|  |  |
| --- | --- |
|  | ***Федеральное агентство по рыболовству***  ***Федеральное государственное бюджетное образовательное учреждение***  ***высшего профессионального образования***  ***«Астраханский государственный технический университет»***  **Разработка и предоставление образовательных услуг в области среднего, высшего и дополнительного профессионального образования, послевузовского образования; воспитательная и научно-исследовательская работа сертифицированы DQS и ГОСТ Р по ISO 9001:2008** |

**ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И КОММУНИКАЦИЙ**

КАФЕДРА «АВТОМАТИЗИРОВАННЫЕ СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ»

ДОПУЩЕНО К ЗАЩИТЕ

Заведующий кафедрой, доцент

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ С.В. Белов

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2014 г.

От \_\_\_\_\_\_\_\_\_\_ № \_\_\_\_\_\_\_\_\_\_\_

**КУРСОВАЯ РАБОТА**

по дисциплине

**«Объектно-ориентированное программирование»**

на тему:

**«Генерация однотипных вариантов заданий»**

КР-230100.62-Б.3.Б.8-20122028

Студент гр. ДИНР-21

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сафрыгин Н.Н..  
«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_2014 г.

Руководитель

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Лаптев В.В.  
«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_2014 г.

Защищено с оценкой  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Руководитель \_\_\_\_\_\_ Лаптев В.В.

Член комиссии \_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_2014 г.

**АСТРАХАНЬ – 2014**

**ФЕДЕРАЛЬНОЕ АГЕНТСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ ПО РЫБОЛОВСТВУ**

**АСТРАХАНСКИЙ ГОСУДАРСТВЕННЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

|  |  |
| --- | --- |
| **УТВЕРЖДАЮ** | Кафедра «Автоматизированные системы  обработки информации и управления» |
| Заведующий кафедрой  к.т.н., доцент  С.В. Белов\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2014г. |

**ЗАДАНИЕ**

**на выполнение курсовой работы**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Студент | | **Сафрыгин Николай Николаевич** | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Группа | | **ДИНР-21** | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Дисциплина | | | **«Объектно-ориентированное программирование»** | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Тема курсовой работы | | | | | **«Тестирование учащихся на умение составлять логические** | | | | | | | | | | | | | | | | | | | | | | | | |
| **выражения»** | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Дата получения задания | | | | | | « | 11 | » | | февраля | | | | | | | | | | | 201 | | 4 | г. | | | | | |
| Срок представления студентом КР на кафедру | | | | | | | | | | | | | | | « | | |  | | | » |  | | | 201 | | 4 | г. | |
| Руководитель | | | | **доцент** | | | | |  | |  | | | | |  | | | **Лаптев Валерий Викторович** | | | | | | | | | | |
|  | | | | должность, степень, звание | | | | |  | | Подпись | | | | |  | | | ФИО | | | | | | | | | | |
|  | | | | | | | | | | | | | « |  | | | » | | |  | | | | | | 201 | | 4 | г. | |
| Студент | | | |  | | | | | | | |  | **Сафрыгин Николай Николаевич** | | | | | | | | | | | | | | | | |
|  | | | | подпись | | | | | | | |  | ФИО | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | « |  | | | » | | |  | | | | | | 201 | | 4 | г. | |
| **Задачи**  Создать приложение, предоставляющее возможность | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | генерация логических выражений и геометрической области по этим выражениям | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | проведения тестирования по сгенерированным выражениям | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |  | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| **Список рекомендуемой литературы** | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | Петцольд Ч. MicrosoftWindows Presentation Foundation: — M.: Издательство «Русская | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | Редакция»; СПб.: Питер, 2008. — 944 с. | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | Мак-Дональд Мэтью. WPF: Windows Presentation Foundation в .NET 3.5 с примерами на | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | C# 2008 для профессионалов, 2-е издание: Пер. с англ. — М. : ООО “И.Д. Вильямс”, | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | 2008. — 928 с. | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | НэшТрей. C# 2008: ускоренный курс для профессионалов. : Пер. с англ. – М. : | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | ООО «И.Д. Вильямс». 2008. – 576 с. | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

|  |  |
| --- | --- |
| **УТВЕРЖДАЮ** | К заданию  на курсовую работу |
| Заведующий кафедрой  к.т.н., доцент  С.В. Белов\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20 \_\_\_г. |

