# 2 РАБОЧИЙ ПРОЕКТ

## 2.1 Общие сведения о работе программы

В результате выполнения курсовой работы была создана программа, позволяющая проводить автоматические тестирования на ввод логических. В программе реализованы алгоритмы генерации логических выражений, генерация необходимой области для логического выражения, сравнения введенного пользователем и сгенерированного выражения. Программа разработана на языке программирования высокого уровня *C#* с использованием средств .NET Framework версии 3,5 и системы Windows Presentation Foundation (WPF) в интегрированной среде *Microsoft Visual Studio 2013* и работает под управлением операционных систем семейства Windows.

1. 1. **Функциональное назначение**

Программа обеспечивает следующие функциональные возможности:

* проведение тестирования на ввод логических выражений;
* проведение тестирования;
* вывод результатов тестирования.

**Функциональные ограничения**

Количество заданий в базе не должно быть меньше 10.

* 1. **Инсталляция и выполнение**

Для установки программного продукта необходимо запустить файл LogicSetup.exe и следовать подсказкам системы. После установки на рабочем столе появится ярлык «Проверка знания булевых выражений» (если было выбрано создание ярлыка) либо в установочной папке запустить файл «Проверка знания булевых выражений».

* 1. **Общий алгоритм работы программного продукта**

Блок-схема, демонстрирующая общий алгоритм работы программы, представлена в приложении на рисунке П.1.

В программе реализованы следующие модули:

1. MainWindow.xaml.
2. MainWindow.xaml.cs.
3. Test.xaml.
4. Test.xaml.cs.
5. Result.xaml.
6. Result.xaml.cs.
7. RPN.cs.
8. TestChec.cs.
9. from\_XML\_to\_Task.cs.
10. Task.cs.
11. Primitive.cs.
12. polyline.cs.
13. poligon.cs.
14. line.cs.
15. ellips.cs.
16. lable.cs
17. point.cs

Функциональное назначение модулей представлено в таблицах 2.1-2.7.

Модуль MainWindow*.xaml.cs* содержит класс MainWindow, который реализует логику работы модуля MainWindow*.xaml,* описывающего процесс навигации в главном меню.

Модуль *Test.xaml.cs* содержит класс *Test*, который реализует логику работы модуля *Test.xaml,* описывающего процесс прохождения тестирования.

В таблице 2.1 приведены методы класса *Test*.

Таблица 2.1 – Описание класса *Test*

|  |  |
| --- | --- |
| **Прототип** | **Назначение** |
| **public void drawXOY(Canvas draw)** | Рисует оси координат. |

Модуль Result*.cs* содержит класс Result, которыйреализует вывод результата тестирования на экран с последующим сохранением его в файл статистики.

В таблице 2.2 приведены методы класса Result.

Таблица 2.2 – Описание методов класса Result

|  |  |
| --- | --- |
| **Прототип** | **Назначение** |
| **private void addStatistic()** | Добавляет статистику тестирования в файл статистики. |

Модуль RPN*.cs* содержит класс RPN, которыйреализует перевод логического выражения из инфиксной записи в постфиксную.

В таблице 2.3 приведены методы класса RPN.

Таблица 2.3 – Описание методов класса RPN

|  |  |
| --- | --- |
| **Прототип** | **Назначение** |
| **private bool isANumber(string value)** | Проверяет символ на принадлежность к целым числам. |
| **private string retEndDel(string returnString)** | Возвращает последний символ строки и удаляет его. |
| **public string inPostfix(string IN)** | Переводит строку в постфиксную запись. |

Модуль *Task.cs* содержит класс *Task*, которыйпредставляет из себя задание для тестирования.

В таблице 2.4 приведены методы класса *Task*.

Таблица 2.4 – Описание методов класса *Task*

|  |  |
| --- | --- |
| **Прототип** | **Назначение** |
| **public void Draw(Canvas myCanvas)** | Выводит на экран задания. |

Модуль from\_XML\_to\_Task*.cs* содержит класс from\_XML\_to\_Task, которыйреализует разбор базы шаблонов на задания.

В таблице 2.5 приведены методы класса from\_XML\_to\_Task.

Таблица 2.5 – Описание методов класса from\_XML\_to\_Task

|  |  |
| --- | --- |
| **Прототип** | **Назначение** |
| **public List<Task> Decode()** | Декодирование файла с базой шаблонов. |
| **private bool isInCircle(int x, int y, int x0, int y0, int r)** | Проверка на принадлежность точки кругу. |

Модуль TestChec*.cs* содержит класс TestChec, которыйреализует проверку введенных пользователем ответов.

В таблице 2.6 приведены методы класса TestChec.

