Министерство образования Республики Беларусь

УО «Бобруйский государственный аграрно-экономический колледж»

|  |  |  |
| --- | --- | --- |
| Специальность 2-40 01 01  **«Программное обеспечение информационных технологий**  **(по направлениям)»** |  | Допущен к защите  Зав. отделением  \_\_\_\_\_\_\_\_\_\_\_ О.П. Глёкова  \_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ 2020 |
|  |  |  |
| Направление специальности 2-40 01 01-35 **«Программное обеспечение обработки экономической и деловой информации»** |  |  |

**КУРСОВОЙ ПРОЕКТ**

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

|  |  |
| --- | --- |
| **ТЕМА:** | Прикладная программа «Клавиатурный тренажер для |
|  | программиста» учащейся отделения «Бухгалтерский учёт, |
|  | анализ и контроль» для УО «БГАЭК» |

Группа: 19П

|  |  |  |  |
| --- | --- | --- | --- |
| Разработал |  |  | Почерняева Д.Н. |

|  |  |  |  |
| --- | --- | --- | --- |
| Руководитель |  |  | Кошелева В.П. |

|  |  |  |
| --- | --- | --- |
| Проект защищен на отметку |  |  |

(отметка)

|  |  |  |
| --- | --- | --- |
| Дата защиты |  |  |

г. Бобруйск, 2020 г.

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc43658902)

[1 ПОСТАНОВКА ЗАДАЧИ 6](#_Toc43658903)

[1.1 Организационно-экономическая сущность задачи 6](#_Toc43658904)

[1.2 Входная оперативная информация 8](#_Toc43658905)

[1.3 Постоянная информация задачи 8](#_Toc43658906)

[1.4 Выходная информация 8](#_Toc43658907)

[2 ОБЪЕКТНО-ОРИЕНТИРОВАННЫЙ АНАЛИЗ ЗАДАЧИ 10](#_Toc43658908)

[2.1 Описание задачи в ключевых абстракциях предметной области 10](#_Toc43658909)

[2.2 Прототипирование графического интерфейса 14](#_Toc43658910)

[2.3 Описание интерфейса классов 15](#_Toc43658911)

[3 ОПИСАНИЕ ПРОГРАММЫ 23](#_Toc43658912)

[3.1 Общие сведения 23](#_Toc43658913)

[3.2 Функциональное назначение 23](#_Toc43658914)

[3.3 Описание логической структуры 23](#_Toc43658915)

[3.4 Используемые технические средства 23](#_Toc43658916)

[3.5 Вызов и загрузка 24](#_Toc43658917)

[4 ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ 26](#_Toc43658918)

[4.1 Цель испытаний 26](#_Toc43658919)

[4.2 Описание используемых методов испытаний и их результаты 28](#_Toc43658920)

[5 ОПИСАНИЕ ПРИМЕНЕНИЯ 37](#_Toc43658921)

[5.1 Назначение программы 37](#_Toc43658922)

[5.2 Условия выполнения программы 37](#_Toc43658923)

[5.3 Выполнение программы 37](#_Toc43658924)

[5.4 Сообщения оператору 42](#_Toc43658925)

[ЗАКЛЮЧЕНИЕ 44](#_Toc43658926)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 46](#_Toc43658927)

[ПРИЛОЖЕНИЕ А 48](#_Toc43658928)

[ПРИЛОЖЕНИЕ Б 49](#_Toc43658929)

[ПРИЛОЖЕНИЕ В 50](#_Toc43658930)

# ВВЕДЕНИЕ

На современном этапе развитие компьютерных технологий происходит быстрыми темпами, поэтому для более производительной работы на персональном компьютере (далее – ПК) нужно обязательно осваивать новые навыки. Одним из таких навыков является быстрый набор данных на клавиатуре.

Чтобы пользователь смог существенно ускорить набор текста было создано множество специализированных программ, которые называются клавиатурными тренажерами. В частности, навык быстрого набора данных необходим специалистам, чья профессия непосредственно связана с работой на компьютере. Такой профессией является профессия программиста и связанные с ней профессии. Но чтобы программисту значительно ускорить процесс написания программ, недостаточно тренироваться в программах, направленных на ускорение печати сплошных текстов на русской раскладке клавиатуры.

Поэтому для курсового проектирования была выбрана тема прикладная программа «Клавиатурный тренажер для программиста».

Объектом курсового проекта является программное обеспечение информационных технологий в образовании.

Предметом курсового проекта является прикладная программа «Клавиатурный тренажер для программиста».

Цель курсового проектирования состоит в разработке, тестировании и отладке прикладной программы «Клавиатурный тренажер для программиста».

Задачами, в соответствии с целью, являются:

1) изучение материала взятой темы;

2) получение навыков в разработке программного обеспечения на языке программирования C#;

3) определение функций, которые должно выполнять разрабатываемое программное обеспечение (далее – ПО);

4) реализация, отладка и тестирование созданного рабочего проекта;

5) разработка технической и пользовательской документации.

Новизна создаваемого программного обеспечения заключается в том, что в существующих аналогах не поддерживается одновременно обучение на языках программирования C#, Pascal, Python, JavaScript, PL/SQL. Благодаря такому подходу, программист (далее – Пользователь) сможет обучиться быстрому набору зарезервированных слов языка программирования, на котором он разрабатывает программное обеспечение.

Разрабатываемая программа должна отвечать следующим функциональным требованиям:

* предусмотреть возможность выбора языка программирования для выполнения заданий;
* предусмотреть несколько уровней заданий различной сложности;
* задания должны загружаться в программу из внешних файлов;
* должен быть контроль ввода задания, то есть выделение ошибок и правильно введенных символов;
* предусмотреть опцию смены цветовой темы программы;
* предусмотреть вывод данных о выполненном задании, такие как количество напечатанных символов, время выполнения задания, количество совершенных в задании ошибок, а также скорость печати, выраженная в символах в минуту.

Для создания программы будет использоваться актуальная версия среды разработки Visual Studio 2019, язык программирования C# (C Sharp) на платформе .NET Framework 4.7.2 с использованием шаблона Windows Forms.

# 1 ПОСТАНОВКА ЗАДАЧИ

* 1. **Организационно-экономическая сущность задачи**

Основанием для проведения разработки является курсовое проектирование по дисциплине «Конструирование программ и языки программирования».

Наименование темы разработки – прикладная программа «Клавиатурный тренажер для программиста».

Функциональным назначением программы является возможность совершенствовать скорость набора ключевых выражений определенных языков программирования (далее – ЯП). Конечными пользователями будут являться программисты, которые ведут разработку на одном из таких ЯП, как C#, Pascal, Python, JavaScript, PL/SQL.

Основные функции приложения:

* выбор ЯП для выполнения заданий;
* открытие задания из внешнего файла;
* выполнение задания с подсчетом времени выполнения, ошибок и скорости печати;
* вывод результатов выполненного задания.

Нефункциональные требования к разрабатываемому программному обеспечению:

* должно работать под управлением операционной среды Windows 7/10;
* должна иметь название и иконку, которая отображается при открытии программы в панели и диспетчере задач;
* должна иметь справочное руководство, содержащее графические материалы с объяснением работы;
* должна иметь понятный и удобный интерфейс на русском языке;
* иметь в наличии инсталлятор с установкой нужной версии Framework при необходимости.

Для разбиения проекта на более управляемые элементы была использована иерархическая структура работ (далее – ИСР), представленная на рисунке 1.1.

Иерархическая структура работ – инструмент для разбиения всего проекта на более управляемые элементы путем декомпозиции результатов проекта на составляющие либо временные результаты выполнения. ИСР отображается в виде иерархической диаграммы, где на первом уровне находится один элемент, это обычно название проекта, а нижние уровни представляют собой все необходимые компоненты либо работы, которые нужны для получения результатов проекта. Второй уровень ИСР разбит либо на крупные составные компоненты и работы проекта.



Рисунок 1.1 – ИСР проекта

Следующий шаг планирование проекта – составление календарного плана проекта (расписание проекта). Календарный план проекта – это инструмент, используемый для планирования, составления расписания и контроля хода выполнения проекта. Оно представляет собой основу информационной системы проекта, которая будет использоваться менеджером проекта для принятия решений, связанных с управлением временем проекта, его стоимостью и ходом выполнения.

Календарный план проекта содержит полный перечень работ проекта, которые необходимо выполнить, логическую последовательность и взаимозависимость этих работ, возможные сроки начала и окончания, как отдельных работ, так и проекта в целом. Календарный план проекта составлен в Microsoft Project и представлен на рисунке 1.2. Диаграмма Ганта, полученная в соответствии с календарным планом, представлена на рисунке 1.3.