**КАЛЕНДАРНЫЙ ГРАФИК**

подготовки выпускной курсовой работы

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| №  п/п | Разделы, темы  и их содержание,  графический материал | По плану | | Фактически | | Отметка руководи-теля о вы-полнении |
| Дата | Объем в % | Дата | Объем в % |
| 1 | Техническое задание | 11.09.2014 | 5 |  |  |  |
| 2 | Разработка модели,  проектирование системы | 11.03.2014 | 35 |  |  |  |
| 3 | Программная реализация системы | 08.04.2014 | 65 |  |  |  |
| 4 | Тестирование и отладка системы | 22.04.2014 | 80 |  |  |  |
| 5 | Оформление документации, подготовка презентации и доклада | 13.05.2014 | 100 |  |  |  |

Руководитель курсовой работы

\_\_\_\_доцент Лаптев В.В.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(ученая степень, звание, фамилия, подпись)

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

Студент\_Сафрыгин Н.Н.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилия, инициалы, подпись)

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20 \_\_\_г.

**АННОТАЦИЯ**

с.30, рис.6, табл.14, прил.2

Разработан программный продукт, предоставляющий возможность генерировать некую геометрическую область, генерировать логическое выражение, удовлетворяющее этой области, компоновать область и выражение в задачи и проводить тест по этим задачам. Программный продукт хранится на CD-диске вместе с файлами исходного кода и презентацией.

СОДЕРЖАНИЕ

[**ВВЕДЕНИЕ 6**](#_Toc403691056)

[**1. ТЕХНИЧЕСКИЙ ПРОЕКТ 7**](#_Toc403691057)

[1.1 Описание предметной области 7](#_Toc403691058)

[**1.1.1 Основные теоретические сведения 7**](#_Toc403691059)

[1.2 Технология обработки информации 11](#_Toc403691060)

[**1.2.1 Описание вариантов использования 11**](#_Toc403691061)

[**1.2.2 Тестирование 11**](#_Toc403691062)

[**1.2.3 Алгоритм обработки базы с шаблонами 12**](#_Toc403691063)

[**1.2.4 Алгоритм подготовки базы заданий для тестирования 12**](#_Toc403691064)

[**1.2.5 Алгоритм вывода задания на экран 12**](#_Toc403691065)

[**1.2.6 Алгоритм проверки ответа тестируемого 12**](#_Toc403691066)

[1.3 Входные и выходные данные 13](#_Toc403691067)

[1.4 Требования к техническому и программному обеспечению 13](#_Toc403691068)

[**2. РАБОЧИЙ ПРОЕКТ 14**](#_Toc403691069)

[2.1. Общие сведения о работе программы 14](#_Toc403691070)

[2.2 Функциональное назначение 14](#_Toc403691074)

[2.3 Инсталляция и выполнение 14](#_Toc403691075)

[2.4 Общий алгоритм работы программного продукта 15](#_Toc403691076)

[2.5 Разработанные меню и интерфейсы 20](#_Toc403691077)

[**3 ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ 24**](#_Toc403691079)

[**4 ЛИТЕРАТУРА 26**](#_Toc403691080)

[**ПРИЛОЖЕНИЕ 1 27**](#_Toc403691081)

[**ПРИЛОЖЕНИЕ 2 28**](#_Toc403691082)

# ВВЕДЕНИЕ

Умение правильно составить логическое выражение для условного оператора является обязательным для программиста. Задачи на проверку принадлежности точки некоторой области на плоскости позволяют проверить умение правильно составлять логические выражения и использовать в них различные сочетания арифметических выражений и логических операторов, что является немаловажным для любого начинающего программиста. Подобные задачи являются заданием С1 в ЕГЭ по информатике, но количество этих задач ограничено. Возникает необходимость иметь большое количество подобных заданий. Решением этой проблемы может стать создание инструмента, позволяющего некоторым образом генерировать область на плоскости.

Целью разрабатываемого продукта является автоматизация проверки способностей учащихся составлять сложные логические выражения для задач о принадлежности точки заданной области.

Назначением конечного продукта является упрощение проверки студентов, автоматизированный сбор результатов тестирования для подсчета статистики.

# ТЕХНИЧЕСКИЙ ПРОЕКТ

## Описание предметной области

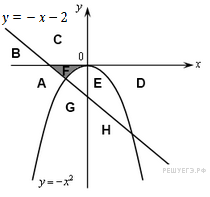
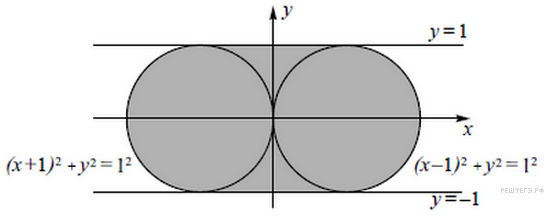
### Основные теоретические сведения

При изучении курса информатики, одним из важных разделов является умение правильно работать с логическими операциями, которые в свою очередь образуют логические выражения.