Таблица 2.6 – Описание методов класса TestChec

|  |  |
| --- | --- |
| **Прототип** | **Назначение** |
| **public bool getChek(string Etal, string User)** | Сверяет строку эталонного решения с ответом пользователя. |

­­­­­­­­­­­­­­

Модуль Primitive*.cs* содержит класс Primitive, описывающий модель примитива.

В таблице 2.7 приведены методы класса Primitive.

Таблица 2.7 – Описание методов класса Primitive

|  |  |
| --- | --- |
| **Прототип** | **Назначение** |
| **public virtual void Draw(Canvas myCanvas)** | Выводит на экран примитив. |
| **public virtual void Draw(Canvas myCanvas, bool lbl)** | Выводит на экран примитив с подсказкой. |
| **public virtual void aDraw(ref bool[,] points)** | Выделяет область выше линии. |
| **public virtual void bDraw(ref bool[,] points)** | Выделяет область ниже линии. |
| **public virtual void iDraw(ref bool[,] points)** | Выделяет область внутри замкнутого примитива. |
| **public virtual void uDraw(ref bool[,] points)** | Выделяет область снаружи замкнутого примитива. |
| **public virtual bool Included(Point point)** | Проверяет принадлежность точки примитиву. |

Продолжение таблицы 2.7.

|  |  |
| --- | --- |
| **public virtual bool Above(Point point)** | Проверяет принадлежность точки области, расположенной выше линии. |
| **public virtual bool Below(Point point)** | Проверяет принадлежность точки области, расположенной ниже линии. |
| **public virtual string Above()** | Возвращает уравнение области, расположенной выше линии. |
| **public virtual string Below()** | Возвращает уравнение области, расположенной ниже линии. |
| **public virtual string Included()** | Возвращает уравнение области, расположенной внутри замкнутого примитива. |
| **public virtual string unIncluded()** | Возвращает уравнение области, расположенной вне замкнутого примитива. |

Модуль *line.cs, label.cs, polyline.cs, polygon.cs, ellips.cs, point.cs* наследуются от класса Primitive и используют методы, объявленные в нем.

* 1. **Разработанные меню и интерфейсы**

Главное окно программы и меню показаны на рисунке 2.1.

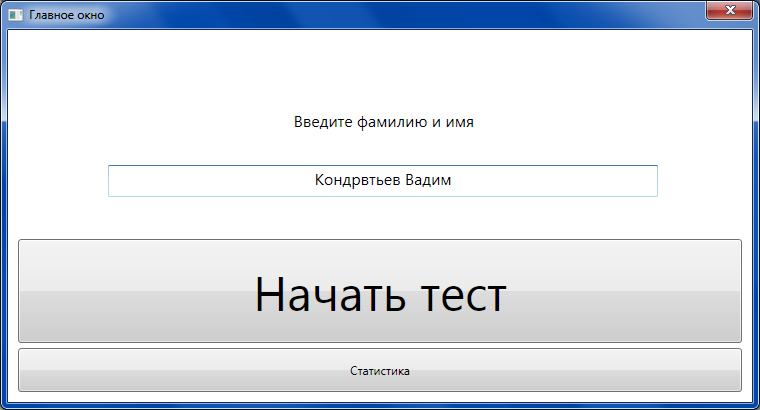


Рисунок 2.1 – Основное окно Semantic IDE и меню программы на основной ленте

Основное меню программы содержит следующие кнопки:

1. кнопка перехода в режим тестирования;
2. кнопка перехода к окну статистики.

Чтобы начать тестирование, необходимо нажать кнопку «Начать тест». Чтобы перейти к окну статистики, необходимо нажать кнопку «Статистика».

* + 1. **Интерфейс окна режима тестирования**

После нажатия кнопки «Начать тестирование» пользователь попадает в окно режима тестирования, которое продемонстрированно на рисунке 2.2.

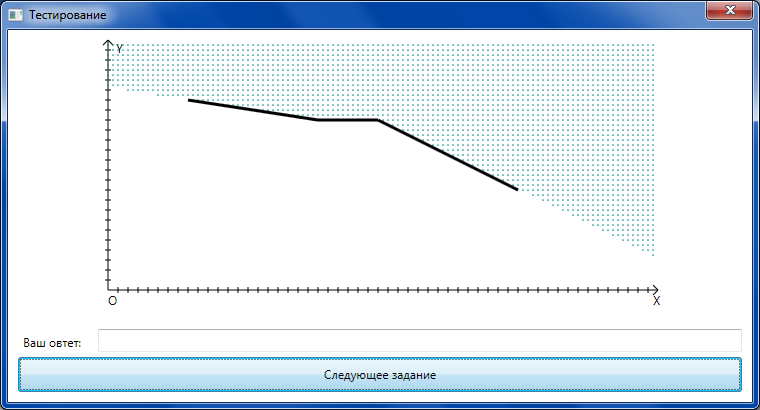


Рисунок 2.2 – Интерфейс окна режима тестирования

Окно режима тестирования содержит графическое представление области, подсказки по вводу функций, поле ввода ответа и кнопку следующего задания.