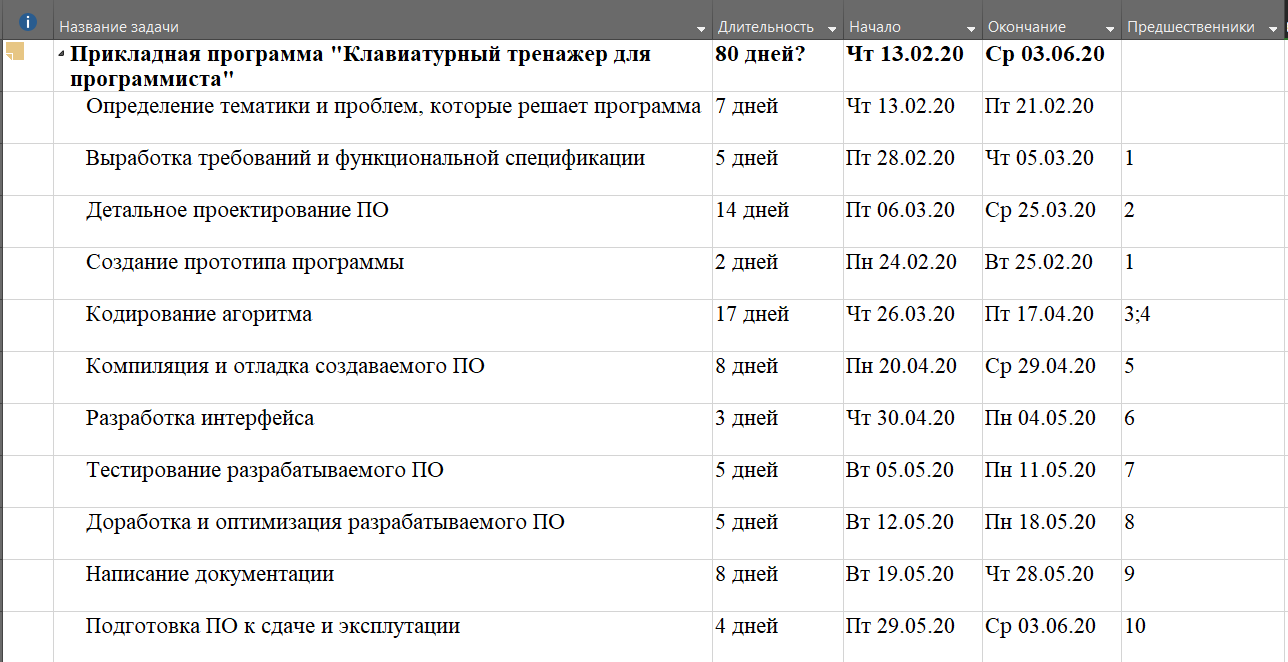


Рисунок 1.2. – Расписание проекта

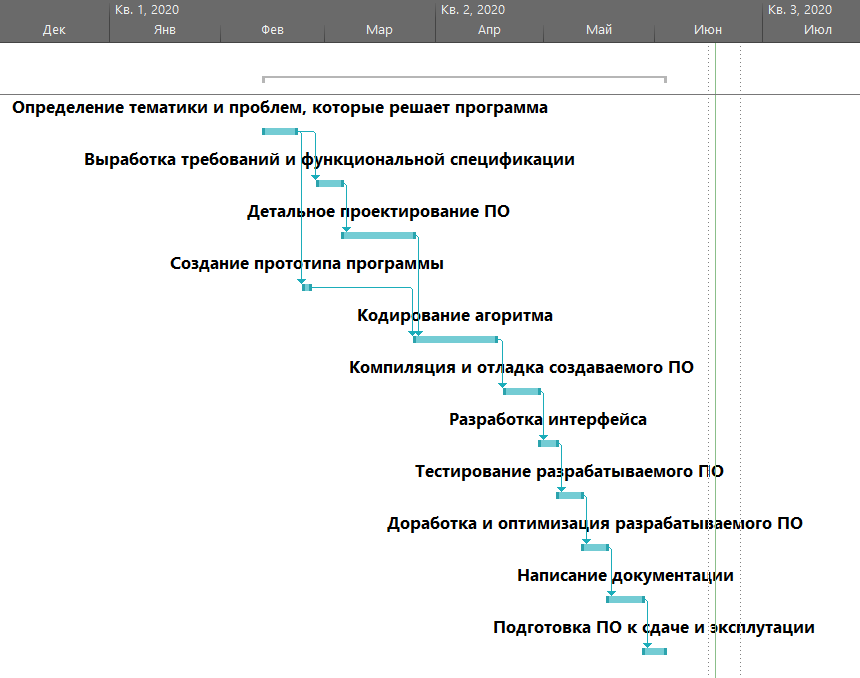


Рисунок 1.3 – Диаграмма Ганта проекта

* 1. **Входная оперативная информация**

Входной информацией в программе будут являться данные выбранные Пользователем перед началом выполнения задания такие как ЯП, на котором должны выполняться задания, и уровень задания.

В качестве компонента для выбора языка программирования планируется использовать ComboBox, выбранная строка которого будет являться входной информацией, то есть выбранным языком программирования для дальнейшего прохождения заданий.

В качестве компонента для выбора уровня заданий планируется использовать Button или Label, то есть компоненты будут представлять собой кнопки с номером задания.

Данная входная информация позволяет начать задание с выбранными пользователями данными. Входной набор параметров выбора языка программирования и выбора номера заданий имеет постоянный неизменяемый набор значений, потому что Пользователю уже представлен набор входных данных, из которых он должен выбрать подходящие для него данные.

* 1. **Постоянная информация задачи**

Постоянная информация – это неизменная и многократно используемая в течение длительного периода времени информация. К постоянной информации в программном средстве будут относиться текстовые файлы формата txt, считанные данные из которых являются непосредственно текстом заданий. Также к дополнительной постоянной информации относится справочные данные, которые предоставляют информацию о программе такую как описание программы, работу с программой и информацию об авторе.

* 1. **Выходная информация**

К выходной информации будут принадлежать данные о выполненном задании, такие как количество напечатанных символов, время выполнения задания, количество совершенных в задании ошибок, а также скорость печати, выраженная в символах в минуту.

Количество напечатанных символов будут рассчитываться в течении всего выполнения задания Пользователем с проверкой на соответствие текущего символа в задании с нажатым символом Пользователем на клавиатуре. Если текущий символ совпадает с нажатым символом на клавиатуре, то осуществляется подсчет правильно нажатых символов, иначе происходит подсчет неправильно нажатых символов. По итогу выполнения задания количество неправильно нажатых символов и правильно нажатых символов складывается и рассчитывается общее количество напечатанных символов.

Время выполнения задания начинает отсчет после открытия задания программой и до конца выполнения задания Пользователем.

Скорость печати рассчитывается исходя из общего количества напечатанных Пользователем символов и времени выполнения задания по формуле:

,

где – скорость печати, выраженная в символах в минуту;

– общее количество напечатанных символов;

– общее время выполнения задания.

# ОБЪЕКТНО-ОРИЕНТИРОВАННЫЙ АНАЛИЗ ЗАДАЧИ

Одной из основных задач, решаемых на этапе анализа и ранних стадиях проектирования, является выявление ключевых абстракций задачи – классов и объектов, составляющих словарь предметной области. Объектно-ориентированное программирование – это подход к разработке программного обеспечения, основанный на объектах, а не на процедурах. Объектно-ориентированное программирование базируется на связывании или инкапсуляции структур данных и процедуры, которая работает с данными в структуре, с модулем. Цели проведения объектно-ориентированного анализа:

* понять проблему или проблемы, которые программная (или иная) система должна решить;
* задать значимые вопросы о проблеме и о ПО;
* обеспечить основу для ответов на вопросы о специфических свойствах проблемы и ПО;
* определить, что ПО должно делать;
* определить, что ПО не должно делать;
* убедиться, что ПО удовлетворит потребности ее пользователей, и определить критерии ее приемки;
* обеспечить основу для разработки ПО.

Объектный подход применяется на всех основных стадиях жизненного цикла ПО и включает в себя три ключевых понятия:

OOA (object oriented analysis) – объектно-ориентированный анализ.

OOD (object oriented design) – объектно-ориентированное проектирование.

OOP (object oriented programming) – объектно-ориентированное программирование.

* 1. **Описание задачи в ключевых абстракциях предметной области**

На современном этапе насчитывается достаточно большое количество методологий, поддерживающих объектно-ориентированный подход к разработке, к основным относятся:

* итеративная методология, включающая RAD-концепцию (Rapid Application Development — быстрая разработка приложений).
* гибкая методология разработки, включающая в себя такие концепции, как Lean, Scrum, Kanban, XP, Agile.

Для объектно-ориентированного анализа и проектирования используется UML (унифицированный язык моделирования), который применяется для визуализации, спецификации, конструирования и документирования программных средств.