Умение правильно составлять подобные выражения является важным критерием для будущих программистов.

В экзаменационных задачах встречаются несколько разных видов: проверка на принадлежность точки заданным интервалам, проверка на принадлежность точки заданной области на плоскости и проверка на корректность ввода данных.

В разрабатываемом продукте будут реализованы задачи на принадлежность точки заданной области на плоскости. Примеры задач из части С ЕГЭ по информатике представлены на рисунке 1.1.



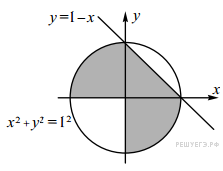
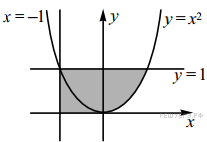


Рисунок 1.1 – Примеры задач из ЕГЭ по информатике

В ЕГЭ требуется составить программный код, который принимает данные (координаты точки) и проверяет ее на принадлежность указанной области. В данном программном продукте необходимо будет указать только логическое выражение, необходимое для проверки.

Логическое выражение в программировании — конструкция языка программирования, результатом вычисления которой является «истина» или «ложь».

В логическом выражении присутствуют операции сравнения, необходимые для выяснений значений логического выражения. Операции сравнения для логических выражений представлены в таблице 1.1.

Таблица 1.1 – Логические операции сравнения

|  |  |
| --- | --- |
| **Значение операции** | **Представление в программе** |
| Равно | == |
| Не равно | != |
| Больше | > |
| Меньше | < |
| Больше или равно | >= |
| Меньше или равно | <= |

Логические выражения, составленные из нескольких частей с операциями сравнения, называются сложными. Для сложных логических выражений характерны операции объединения. Логические операции объединения представлены в таблице 1.2.

Таблица 1.2 – Логические операции

|  |  |
| --- | --- |
| **Значение операции** | **Знак операции** |
| Логическое «И» | Выражение истинно, если оба операнда истинны. |
| Логическое «ИЛИ» | Выражение истинно, если хотя бы один операнд истинен. |
| Логическое «НЕ» (отрицание) | Меняет значение выражения с истинного на ложное, и наоборот. |

Операции в логическом выражении также подчиняются законам приоритета. Наивысший приоритет имеют выражения в скобках. Из операций, приведенных выше, наивысшим приоритетом обладает операция «НЕ», далее идет операция «И» и наименьший приоритет имеет операция «ИЛИ».

При написании логических выражений на языках высокого уровня компилятор данного языка проверяет строку на корректность ввода (не нарушены правила расстановки скобок, используются только те операции, которые заявлены как логические) и выполняют подстановку всех переменных и констант в булево выражение, что позволяет проверить правильность составления выражения на тестовом наборе точек. Также присутствует возможность переставлять местами операнды, меняя логические и арифметические знаки.

Сложность данного метода состоит в том, что проверять логическое выражение в рамках разрабатываемого продукта является затруднительным и ресурсоемким

Поэтому было принято решение, что проверка ответа будет производиться с помощью преобразования выражения в обратную польскую запись, что избавит от скобок, которые могут быть расставлены в местах, не требующих повышения приоритета.

Обратная польская запись - форма записи математических выражений, в которой операнды расположены перед знаками операций. Также именуется как обратная польская запись, обратная бесскобочная запись (ОБЗ), постфиксная нотация, бесскобочная символика Лукашевича, польская инверсная запись, ПОЛИЗ. Примером обратной польской записи является выражение , что в обычном представлении имеет вид .

Ввиду того, что обратная польская запись представлена строкой, то и проверка будет происходить посимвольно, поэтому необходимо разработать ряд ограничений, накладываемых на формат ввода ответа, позволяющих уберечь пользователя от некорректного ввода, что в свою очередь приведет к отрицательному результату проверки.

Ввиду сложности реализации аппарата проверки логических выражений было принято решение использовать две элементарные фигуры: окружность и прямую линию.

Уравнение окружности представлено в формуле 1.1.

|  |  |
| --- | --- |
| , | (1.1) |

где – координаты центра окружности; – радиус окружности; – переменные.

