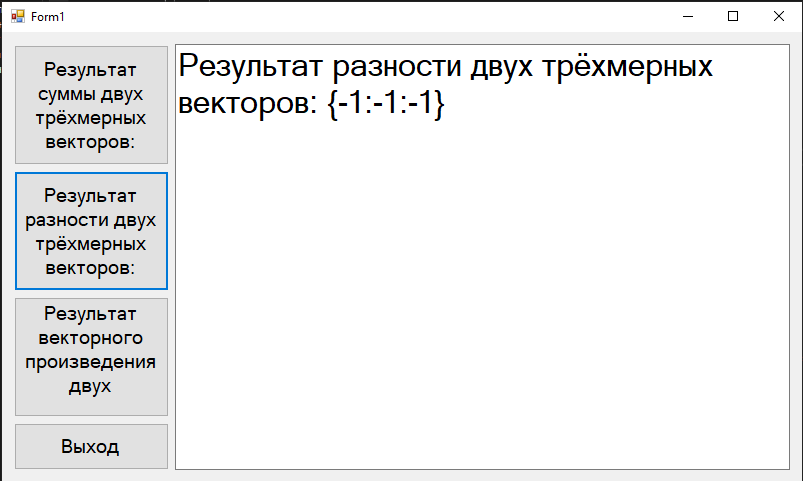


Рисунок 3 – Пункт 2

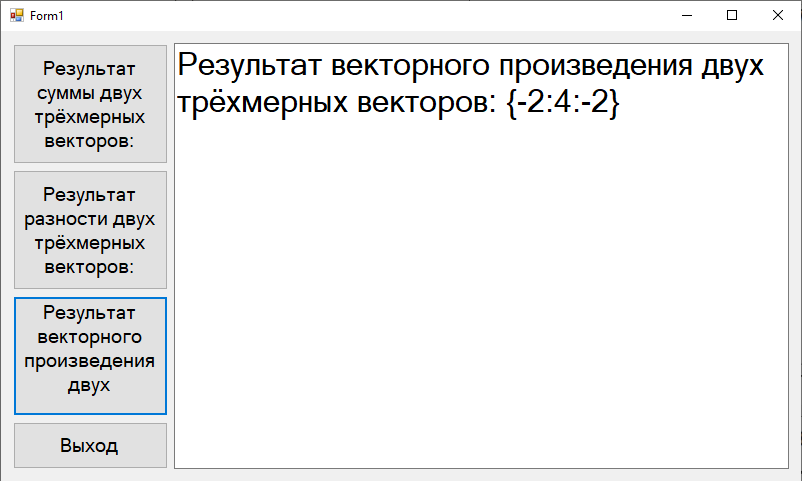


Рисунок 4 – Пункт 3

**Вывод**: научились находить слабые места в программе и обрабатывать их.

**ПРИЛОЖЕНИЕ А**

**ЛИСТИНГ ПРОГРАММЫ**

**Program.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab\_09

{

internal class Program

{

static void Main(string[] args)

{

Vector vector1;

Vector vector2;

vector1 = new Vector() { };

vector2 = new Vector() {VectorX = 2,VectorY = 4, VectorZ = 6 };

while (true)

{

try

{

vector1.VectorX = Convert.ToInt32(Console.ReadLine());

vector1.VectorY = Convert.ToInt32(Console.ReadLine());

vector1.VectorZ = Convert.ToInt32(Console.ReadLine());

break;

}

catch (SystemException ex)

{

Console.WriteLine($"Ошибка: {ex.Message}");

}

}

Vector vectorAddition = vector1 + vector2;

Vector vectorSubtraction = vector1 - vector2;

Vector vectorVectorMultiplication = vector1 \* vector2;

int choice = 0;

try

{

while (choice != 4)

{

Console.WriteLine("====================");

Console.WriteLine($"Векторы равны: {{{vector1.VectorX}:{vector1.VectorY}:{vector1.VectorZ}}} и {{{vector2.VectorX}:{vector2.VectorY}:{vector2.VectorZ}}} ");

Console.WriteLine("====================");

Console.WriteLine("1)Сумма векторов");

Console.WriteLine("2)Разность векторов");

Console.WriteLine("3)Произведение векторов");

Console.WriteLine("4)Выход");

Console.WriteLine();

Console.WriteLine("====================");

Console.Write("Введите пункт меню: ");

try

{

choice = Convert.ToInt32(Console.ReadLine());

}

catch

{

Console.WriteLine("Введенная вами строка имела неверный формат!");

Console.ReadKey();

Console.Clear();

continue;

}

Console.ReadKey();

switch (choice)

{

case 1:

Console.WriteLine($"Результат суммы двух трёхмерных векторов: {{{vectorAddition.VectorX}:{vectorAddition.VectorY}:{vectorAddition.VectorZ}}}");

Console.ReadKey();

Console.Clear();

break;

case 2:

Console.WriteLine($"Результат разности двух трёхмерных векторов: {{{vectorSubtraction.VectorX}:{vectorSubtraction.VectorY}:{vectorSubtraction.VectorZ}}}");

Console.ReadKey();

Console.Clear();

break;

case 3:

Console.WriteLine($"Результат векторного произведения двух трёхмерных векторов: {{{vectorVectorMultiplication.VectorX}:{vectorVectorMultiplication.VectorY}:{vectorVectorMultiplication.VectorZ}}}");

Console.ReadKey();

Console.Clear();

break;

default:

choice = 4;

Console.ReadKey();

Console.Clear();

Console.WriteLine("Выход!");

break;

}

}

}

catch (VectorException)

{

Console.WriteLine("Вы ввели не коректное значение");

}

}

class VectorException : ArgumentException

{

public VectorException(string message) : base(message)

{

}

}

class VectorExceptionWithValue : ArgumentException

{

public double Value { get; }

public VectorExceptionWithValue(string message, double value) : base(message)

{

Value = value;

}

}

}

}

**Vector.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab\_09

{

public class Vector

{

public double VectorX { get; set; }

public double VectorY { get; set; }

public double VectorZ { get; set; }

public static Vector operator +(Vector vector1, Vector vector2)

{

return new Vector

{

VectorX = vector1.VectorX + vector2.VectorX,

VectorY = vector1.VectorY + vector2.VectorY,

VectorZ = vector1.VectorZ + vector2.VectorZ

};

}

public static Vector operator -(Vector vector1, Vector vector2)

{

{

return new Vector

{

VectorX = vector1.VectorX - vector2.VectorX,

VectorY = vector1.VectorY - vector2.VectorY,

VectorZ = vector1.VectorZ - vector2.VectorZ

};

}

}

public static Vector operator \*(Vector vector1, Vector vector2)

{

return new Vector

{

VectorX = vector1.VectorY \* vector2.VectorZ - vector1.VectorZ \* vector2.VectorY,

VectorY = -(vector1.VectorX \* vector2.VectorZ - vector1.VectorZ \* vector2.VectorX),

VectorZ = vector1.VectorX \* vector2.VectorY - vector1.VectorY \* vector2.VectorX

};

}

class VectorExceptionWithValue : ArgumentException

{

public double Value { get; }

public VectorExceptionWithValue(string message, double value) : base(message)

{

Value = value;

}

}

}

}