Одной из неотъемлемых частей объектно-ориентированного анализа является разработка диаграммы вариантов использования, позволяющей наглядно представить возможности разрабатываемой программы (то, что она будет делать). Каждый вариант использования – последовательность действий, отображающий отдельную функцию системы. В качестве инструментальной среды проектирования используется редактор диаграмм и блок-схем Microsoft Visio 2010. В качестве языка моделирования используется язык Unified Modelling Language (UML).

Для построения диаграммы вариантов использования необходимо:

* выбрать актера, который будет являться инициатором действия;
* определить функционирование системы при том или ином варианте использования.

Диаграмма вариантов использования разрабатываемого программного средства изображена на рисунке 2.1.



Рисунок 2.1 – Диаграмма вариантов использования

Разберем основные варианты использования, представленные на рисунке 2.1.

Вариант использования – Просмотр справки.

Исполнитель: Пользователь.

Описание: Открытие справки для просмотр дополнительных сведений о программе. Контекст использование: Необходимость в помощи при работе с программой.

Вариант использования – Смена цветовой темы. Исполнитель: Пользователь. Описание: изменение темы программы на другую цветовую гамму. Контекст использование: необходимость смены контрастности программы, исходя из предпочтений Пользователя.

Вариант использования – Выполнение задания. Исполнитель: Пользователь. Описание: прохождение задания включает такие прецеденты, как выбор языка программирования, на котором должны выполняться задания, выбор номера задания для прохождения, печать текста задания и просмотр результатов выполненного задания. Контекст использования: проверка скорости печати Пользователя, совершенствование набора ключевых слов.

Для описания логики будущих методов и поведение программного средства была построена диаграмма деятельности. Диаграмма деятельности – это технология, позволяющая описывать логику процедур, бизнес-процессы и потоки работ. Диаграмма деятельности разрабатываемой программы представлена на рисунке 2.2.



Рисунок 2.2 – Диаграмма деятельности

Диаграмма компонентов, в отличие от выше представленных диаграмм, описывает особенности физического представления системы. Диаграмма компонентов позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами, в роли которых может выступать исходный, бинарный и исполняемый код. Пунктирные стрелки, соединяющие модули, показывают отношения взаимозависимости, аналогичные тем, которые имеют место при компиляции исходных текстов программ. Основными графическими элементами диаграммы компонентов являются компоненты, интерфейсы и зависимости между ними.



Рисунок 2.3 – Диаграмма компонентов

На диаграмме компонентов отображены системные компоненты, из которых, в результате складывается рабочая программа. На данной диаграмме мы можем наблюдать, что данная программа состоит из следующих компонентов:

Тренажер для программиста.exe – исполняемый файл проекта;

.dll – компонент, в котором находятся библиотеки, которые используются в программе;

.txt – компонент, содержащий документы текстового формата, которые содержат текст заданий;

.html – компонент, представляющий собой файл справочного руководства, открывающийся в браузере;

.files – пакет, в котором выгружены все файлы и изображения для открытия файла справки;

Тренажер для программиста.sln – компонент, содержащий структурированный файл, который используется для организации проекта C#;

.cs – компонент файлов исходного кода, созданный на языке C#, представляет собой формы;

.resx – компонент файлов ресурсов проекта;

.config – в данном компонента содержатся файлы конфигурации приложения;

.designer.cs – в данном компоненты содержаться информацию об элементах управления, размещенных на формах;

.csproj – в данном компоненте содержится проект, в нем содержатся всевозможные настройки проекта, а также ссылки на все файлы, используемые данным проектом;

Resources – пакет, представляющий собой папку, в которой хранятся все ресурсы используемые для компиляции проекта;

* 1. **Прототипирование графического интерфейса**

Правильное размещение элементов интерфейса повышает его удобство использования, делает программное средство более привлекательным для Пользователя. Нужно четко представлять, чего ожидают от программы, какие элементы должны присутствовать в каждом окне. С этой целью на этапе проектирования предварительно создают прототип – наглядная модель пользовательского интерфейса, созданная на основе представлений о потребностях Пользователей. Он содержит все основные элементы, представленные в виде блоков.

Создание прототипа дает множество преимуществ:

* возможность сосредоточиться на решении ключевых задач. Необходимо сосредоточить усилия Разработчика на самых важных частях продукта, то есть на ключевых задачах, наиболее ценных, нужных и важных для пользователя;
* точность. Одна из самых серьезных проблем при разработке программы – исправления по причине многократных изменений конструкции и реализации пользовательского интерфейса;
* планирование. Нельзя составить точный план, не зная структуры пользовательского интерфейса. Если интерфейс будет изменяться по ходу цикла разработки, то в план проекта также придется вносить соответствующие изменения. Планирование требует четкого понимания структуры пользовательского интерфейса уже в начале работы над проектом;
* тестирование. Тестирование программы также сильно зависит от ее интерфейса. Обладая прототипом пользовательского интерфейса, Тестировщик смогут изучить полностью программное средств, а разобравшись в работе функций программы, команда сможет подготовить лучшие планы тестов и испытаний до окончания разработки программы.

Макет разрабатываемого программного обеспечения, на котором будет основываться разработка программного обеспечения, был разработан в Adobe Photoshop CS6 и представлен на рисунках 2.4 и 2.5.



Рисунок 2.4 – Прототип главного окна со свернутым меню

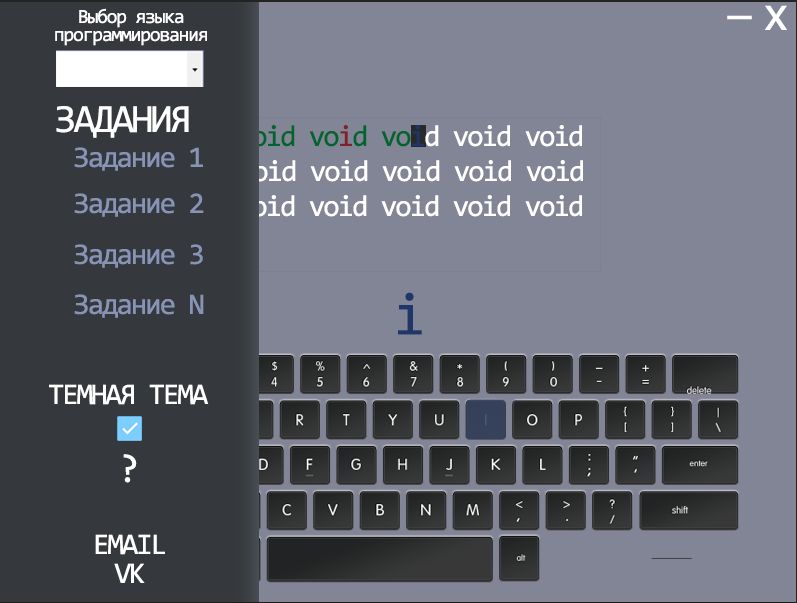


Рисунок 2.5 – Прототип главного окна с развернутым меню

* 1. **Описание интерфейса классов**

Для реализации прикладной программы используются следующие встроенные компоненты Windows Forms: ComboBox, FlowLayoutPanel, Label, Panel, PictureBox, RichTextBox, Timer и TextBox, среди которых только Timer является невизуальным.

Элемент ComboBox образует выпадающий список и совмещает функциональность компонентов ListBox и TextBox.

Элемент FlowLayoutPanel представляет панель, которая динамически располагает содержимое по горизонтали или вертикали.

Элемент Label служит для отображения простого текста на форме, доступного только для чтения.

Элемент Panel представляет панель, который объединяет элементы в группы.

Элемент TextBox представляет собой текстовое поле для ввода и редактирования текста.

Элемент RichTextBox является текстовым полем, которое поддерживает форматирование.

Элемент Timer является компонентом для запуска действий, повторяющихся через определенный промежуток времени.

Также был использован класс ToolTip, представляющий собой небольшое прямоугольное всплывающее окно, в котором отображается краткое описание назначения элемента управления.

Полное описание использованных компонентов представлено в таблице 2.1.