Уравнение окружности представлено в формуле 1.2.

|  |  |
| --- | --- |
| , | (1.2) |

где – угловой коэффициент; – свободный член; – переменные.

Задание выстраивается из данных фигур, и количество фигур, одновременно представленных в задании должно быть меньше 5.

Правильной записью для каждой представленной фигуры в задании должно быть уравнение, написанное в формулах 1.1 и 1.2. Любая другая запись будет являться ошибочной с точки зрения проверяющего аппарата.

Само задание должно состоять из двух частей: графическое представление заданной области (сочетание примитивов: окружность, ломаная, многоугольник и линия) и текстовое представление, описывающее необходимую область в Си – подобном стиле. Текстовое представление храниться в памяти компьютера и используется только для проверки ответа, введенного пользователем.

Все задания представлены шаблонами, хранящимися в заранее подготовленной базе шаблонов.

Пример задания, который должен генерироваться программным продуктом представлен на рисунке 1.2.

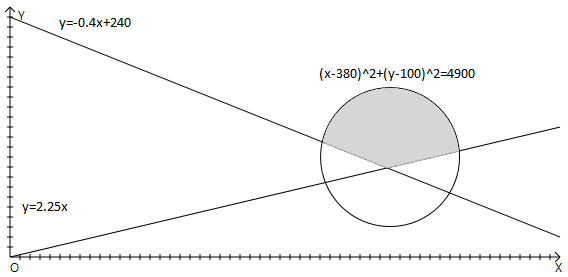


Рисунок 1.2 – Пример генерируемого задания

Для данной области эталонным решением будет являться следующее логическое выражение .

База шаблонов представляет собой XML-файл, разделенный тегами. Тег - элемент языка разметки гипертекста. Названия тегов и их назначение представлено в таблице 1.3.

Таблица 1.3 – Теги и их назначения

|  |  |
| --- | --- |
| **Название** | **Назначение** |
| <base>…</base> | Границы базы с шаблонами. |
| <template>…</template> | Границы шаблона. Имеет свойство «id», отвечающее за номер шаблона. |
| <primitive>…</primitive> | Количество и вид фигуры, используемой в шаблоне. |
| <circle>…</circle> | Окружность. Имеет свойство «id», необходимое для идентификации фигуры, и свойство «link», указывающее, с какой фигурой необходимо связать текущую. |
| <poluline>…</polyline> | Ломаная линия. Имеет свойство «id», необходимое для идентификации фигуры, свойство «link», указывающее, с какой фигурой необходимо связать текущую, и свойство «linecount», указывающее сколько линий будет. |

Продолжение таблицы 1.3.

|  |  |
| --- | --- |
| <polygon>…</ polygon > | Многоугольник. Имеет свойство «id», необходимое для идентификации фигуры, свойство «link», указывающее, с какой фигурой необходимо связать текущую, и свойство «linecount», указывающее сколько линий будет. |
| < area>…</area> | Обозначает границы эталонного решения для текущего шаблона с обозначением мест, куда необходимо вставить уравнения фигур, используемых в задании. |

Все вышеперечисленные теги являются необходимыми, и их искажение приведет к неисправности базы шаблонов.

## Технология обработки информации

### Описание вариантов использования

В ходе изучения предметной области выяснилось, что необходим один тип пользователя – пользователь, который может пройти тестирование на умение составлять логические выражения.

Обычный пользователь имеет возможность пройти тестирование (для этого необходимо запросить имя и фамилию), его результаты будут записаны в базу и выведены на экран.

### Тестирование

Тестирование проводится для того, чтобы оценить навыки студента в составлении логических выражений для оператора ветвления. Результаты тестирования оцениваются по 4-х балльной шкале (плохо, удовлетворительно, хорошо, отлично). Баллы зависят от процента правильных ответов тестируемого от общего количества вопросов.

Оценочная шкала описана в таблице 1.3.

Таблица 1.5 – Оценочная шкала по умолчанию

|  |  |
| --- | --- |
| **Оценка** | **Количество процентов** |
| Плохо | 0 – 59 |
| Удовлетворительно | 60 – 79 |
| Хорошо | 80 - 89 |
| Отлично | 90 - 100 |

Условия получения той или иной оценки неизменны и являются рекомендациями для преподавателя.

### Алгоритм обработки базы с шаблонами

Для представления в компьютере задания создается новый тип данных - Task. Он хранит в себе последовательность примитивов, выделяемую область и правильный ответ.