После завершения теста появляется окно результата, показанное на рисунке 2.3.

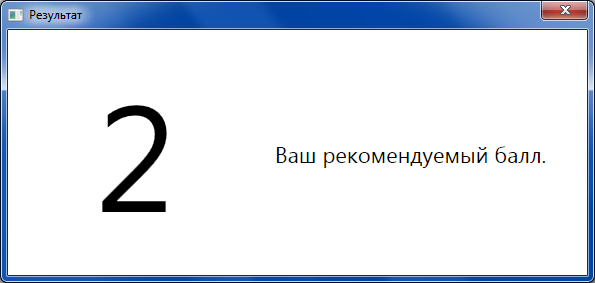


Рисунок 2.3 – Окно результата тестирования

Окно статистики тестирования показано на рисунке 2.4.

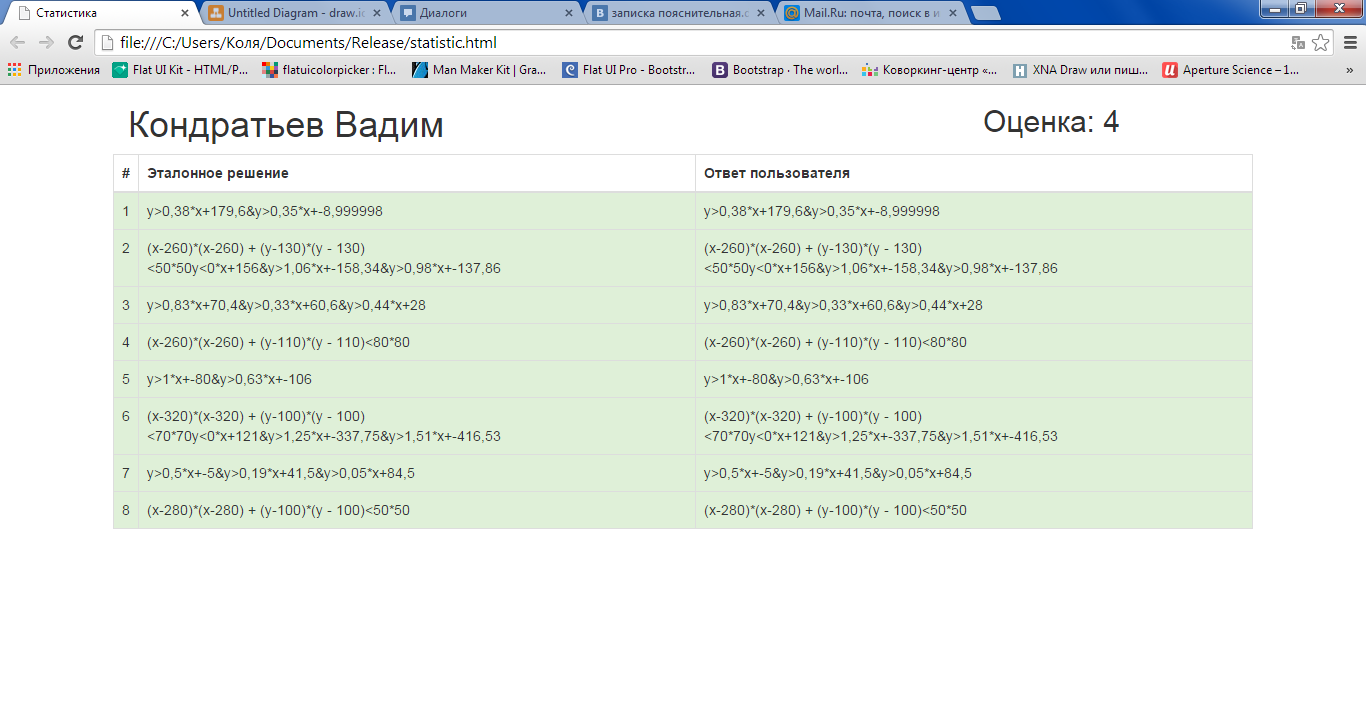


Рисунок 2.4 – Окно результатов тестирования с информацией о пройденных заданиях

# ПРИЛОЖЕНИЕ

**Блок-схемы и рисунки**



Рисунок П.1 – Блок-схема, демонстрирующая общий алгоритм работы программы