Таблица 2.1 – Используемые компоненты

|  |  |  |  |
| --- | --- | --- | --- |
| Имя формы | Используемые компоненты | Имя объекта | Предназначение компонента |
| 1 | 2 | 3 | 4 |
| Trenazher (Главная форма) | Button | b0 – b9 | Отображают цифры на панели с клавиатурой |
| ba – bz | Отображают буквы на панели с клавиатурой |
| bbackslash | Отображает слеш вправо на панели с клавиатурой |
| bbackspase | Отображает знак удаления возврата на панели с клавиатурой |
| bcaps | Отображает знак Caps Lock на панели с клавиатурой |
| bclosebrace | Отображает знак закрытой квадратной скобки на панели с клавиатурой |
| bcomma | Отображает знак запятой на панели с клавиатурой |
| benter | Отображает знак Enter на панели с клавиатурой |
| Button | bminus | Отображает знак минуса на панели с клавиатурой |
| bopenbrace | Отображает знак открытой квадратной скобки на панели с клавиатурой |
| bpoint | Отображает знак точки на панели с клавиатурой |
| bquote | Отображает знак одинарной кавычки на панели с клавиатурой |
| bravno | Отображает знак равенства на панели с клавиатурой |

*Продолжение таблицы 2.1*

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Trenazher | Button | bsemi | Отображает знак точки с запятой на панели с клавиатурой |
| bshift | Отображает знак левый Shift на панели с клавиатурой |
| bshiftright | Отображает знак правый Shift на панели с клавиатурой |
| bslash | Отображает слеш влево на панели с клавиатурой |
| bspacebar | Отображает знак пробела на панели с клавиатурой |
| btab | Отображает знак Tab на панели с клавиатурой |
| ComboBox | cb\_lang | Позволяет выбрать язык программирования в выпадающем списке |
| CheckBox (библиотека MaterialSkin) | cb\_Thema | Позволяет сменить тему в программе по нажатию |
| FlowLayoutPanel | flp\_zhad | Представляет собой панель с заданиями |
| Label | l\_zhad | Представляет собой кнопку «Задания» |
| lsled | Представляет собой надпись, на которой отображен символ, который Пользователь должен нажать следующим |
| lThema | Представляет собой надпись «Темная тема» |
| lvibor | Представляет собой надпись «Выбор языка программирования» |
| lVr | Отображает время, считанное с таймера |
| lVremya | Представляет собой надпись «Время: » |
| Panel | p\_klav | Представляет собой панель, на которой отображается клавиатура с кнопок |
| p\_lang | Представляет собой панель, на которой доступна смена языка программирования |
| Panel | pMenu | Представляет собой выплывающее меню |
| p\_zhadan | Представляет собой панель, на которой находится кнопка «Задания» |
| PictureBox | pb\_email | Представляет собой изображение, по нажатию на которое открывается почтовый клиент |

*Продолжение таблицы 2.1*

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Trenazher | PictureBox | pb\_exit | Представляет собой изображение, по нажатию на которое закрывается программа |
| pb\_inst | Представляет собой изображение, по нажатию на которое открывается Instagram разработчика |
| pb\_minim | Представляет собой изображение, по нажатию на сворачивается программа |
| pb\_sprav | Представляет собой изображение, по нажатию на которое открывается справка к программе |
| pb\_ukaz | Представляет собой изображение, по нажатию на выплывает меню |
| pb\_vk | Представляет собой изображение, по нажатию на которое открывается ВК разработчика |
| RichTextBox | rtZhadanie | Отображает текст с заданием |
| Timer | t\_menu | Таймер, задействованный в разворачивании меню |
| t\_vremya | Таймер, производящий отсчет времени выполнения задания |
| t\_zhadan | Таймер, задействованный в разворачивании панели с заданиями |
| TextBox | tbPole | Текстовое поле, в которое загружается текст задания |
| Label | zhad\_1 | Надпись, выступающая в роли кнопки с заданием «1» |
| zhad\_2 | Надпись, выступающая в роли кнопки с заданием «2» |
| zhad\_3 | Надпись, выступающая в роли кнопки с заданием «3» |
| zhad\_4 | Надпись, выступающая в роли кнопки с заданием «4» |
| zhad\_5 | Надпись, выступающая в роли кнопки с заданием «5» |
| MessageBox (Форма с выводом информации) | Label | l\_mes1 | Надпись, выводящее сообщение перед началом задания |
| l\_mes2 | Надпись, выводящее сообщение, если не был выбран язык программирования |
| l\_mes3 | Надпись, выводящее результаты выполнения задания |
| l\_ok | Надпись, выступающая в роли кнопки «OK», закрывающая окно по нажатию |

Методы, которые были разработаны в программе, подробно описаны в таблице 2.2.

В программе представлено два модуля с названиями Trenazher и MessageBox, содержащие содержит в себе набор методов, каждый из которых выполняет определенный блок кода.

Таблица 2.2 – Методы, использованные в программе

|  |  |  |
| --- | --- | --- |
| Модуль | Метод | Описание |
| 1 | 2 | 3 |
| Trenazher | private void Trenazher\_Load(object sender, EventArgs e) | Действия, которые происходят при загрузке формы Пользователем. |
| private string Chtenie(string f) | Считывание информации из файла задания в текстовое поле. |
| private void Otkritie(string f) | Действия, устанавливающие параметры для выполнения задания. Начало выполнения задания. |
| private void Cvet (Button clav, char[] s, int next, char key) | Действия, меняющие цвет клавиши при нажатии с проверкой на соответствие нажатой клавиши и текущей по заданию. Изменение цвета символа в тексте задания, а также подсчет правильно и неправильно нажатых символов. |
| private void rtZhadanie\_KeyPress (object sender, KeyPressEventArgs e) | Отслеживание нажатия символов и подсчет результатов выполнения задания. |
| private void t\_vremya\_Tick (object sender, EventArgs e) | Работа таймера с отсчетом времени выполнения задания |
| private void pb\_ukaz\_Click (object sender, EventArgs e) | Сворачивание и разворачивание меню, по нажатию на элемент pb\_ukaz. |
| private void Form1\_MouseMove (object sender, MouseEventArgs e) | Если мышь активна на форме, то выплывающее меню сворачивается. |
| private void t\_menu\_Tick (object sender, EventArgs e) | Таймер, управляющий временем сворачивания, разворачивания меню. |
| private void t\_zhadan\_Tick (object sender, EventArgs e) | Таймер, управляющий временем сворачивания, разворачивания панели с заданиями. |
| private void zhad\_1\_Click(object sender, EventArgs e) – private void zhad\_5\_Click(object sender, EventArgs e) | Методы, вызывающие открытие соответствующего задания. |

*Продолжение таблицы 2.2*

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| Trenazher | private void pb\_sprav\_Click (object sender, EventArgs e) | Открытие справки. |
| private void pb\_sprav\_MouseHover (object sender, EventArgs e);  private void pb\_email\_MouseHover (object sender, EventArgs e);  private void pb\_inst\_MouseHover (object sender, EventArgs e);  private void pb\_vk\_MouseHover (object sender, EventArgs e) | При некоторой задержкой над элементом появляется соответствующая подсказка к элементу. |
| private void l\_zhad\_Click(object sender, EventArgs e) | По нажатию на элемент l\_zhad происходит сворачивание, разворачивание заданий. |
| private void pb\_vk\_Click (object sender, EventArgs e) | Открытие страницы ВК разработчика. |
| private void pb\_inst\_Click (object sender, EventArgs e) | Открытие страницы Instagram разработчика. |
| private void cbThema\_Click(object sender, EventArgs e) | Вызов смены цветовой схемы. |
| private void pb\_email\_Click(object sender, EventArgs e) | Открытие почтового клиент с написанием письма разработчику. |
| private void pb\_minim\_Click(object sender, EventArgs e) | Происходит сворачивание программы в панель задач. |
| private void pb\_exit\_Click(object sender, EventArgs e) | Происходит закрытие программы. |

*Продолжение таблицы 2.2*

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| Trenazher | private void svetlaya\_thema() | Светлая тема, то есть изменение цветов элементов на светлые оттенки. |
| private void temnaya\_thema() | Темная тема, то есть изменение цветов элементов на темные оттенки. |
| private void l\_zhad\_MouseMove (object sender, MouseEventArgs e) и аналогичные методы | Если мышь активна над данным компонентом, то вызывается определенный цвет или изображение, в соответствии с типом элемента. |
| private void l\_zhad\_MouseLeave (object sender, EventArgs e) и аналогичные методы | Если мышь не активна над данным компонентом то вызывается определенный цвет или изображение, в соответствии с типом элемента. |
| MessageBox | private void l\_ok\_MouseMove (object sender, MouseEventArgs e) | Если мышь активна над данным компонентом, то вызывается определенный цвет. |
| private void l\_ok\_MouseLeave (object sender, EventArgs e) | Если мышь не активна над данным компонентом, то вызывается определенный цвет. |
| private void l\_ok\_Click(object sender, EventArgs e) | Происходит закрытие формы по нажатию на данный элемент |

# ОПИСАНИЕ ПРОГРАММЫ

* 1. **Общие сведения**

Наименование программы – Клавиатурный тренажер для программиста. Обозначение программы – КТДП. Программа написана на языке программирования C# в среде Visual Studio. Программа работает под управлением операционной системы Windows 7/10 c целевой рабочей средой .NET Framework версии 4.7.2.

* 1. **Функциональное назначение**

Данный программный продукт разрабатывался для обучения слепому методу печати Пользователя, в частности, задействовать для набора все десять пальцев рук, для увеличения скорости набора ключевых слов языка программирования и для уменьшения количество опечаток при написании программ.