Заполнение полей экземпляра класса Task’a происходит по следующему алгоритму:

1. прочитать из тэга <template> значение поля “id” и записать в *Task*;
2. из элемента <primitives> прочитать в список примитивов все примитивы, используемые в шаблоне, и их свойства;
3. из элемента <area> прочитать область выделения и записать ее в *Task*;
4. конец.

### Алгоритм подготовки базы заданий для тестирования

1. Прочитать в *k* количество заданий в тесте из файла настроек;
2. объявить список заданий для тестирования;
3. открыть базу с шаблонами;
4. повторять от *i* = 0 до *k* с шагом 1
   1. прочитать очередной шаблон и записать его в список заданий;
5. конец.

### Алгоритм вывода задания на экран

1. Расставить элементы – примитивы на полотне;
2. в соответствии с заданием, выделить необходимую область на плоскости;
3. каждому элементу-примитиву добавить описание при наведении (указать координаты точек, размеры и прочие свойства, характерные для примитивов);
4. конец.

### Алгоритм проверки ответа тестируемого

1. Запросить у тестируемого ответ;
2. получить ответ от тестируемого в виде строки, удовлетворяющей требованиям C++;
3. перевести ответ пользователя в обратную польскую запись;
4. перевести эталонное решение в обратную польскую запись;
5. посимвольно сверить полученные строки из пунктов 3 и 4:
   1. если символы различны, то выдать отрицательный результат;
   2. если дошли до конца строки, а другая строка еще не кончилась, то вывести отрицательный результат;
   3. если обе строки закончились и не было найдено различающихся, то выдать положительный результат;
6. конец.

## Входные и выходные данные

Входные данные:

1. ФИО испытуемого (текст)
2. база с шаблонами заданий (XML файл, отвечающий требованиям таблицы 1.3);
3. ответы пользователя (строка, отвечающая требованиям корректности ввода – логическое выражение на Си подобном языке).

Выходные данные:

1. изображение на экране(картинка);
2. результаты теста (текст, числа).

## Требования к техническому и программному обеспечению

Разрабатываемая программная система должна соответствовать следующим минимальным требованиям, обусловленным заданием к проекту.

Рекомендуемая аппаратная конфигурация:

* Процессор с частотой 1,6 ГГц или выше
* 512 МБ ОЗУ
* 128 MБ свободного места на диске
* Жесткий диск
* Дисковод CD-ROM/DVD-ROM

Рекомендуемая операционная система:Microsoft Windows XP с пакетом обновлений 3(SP3) или старше.

1. **РАБОЧИЙ ПРОЕКТ**
   1. **Общие сведения о работе программы**

В результате выполнения курсовой работы была создана программа, позволяющая проводить автоматические тестирования на составление булевых выражений. В программе реализованы следующие алгоритмы:

1. генерации логических выражений;
2. генерация графической области, соответствующей логическому выражению;
3. 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.

2. 2. **Функциональное назначение**

Программа обеспечивает следующие функциональные возможности:

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

**Функциональные ограничения**

В базу шаблонов нельзя добавить новое задание или удалить существующее. Шаблоны, имеющиеся в базе представлены в приложении в таблице П.2.

* 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».

После инсталляции в папке программы появятся файлы, перечисленные в таблице 2.1.

Таблица 2.1 – Файлы программного продукта.

|  |  |
| --- | --- |
| Файл | Назначение |
| Генерация логических задач.exe | Исполняемый файл. |
| statistic.html | Файл статистики. |

Продолжение таблицы 2.1.

|  |  |
| --- | --- |
| base.xml | База шаблонов. |
| help.html | Файл справки. |
| Папка img | Содержит 3 изображения для справки. |

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

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

Модули, реализованные в программе представлены в таблице 2.2.

Таблица 2.2 – Реализованные модули и классы.

|  |  |
| --- | --- |
| Модуль | Класс |
| MainWindow.xaml | MainWindow |
| MainWindow.xaml.cs |
| Test.xaml | Test |
| Test.xaml.cs |
| Result.xaml | Result |
| Result.xaml.cs |
| Help.xaml | Help |
| Help.xaml.cs |
| Statistic.xaml | Statistic |
| Statistic.xaml.cs |
| RPN.cs | RPN |
| TestChec.cs | TestChec |
| from\_XML\_to\_Task.cs | from\_XML\_to\_Task |
| Task.cs | Task |
| Primitive.cs | Primitive |
| polyline.cs | polyline |
| poligon.cs | poligon |
| line.cs | line |
| ellips.cs | ellips |

Продолжение таблицы 2.2.

|  |  |
| --- | --- |
| lable.cs | lable |
| point.cs | point |

Функциональное назначение модулей представлено в таблицах 2.3-2.9.

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 2.3 приведены методы класса *Test*.

Таблица 2.3 – Описание класса *Test*

|  |  |  |  |
| --- | --- | --- | --- |
| **Поле** | **Тип** | | **Назначение** |
| **taskList** | **List<Task>** | | Список заданий. |
| **userName** | **string** | | Фамилия и имя тестируемого. |
| **fileAdress** | **string** | | Адрес базы шаблонов. |
| **Base** | **from\_XML\_to\_Task** | | Дешифратор базы. |
| **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 2.4 приведены методы класса Result.

Таблица 2.4 – Описание методов класса Result

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип** | **Назначение** |
| **ChekTest** | **TestChek** | Проверка теста. |

Продолжение таблицы 2.4.

|  |  |  |  |
| --- | --- | --- | --- |
| **taskResult** | **List<bool>** | | Список результатов теста. |
| **answer** | **List<string[]>** | | Список ответов. |
| **userName** | **string** | | Фамилия и имя тетсируемого. |
| **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 2.5 приведены методы класса RPN.

Таблица 2.5 – 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 2.6 приведены методы класса *Task*.

Таблица 2.6 – Описание методов класса *Task*

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип** | **Назначение** |
| **ID** | **int** | Идентификационный номер. |
| **ListPrimitives** | **List<Primitive>** | Список примитивов. |

Продолжение таблицы 2.6.

|  |  |  |  |
| --- | --- | --- | --- |
| **points** | **bool[,]** | | Массив точек, которые необходимо закрашивать. |
| **Area** | **string** | | Область выделения |
| **UserAnswer** | **string** | | Ответ пользователя. |
| **EtalonAnswer** | **string** | | Эталонное решение. |
| **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 2.7 приведены методы класса from\_XML\_to\_Task.

Таблица 2.7 – Описание методов класса from\_XML\_to\_Task

|  |  |  |  |
| --- | --- | --- | --- |
| **Поле** | **Тип** | | **Назначение** |
| **rand** | **Random** | | Случайное число. |
| **fileAdress** | **string** | | Адрес файла. |
| **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 2.8 приведены методы класса TestChec.

Таблица 2.8 – Описание методов класса TestChec

|  |  |  |  |
| --- | --- | --- | --- |
| **Поле** | **Тип** | | **Назначение** |
| **newRPN** | **RPN** | | Переменная, переводящая логическое выражение из инфиксной записи в постфиксную. |
| **Прототип** | | **Назначение** | |
| **public bool getChek(string Etal, string User)** | | Сверяет строку эталонного решения с ответом пользователя. | |

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

В таблице 2.9 приведены методы класса Primitive.

Таблица 2.9 – Описание методов класса Primitive

|  |  |  |  |
| --- | --- | --- | --- |
| **Поле** | **Тип** | | **Назначение** |
| **ID** | **int** | | Идентификационный номер. |
| **LineWidth** | **int** | | Ширина линии. |
| **Width** | **int** | | Ширина примитива. |
| **Hieght** | **int** | | Высота примитива. |
| **R** | **int** | | Радиус круга. |
| **Coordinate** | **point** | | Центр круга. |
| **One** | **point** | | Первая точка линии. |
| **Two** | **point** | | Вторая точка линии. |
| **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. | |

Продолжение таблицы 2.9.

|  |  |
| --- | --- |
| **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. |
| **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. |

Классы line, polyline, polygon, label, point, ellipse наследуются от класса Primitive и используют методы, объявленные в нем.

Модуль Help*.xaml.cs* содержит класс Help, который реализует логику работы модуля Help*.xaml,* описывающий вывод справки по проведению тестирования.

* 1. **Разработанные меню и интерфейсы**

После запуска исполняемого файла программы, откроется главное окно.

Главное окно программы показаны на рисунке 2.1.