Программа реализует следующие основные группы функций:

* выбор ЯП для выполнения заданий;
* открытие задания из внешнего файла;
* выполнение задания с подсчетом времени выполнения, ошибок и скорости печати;
* вывод результатов выполненного задания.
  1. **Описание логической структуры**

Для детального описания логической структуры работы программы была разработана функциональная схема. Функциональная схема или схема данных (в соответствии с ГОСТ 19. 701-90) – это схема взаимодействия компонентов программного обеспечения с описанием информационных потоков, состава данных в потоках и указанием используемых файлов и устройств.

Для изображения функциональных схем используют специальные обозначения, установленные стандартом. Функциональная схема, разработанная к данному ПО, представлена в Приложении А.

Для подробного описания алгоритма программы была спроектирована блок-схема метода Cvet(), представленная в Приложении Б.

* 1. **Используемые технические средства**

Рассмотрим минимальные системные и программные требования к программе, то есть набор условий, необходимых для возможности запуска и работы программного продукта:

* персональная ЭВМ, клавиатура.
* процессор: тактовая частотой 1 ГГц или выше;
* операционная система: Microsoft Windows ХР или более поздние версии Windows (32 или 64 bit);
* емкость оперативной памяти: не меньше 256 Мб;
* монитор: SVGA 800 x 600 или с более высоким разрешением;
* объем свободного места: ОС Windows 7 – 720 Мб, ОС Windows 10 – 2048 Мб (при установке программы используется около 15 Мб);
* дополнительно требуется установленная целевая рабочая среда .NET Framework не ниже версии 4.0.

Далее рассмотрим рекомендуемые системные и программные требования к программе, то есть набор характеристик, подразумевающих оптимальную работу большей части возможностей продукта:

* операционная система: Microsoft Windows 7/10 (32 или 64 bit);
* процессор: таковая частота 2,5-3,2 Ггц;
* емкость оперативной памяти: 4 Гб;
* объем свободного места: ОС Windows 7 – 1024 Мб, ОС Windows 10 – 2048 Мб (при установке программы используется около 15 Мб);
* монитор;
* мышь, клавиатура.
  1. **Вызов и загрузка**

Для установки данного программного средства необходимо загрузить инсталлятор данного ПО. Запускаем установочный файл. Следуя указаниям инсталлятора, устанавливаем программу.

Шаг 1. Для установки программы следует запустить файл Установка КТДП.exe. Перед началом установки рекомендуется закрыть запущенные приложения. Это позволит программе установки обновить системные файлы без перезагрузки (см. рис. 3.1 и 3.2) Прочитав указания и рекомендации для установки в окне приветствия, нужно продолжить установку, нажав кнопку «Далее».

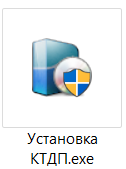


Рисунок 3.1 – Инсталлятор программы

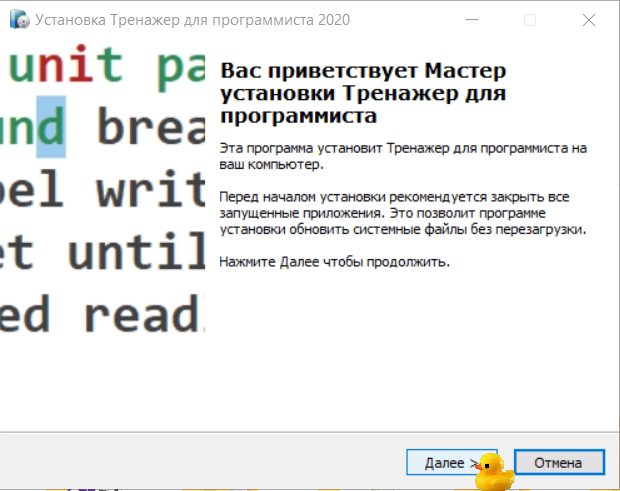


Рисунок 3.2 – Окно приветствия инсталлятора

Шаг 2. В следующем окне необходимо согласиться с лицензионным соглашением, нажав кнопку «Согласен» (см. рис. 3.3).

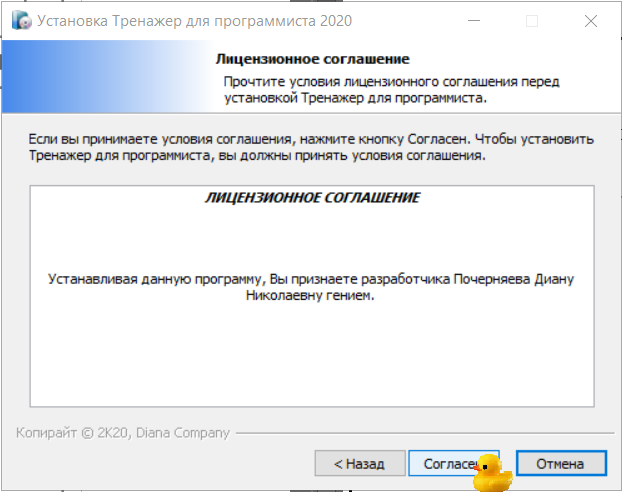


Рисунок 3.3 – Окно лицензионного соглашения

Шаг 3. В следующем окне необходимо выбрать директорию установки, нажав кнопку «Обзор» или же согласиться со стандартным местоположением установки и нажать кнопку «Далее» (см. рис. 3.4).

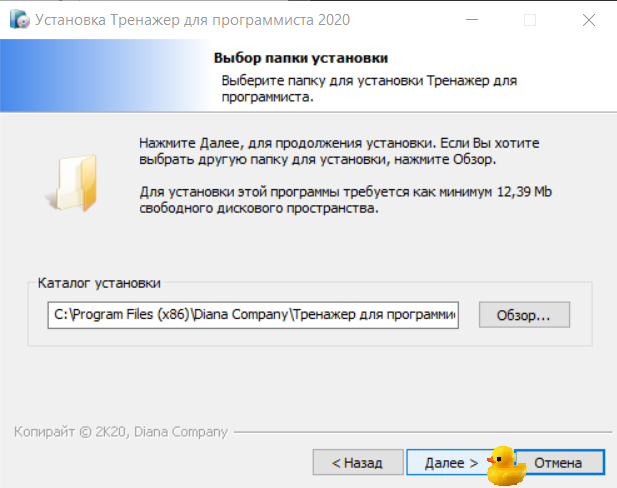


Рисунок 3.4 – Окно выбора папки установки

Шаг 4. В следующем окне необходимо выбрать ярлыки программы, которые необходимо создать, нажав на флажок нужных ярлыков. Для продолжения установки необходимо нажать кнопку «Далее» (см. рис. 3.5).

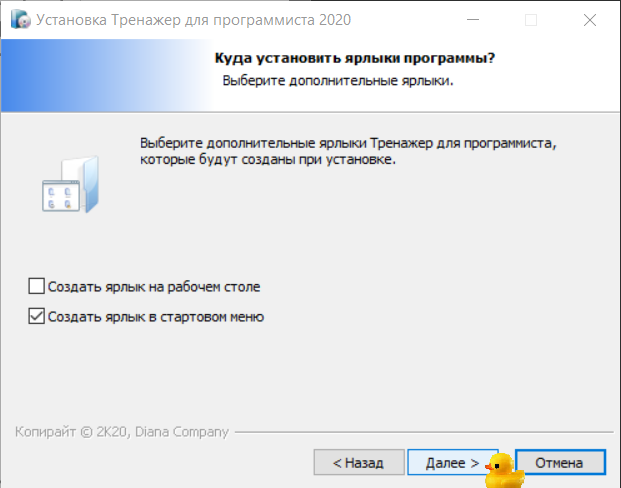


Рисунок 3.5 – Окно выбора ярлыков

Шаг 5. В следующем окне программа готова для установки и необходимо проверить параметры ее установки. Для начала установки нужно нажать кнопку «Установить», после чего начнется процесс установки. Для завершения установки необходимо нажать кнопку «Готово», выбрав необходимые операции, такие как «Запустить Тренажер для программиста», посмотреть ReadMe, представленный справкой или же посетить web-сайт программы, представленный страницей разработчика (см. рис. 3.6).

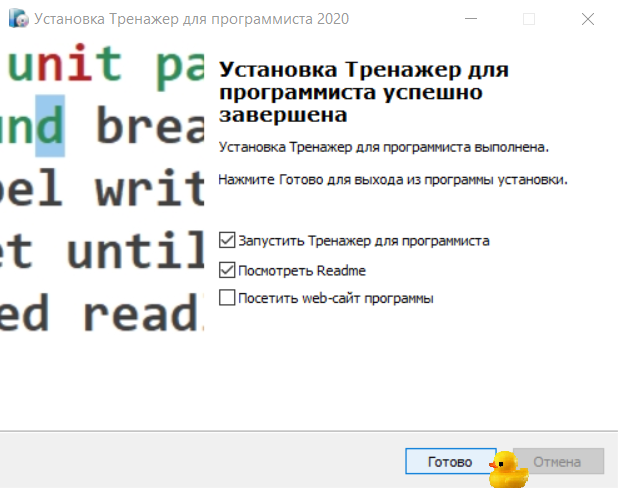


Рисунок 3.6 – Окно выполненной установки

По завершению процесса установки программа развернута и готова к использованию. В меню Пуск и на рабочем столе появится ярлык программы «Тренажер для программиста.exe», через которым можно выполнять запуск программы.

# ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ

* 1. **Цель испытаний**

Тестированием (испытанием) программного обеспечения называется процесс исследования ПО с целью выявления ошибок и определения соответствия между реальным и ожидаемым поведением ПО, осуществляемый на основе набора тестов, выбранных определенным образом. Фаза испытаний начинается, как только все модули отлажены и собраны вместе в одно целое, то есть после полной отладки готового программного продукта.

Тестирование обеспечивает:

* обнаружение ошибок;
* демонстрацию соответствия функций программы ее назначению;
* демонстрацию реализации требований к характеристикам программы;
* отображение надежности как индикатора качества программы.

Целью испытаний является обнаружение ошибок путем реального выполнения специально подобранных контрольных примеров. Шаги процесса задаются тестами.

Полную проверку программы гарантирует исчерпывающее тестирование. Оно требует проверить все наборы исходных данных, все варианты их обработки и включает большое количество тестовых вариантов. Хорошим считают тестовый вариант с высокой вероятностью обнаружения еще не раскрытой ошибки. Успешным называют тест, который обнаруживает до сих пор не раскрытую ошибку.

Для наглядности процесса оценки качества программы была построена IDEF0 диаграмма с помощью CASE-средства BPWin (см. рис. 4.1).

На входе процесса тестирования находятся три потока:

* + текст программы;
  + исходные данные для запуска программы;
  + ожидаемые результаты.

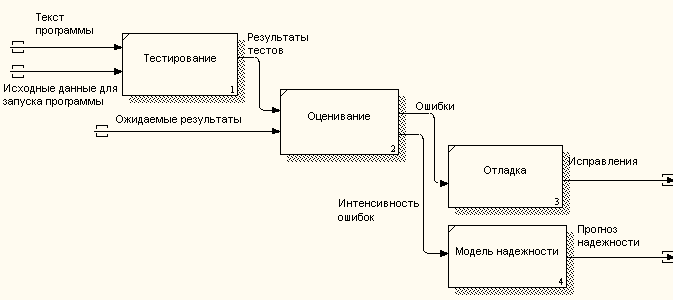


Рисунок 4.1 – IDEF0 диаграмма процесса оценки качества ПО

Выполняются тесты, все полученные результаты оцениваются. Это значит, что реальные результаты тестов сравниваются с ожидаемыми результатами. Когда обнаруживается несовпадение, фиксируется ошибка – начинается отладка. Процесс отладки непредсказуем по времени. Неопределенность в отладке приводит к большим трудностям в планировании действий.

После сбора и оценивания результатов тестирования начинается отображение качества и надежности программного обеспечения. Если регулярно встречаются серьезные ошибки, требующие проектных изменений, то качество и надежность программного обеспечения подозрительны, констатируется необходимость усиления тестирования.

С другой стороны, если функции про­граммного обеспечения реализованы правильно, а обнаруженные ошибки легко исправляются, может быть сделан один из двух выводов:

* + качество и надежность программного обеспечения удовлетворительны;
  + тесты не способны обнаруживать серьезные ошибки.

* 1. **[Описание используемых методов испытаний и их результаты](#_Toc359240384)**

После разработки программного продукта было проведено детальное тестирование на корректность рабочего процесса, на быстроту выполнения программных функций, на удобство используемого интерфейса.

Основная задача тестирования состоит в поиске отсутствующих или неправильно выполняемых функций с целью оценки, насколько хорошо модуль программы отвечает заявленным требованиям. Функциональные тесты подтверждают правильность данных на входе и выходе.

Целью испытания являлось нахождение программных ошибок, которые впоследствии необходимо исправить.

Метод испытания заключался в воссоздании всевозможных исключительных ситуаций и проверки реакции программы на них.

По той причине, что программный продукт не обладает системой ввода входных данных, то протестировать его путем подбора определенных значений – невозможно. Поэтому было принято решение протестировать работоспособность приложения методом «черного ящика».

Метод «черного ящика» подразумевает тестирование внешнего взаимодействия компонентов и их интерфейсов. При использовании этого метода тестирования отсутствует доступ к коду программы.

Для проведения полного функционального тестирования программы написана следующая тестовая документация: чек-лист и тестовые сценарии.

Создание тестовой документации значительно улучшает качество продукта за счет более тесного сотрудничества, уточнения деталей при разработке плана тестирования и документации. После завершения тестирования наличие тестовой документации позволяет проверить, насколько успешно были проведены все этапы тестирования.

Чек-лист (Check List) – высокоуровневый список проверок, набор правил и критериев, по которым проводится тестирование приложения. Описывает основные проверки для типовой функциональности.

Тестовый сценарий (Test Case) – набор входных значений, предусловий выполнения, ожидаемых результатов и постусловий выполнения, разработанный для определенной цели или тестового условия, таких как выполнение определенного пути программы или же для проверки соответствия определенному требованию.

Перед созданием чек-листа была создана карта ПО, для того, чтобы в чек-листе были выделены все операции для тестирования. Карта программы представлена на рисунке 4.1.

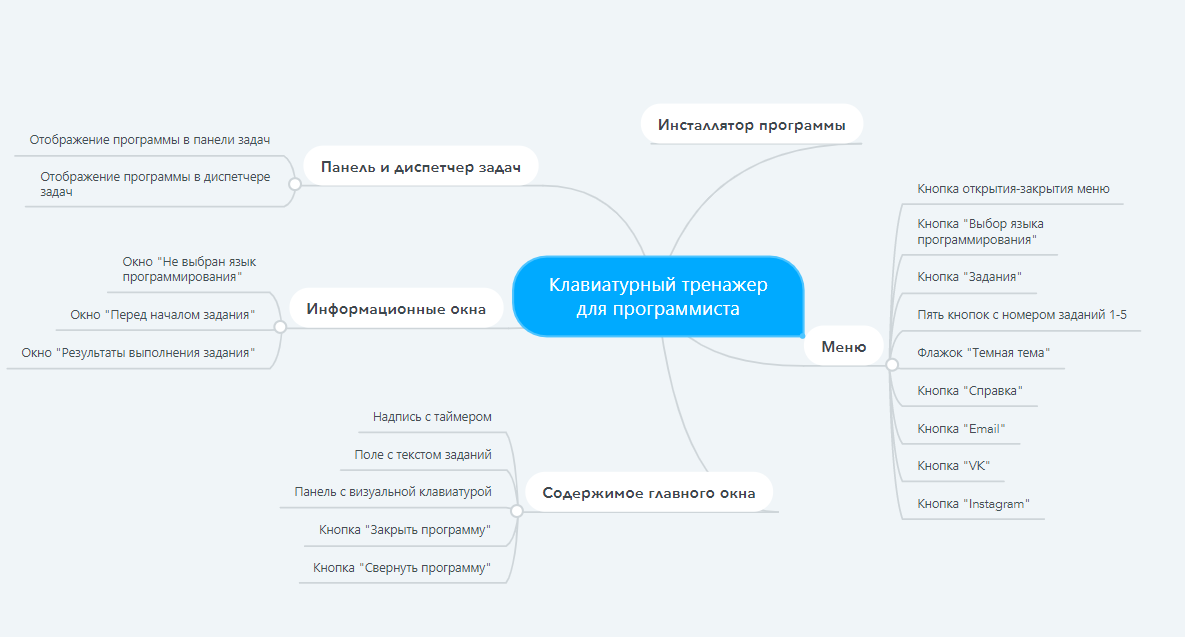


Рисунок 4.1 – Карта программы

Чек-лист, описывающий блоки, которые необходимо протестировать, представлен в таблице 4.1. В чек-листе выделены следующие функциональные блоки, которые необходимо протестировать для определения качества разрабатываемого ПО: инсталляция программы, меню программы, содержимое главного окна, информационные окна, проверка программы в панели задач и диспетчере задач.