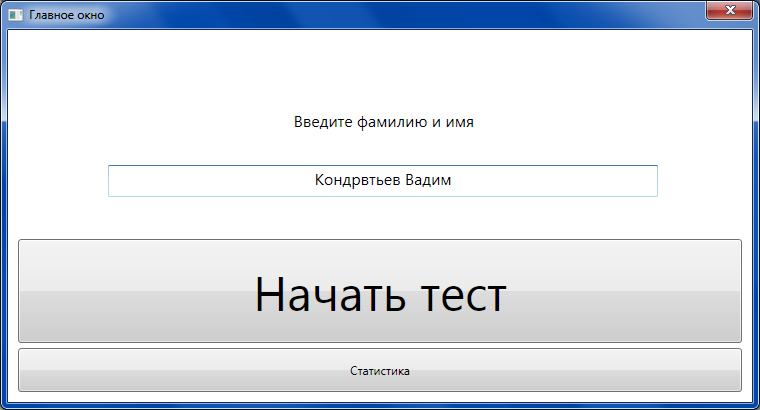


Рисунок 2.1 – Основное окно программы.

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».

Окно статистики представлено на рисунке 2.5.

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

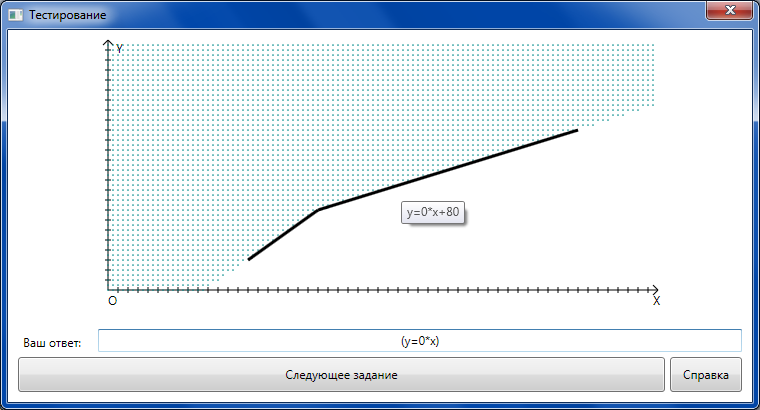


Рисунок 2.2 – Интерфейс окна режима тестирования

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

При нажатии на кнопку «Справка» откроется окно справки, представленное на рисунке 2.4.

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

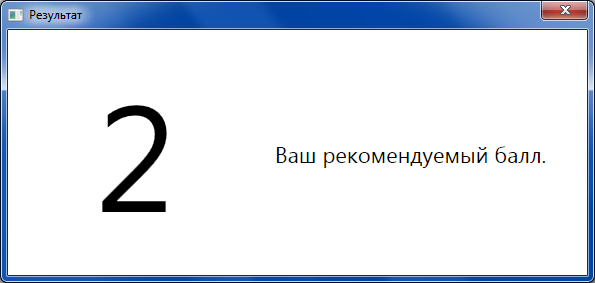


Рисунок 2.3 – Окно результата тестирования

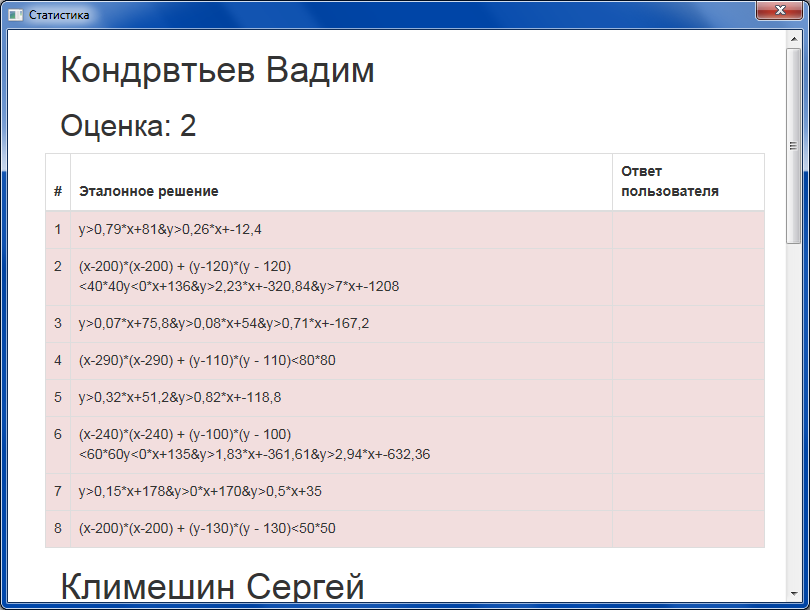


Рисунок 2.4 – Окно результатов тестирования с информацией о пройденных заданиях

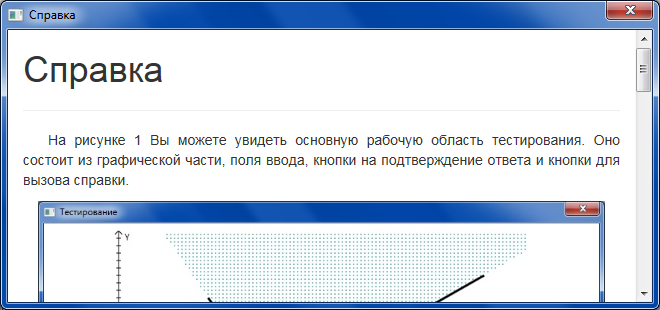


Рисунок 2.5 – Окно справки.



# ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ

* 1. **Проверка работоспособности программы:**
     1. Запустить файл «Генерация логических задач.ехе» на выполнение.
     2. Нажать на кнопку «Статистика».
     3. Убедиться, что открылось новое окно со статистикой.
     4. Закрыть окно.
     5. Ввести имя и фамилию в поле ввода и нажать на кнопку «Начать тест».
     6. Нажать на кнопку «Справка».
     7. Убедиться, что открылось окно со справкой.
     8. Закрыть окно.
     9. Навести курсор на один из графических элементов.
     10. Убедиться, что высветилась подсказка, содержащая уравнение элемента.
     11. Ввести уравнение в поле ответа.
     12. Повторить пункты 3.1.9-3.1.11, составив логическое выражение.
     13. Пройти все задания подобным образом.
     14. Получить свою оценку.
     15. Закрыть окно.
     16. Запустить файл «Генерация логических задач.ехе» на выполнение.
     17. Открыть статистику.
     18. Убедиться, что добавилась статистика по только что пройденному тесту.
     19. Убедиться, что оценка соответствует количеству правильных ответов в соответствии с таблицей 1.5.

**ЗАКЛЮЧЕНИЕ**

В результате выполнения курсовой работы была разработана программа, позволяющая сгенерировать логическое выражение из заготовленного шаблона, построить по этому выражению некую геометрическую область и провести тестирование по сгенерированным заданиям.

В дальнейшем планируется продолжить работу над программой, добавив в нее новые функции и новые геометрические фигуры, а также улучшив аппарат тестирования с помощью интерпретатора логических выражений, что позволит свести к минимуму разницу между эталонным решением и ответом пользователя.

# ЛИТЕРАТУРА

1. Петцольд Ч. MicrosoftWindows Presentation Foundation: — M.: Издательство «Русская Редакция»; СПб.: Питер, 2008. — 944 с.
2. Мак-Дональд Мэтью. WPF: Windows Presentation Foundation в .NET 3.5 с примерами на C# 2008 для профессионалов, 2-е издание: Пер. с англ. — М. : ООО “И.Д. Вильямс”, 2008. — 928 с.
3. НэшТрей. C# 2008: ускоренный курс для профессионалов. : Пер. с англ. – М. : ООО «И.Д. Вильямс». 2008. – 576 с.

# ПРИЛОЖЕНИЕ 1



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

# ПРИЛОЖЕНИЕ 2

Шаблоны, имеющиеся в базе представлены в таблице П.2.

Таблица П.2 – Шаблоны в базе.

|  |  |
| --- | --- |
| **Описание** | **Графическое представление** |
| Графическая часть представлена двумя прямыми.  Необходимо указать область выше них. |  |
| Графическая часть представлена окружностью.  Необходимо указать область внутра круга. |  |
| Графическая часть представлена многоугольником.  Необходимо указать область внутри него. |  |
| Графическая часть представлена двумя окружностями.  Необходимо указать область пересечения этих окружностей. |  |

Продолжение таблицы П.2.

|  |  |
| --- | --- |
| Графическая часть представлена окружностью и тремя линиями.  Необходимо указать область, лежащую внутри окружности и расположенную выше этих линий. |  |
| Графическая часть представлена двумя линиями.  Необходимо указать область ниже линий. |  |
| Графическая часть представлена окружностью.  Необходимо указать область, расположенную вне окружности. |  |
| Графическая часть представлена двумя многоугольниками.  Необходимо указать область пересечения этих многоугольников. |  |
| Графическая часть состоит из окружности и трех линий.  Необходимо указать область, расположенную внутри круга и ниже линий. |  |

Продолжение таблицы П.2.

|  |  |
| --- | --- |
| Графическая часть представлена двумя ломаными линиями по три звена каждая.  Необходимо указать область, расположенную между этими ломанными. |  |