Таблица 4.1 – Чек-лист для тестирования данной программы

|  |
| --- |
| **Блок «Инсталляция программы»** |
| Проверка инсталляции программы |
| Проверка деинсталляции программы |
| **Блок «Меню»** |
| Проверка разворачивания меню |
| Проверка сворачивания меню |
| Проверка выбора языка программирования |

*Продолжение таблицы 4.1*

|  |
| --- |
| Проверка работы кнопки «Задания» |
| Проверка работы кнопок с номером задания |
| Проверка смены темы по нажатию флажка «Темная тема» |
| Проверка работы кнопки «Справка» в виде знака «?» |
| Проверка работы кнопки «@» с открытием почтового клиента |
| Проверка работы кнопки с лого VK с открытием страницы разработчика |
| Проверка работы кнопки с лого Instagram с открытием страницы разработчика |
| Проверка работы подсказок на кнопках со связью с разработчиком |
| **Блок «Содержимое главного окна»** |
| Проверка текстового поля, в котором открывается задание |
| Проверка работы панели с визуальной клавиатурой |
| Проверка изменения размеров окна |
| Проверка кнопки «Закрыть программу» в виде «х» |
| Проверка кнопки «Свернуть программу» в виде «–» |
| Проверка разворачивания программы с панели задач |
| Проверка отсчета времени |
| **Блок «Информационные окна»** |
| Проверка работы функционала окна «Не выбран язык программирования» |
| Проверка работы функционала окна «Перед началом задания» |
| Проверка работы функционала окна «Результаты выполнения задания» |
| **Блок «Программа в панели задач и диспетчере задач»** |
| Проверка правильности названия программы и логотипа в панели задач |
| Проверка правильности названия программы и логотипа в диспетчере задач |

После того, как чек-лист составлен и изучен был составлен набор тест-кейсов для полного тестирования разрабатываемого программного обеспечения. Тест-кейс №1 с названием «Инсталляция и деинсталляция программы» представлен в таблице 4.2.

Таблица 4.2 – Тест-кейс №1

|  |  |
| --- | --- |
| Номер | 1 |
| Заголовок | Инсталляция и деинсталляция программы |
| Предусловие | Скачен exe-файл инсталляции программы |
|  |  |
| Шаг | Ожидаемый результат |
| 1 | 2 |
| Открытие инсталлятора | Появилось окно приветствия инсталлятора. В конце процесса установки программа установлена успешно. |
| Запуск программы после установки | Программа запускается. |
| Проверка расположение программы в системе. | Папка с программой находится в указанном месте инсталляции. |
| Проверка установленной программы в меню Пуск | Программа находится в списке программ и ищется по алфавитному указателю. |
| Совершение деинсталляции программы через Панель управления | Файлы с папки с программой удалены. ярлык с рабочего стола исчез, если таковой был создан при инсталляции. Удалена запись из меню Пуск. |

Тест-кейс №2 с названием «Проверка функционала главного окна» представлен в таблице 4.3.

Таблица 4.3 – Тест-кейс №2

|  |  |
| --- | --- |
| Номер | 2 |
| Заголовок | Проверка функционала главного окна |
| Предусловие | Программа открыта |
|  |  |
| Шаг | Ожидаемый результат |
| 1 | 2 |
| В левом части программы нажать на полосу с треугольником «▶» | При наведении на полосу курсор меняется на Pointer. Разворачивается меню программы. |
| Наведение и нажатие кнопки «Свернуть программу» в виде «–» | При наведении на кнопки курсор меняется на Pointer, цвет кнопки на темный. Программа сворачивается в панель задач. При нажатии по программе в панели задач свернутая программа разворачивается |
| Наведение и нажатие кнопки «Закрыть программу» в виде «х» | При наведении на кнопки курсор меняется на Pointer, цвет кнопки на темный. Программа закрывается. |

Тест-кейс №3 с названием «Работоспособность меню» представлен в таблице 4.4.

Таблица 4.4 – Тест-кейс №3

|  |  |
| --- | --- |
| Номер | 3 |
| Заголовок | Работоспособность меню |
| Предусловие | Программа открыта, меню развернуто |
|  |  |
| Шаг | Ожидаемый результат |
| 1 | 2 |
| В правой части меню нажать на полосу с треугольником «◀» | При наведении на полосу курсор меняется на Pointer. Сворачивается меню программы. |
| Наведение и нажатие кнопки «Задания» | При наведении на данный элемент курсор меняется на Pointer. Появляется информационное окно «Не выбран язык программирования». Изменяется цвет свернутого выпадающего списка с языками программирования на красный. |
| Наведение и нажатие по выпадающему списку | При наведении на данный элемент курсор меняется на Pointer. Появляется выпадающий список со всеми языками программирования. |
| Нажатие по пунктам в выпадающем списке | Отображается выбранный пункт в заголовке выпадающего списка. |
| Наведение и нажатие кнопки «Задания» | При наведении на кнопку курсор меняется на Pointer, цвет кнопки на темный. Разворачивается панель с кнопками с номерами заданий от 1 до 5. Если панель развернута при нажатии на кнопку «Задания» панель с кнопками с номерами заданий сворачивается. |

*Продолжение таблицы 4.4*

|  |  |
| --- | --- |
| 1 | 2 |
| Наведение и нажатие флажка «Темное меню» | При наведении на флажок курсор меняется на Pointer.  При выделенном флажке цвет фона программы и меню изменяется на темные тона.  При невыделенном флажке цвет фона программы и меню изменяется на белый и голубой соответственно. |
| Наведение и нажатие кнопки «Справка» в виде «?» | При наведении на кнопку курсор меняется на Pointer, цвет кнопки на темный. В браузере открывается справочное руководство к программе. |
| Наведение и нажатие кнопки «Email» в виде «@» | При наведении на кнопку курсор меняется на Pointer, цвет кнопки на темный, появляется подсказка. Открывается почтовый клиент с окном нового сообщения в адресатах указана почта разработчика, в теме письма указано название программы. |
| Наведение и нажатие кнопки «VK» в виде лого данной социальной сети | При наведении на кнопку курсор меняется на Pointer, цвет кнопки на темный, появляется подсказка. Открывается страница разработчика в социальной сети в браузере. |
| Наведение и нажатие кнопки «Instagram» в виде лого данной социальной сети | При наведении на кнопку курсор меняется на Pointer, цвет кнопки на темный, появляется подсказка. Открывается страница разработчика в социальной сети в браузере. |
| Наведение и нажатие кнопки с номером задания | При наведении на кнопки курсор меняется на Pointer, цвет кнопки на темный. Появляется информационное окно «Перед началом задания». Сворачивается меню. Начинается отчет таймера около надписи «Время» в главном окне. |

Тест-кейс №4 с названием «Выполнение задания» представлен в таблице 4.5.

Таблица 4.5 – Тест-кейс №4

|  |  |
| --- | --- |
| Номер | 4 |
| Заголовок | Выполнение задания |
| Предусловие | Программа открыта, меню развернуто, язык программирования выбран, задания развернуты, пункт выбран, информационное окно «Перед началом задания» закрыто |
|  |  |
| Шаг | Ожидаемый результат |
| 1 | 2 |
| Переключение раскладки клавиатуры на английский язык | Язык системной раскладки переключен на ENG. |
| Проверка работы таймера | Таймер производит отсчет времени в секундах. |
| Проверка отображения задания в текстовом поле | Задание отображено одним цветом. |
| Проверка отображения следующего символа | Над визуальной клавиатурой отображается символ, который должен быть нажат следующим. |

*Продолжение таблицы 4.5*

|  |  |
| --- | --- |
| 1 | 2 |
| Проверка выделения текущего символа в текстовом поле с заданием. | Текущий символ выделен другим цветом. |
| Нажатие клавиш на клавиатуре | При нажатии верной клавиши клавиша на визуальной клавиатуре меняет цвет на зеленый. Надпись со следующим символом меняется на следующий. Символ в текстовом поле меняет цвет на зеленый.  При нажатии неверной клавиши клавиша на визуальной клавиатуре меняет цвет на красный. Надпись со следующим символом меняется на следующий. Символ в текстовом поле меняет цвет на красный. Воспроизводится системный звуковой сигнал ошибки. |
| Прохождение задания до конца | После нажатия последнего символа в задании появляется информационное окно «Результат выполнения задания» с количеством всех символов, ошибок, общим временем и скоростью печати. Таймер остановлен. |

Тест-кейс №5 с названием «Проверка информационного окна» представлен в таблице 4.6.

Таблица 4.6 – Тест-кейс №5

|  |  |
| --- | --- |
| Номер | 5 |
| Заголовок | Проверка информационного окна |
| Предусловие | Программа открыта, меню развернуто, язык программирования выбран, задания развернуты, кнопка с номером задания нажата |
|  |  |
| Шаг | Ожидаемый результат |
| 1 | 2 |
| Нажатие по форме вне информационного окна | Воспроизводится системный звуковой сигнал ошибки, если громкость стоит выше 0. В панели задач мерцанием цвета выделяется программа. |
| Наведение и нажатие кнопки «ОК» | При наведении на кнопки курсор меняется на Pointer, цвет кнопки на темный. Информационное окно закрылось. |

Тест-кейс №6 с названием «Проверка программы в панели задач и диспетчере задач» представлен в таблице 4.7.

Таблица 4.7 – Тест-кейс №6

|  |  |
| --- | --- |
| Номер | 5 |
| Заголовок | Проверка программы в панели задач и диспетчере задач |
| Предусловие | Программа открыта |
|  |  |
| Шаг | Ожидаемый результат |
| 1 | 2 |
| Проверка отображения в панели задач | В панели задач отображается иконка программы, при наведении на программу наименование программы. |

*Продолжение таблицы 4.7*

|  |  |
| --- | --- |
| 1 | 2 |
| Открытие диспетчера задач | В диспетчере задач отображается правильная иконка программы и правильное наименование программы. |

Дефекты, обнаруженные при тестировании программы, должны быть корректно и понятно описаны, чтобы Разработчик смог воспроизвести данный дефект и устранить его. В багтрэкинговых системах для каждого дефекта автоматически генерируется его уникальный номер, в случае использования Microsoft Excel или Word номер дефекту необходимо присваивать вручную.

Пройдя все составленные тест-кейсы, был составлен баг-репорт, представленный в таблице 4.8. Баг-репорт – это документ, описывающий ситуацию или последовательность действий, приведшую к некорректной работе объекта тестирования, с указанием причин и ожидаемого результата.

Таблица 4.8 – Баг-репорт к тестированию программы

|  |  |  |
| --- | --- | --- |
| Наименование дефекта | **№1**  Звуковой сигнал после закрытия окна с результатом прохождения, если последний символ был неправильно нажат | |
| Платформа, операционная система | ПК «Lenovo», операционная система – Windows 10 | |
| Степень критичности | S5 - Тривиальная | |
| Приоритет | P3 - Низкий | |
| Алгоритм воспроизведения: | | |
| 1. Открытие программы, открытие меню программы.  2. Выбор любого языка программирования, открытие любого задания.  3. Прохождение задания до конца с нажатием последнего символа в задании неверно.  4. Закрытие информационного окна с результатом прохождения задания. | | |
| Фактический результат | | Ожидаемый результат |
| Воспроизведение звукового сигнала совершения ошибки | | Воспроизведения звукового сигнала совершения ошибки нет |
|  | | |
| Наименование дефекта | **№2**  Орфографическая ошибка в справочном руководстве | |
| Платформа, операционная система | ПК «Lenovo», операционная система – Windows 7 | |
| Степень критичности | S5 - Тривиальная | |
| Приоритет | P2 - Средний | |
| Алгоритм воспроизведения: | | |
| 1. Открытие программы, открытие меню программы.  2. Открытие справки.  3. Переход на раздел «Описание программы»  4. Переход к последнему абзацу раздела. | | |
| Фактический результат | | Ожидаемый результат |
| курсовой | | курсовое |

*Продолжение таблицы 4.8*

|  |  |  |
| --- | --- | --- |
|  | | |
| Наименование дефекта | **№3**  Неправильный цвет фона надписи «Время» после смены светлой темы на темную | |
| Платформа, операционная система | ПК «Lenovo», операционная система – Windows 10 | |
| Степень критичности | S4 - Незначительная | |
| Приоритет | P2 - Средний | |
| Алгоритм воспроизведения: | | |
| 1. Открытие программы, открытие меню программы.  2. Смена светлой темы на темную нажатием на флажок «Темная тема».  3. Выбор любого языка программирования, открытие любого задания.  4. Проверка надписи «Время» | | |
| Фактический результат | | Ожидаемый результат |
| Неправильный цвет фона надписи | | Цвет фона надписи совпадает с цветом окна |
|  | | |
| Наименование дефекта | **№4**  Курсор Pointer у некликабельного элемента «Темная тема» | |
| Платформа, операционная система | ПК «Lenovo», операционная система – Windows 7 | |
| Степень критичности | S5 - Тривиальная | |
| Приоритет | P2 - Средний | |
| Алгоритм воспроизведения: | | |
| 1. Открытие программы, открытие меню программы.  2. Наведение курсора на надпись «Темная тема» | | |
| Фактический результат | | Ожидаемый результат |
|  | | Курсор должен оставаться Arrow |

Дефекты, найденные и описанные в баг-репорте, представленном в таблице 4.8, были исправлены. Повторно пройденные все составленные тест-кейсы были успешно завершены.

Также для разработанного ПО были проведено нагрузочное тестирование с помощью профилировщика производительности, предоставляемым Microsoft Visual Studio. Итоговый результат тестирования представлен на рисунке 4.2.

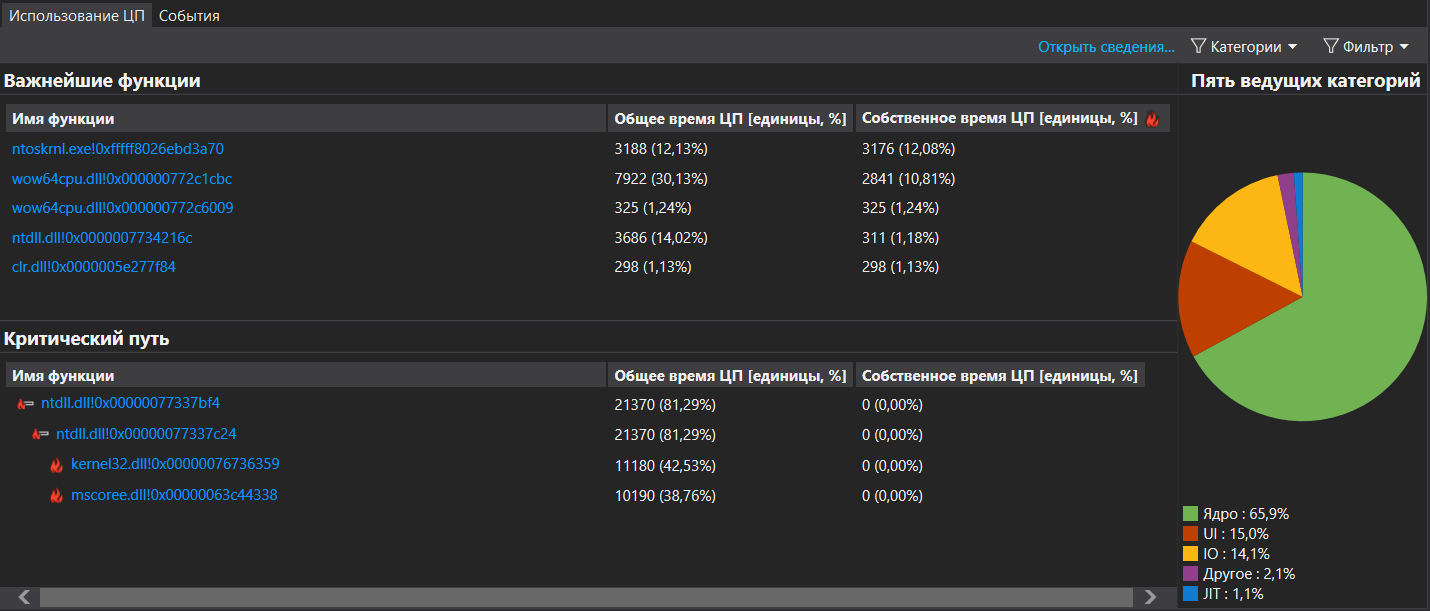


Рисунок 4.2 – Тест использования центрального процессора программой

По итогам нескольких видов тестирования можно сделать вывод, что разработанное программное обеспечение «Клавиатурный тренажер для программиста» разработан качественно и готово к использованию.

# ОПИСАНИЕ ПРИМЕНЕНИЯ

* 1. **Назначение программы**

Назначением программы является возможность совершенствовать скорость набора ключевых выражений определенных языков программирования. Конечными пользователями являются программисты, которые ведут разработку на одном из таких ЯП, как С#, Pascal, Python, JavaScript, PL/SQL. Программа разработана с понятным и интуитивным интерфейсом, поэтому не возникнет сложности с обучением слепой печати в данной программе.

* 1. **Условия выполнения программы**

Рассмотрим минимальные системные и программные требования к программе, то есть набор условий, необходимых для возможности запуска и работы программного продукта:

* персональная ЭВМ, клавиатура.
* процессор: тактовая частотой 1 ГГц или выше;
* операционная система: Microsoft Windows ХР или более поздние версии Windows (32 или 64 bit);
* емкость оперативной памяти: не меньше 256 Мб;
* монитор: SVGA 800 x 600 или с более высоким разрешением;
* объем свободного места: ОС Windows 7 – 720 Мб, ОС Windows 10 – 2048 Мб (при установке программы используется около 50 Мб);
* дополнительно требуется установленная целевая рабочая среда .NET Framework не ниже версии 4.0.
  1. **Выполнение программы**
     1. Запуск программы

Для запуска прикладной программы для начала необходимо выполнить установку программы, следуя инструкциям инсталлятора, начальное окно приветствия которого представлено на рисунке 5.1.

После завершения установки запустить программу двойным нажатием левой кнопки мыши по ярлыку программы, созданному в меню или на рабочем столе (см. рис. 5.2). После запуска откроется главное окно программы, на котором будут доступны все функции программы

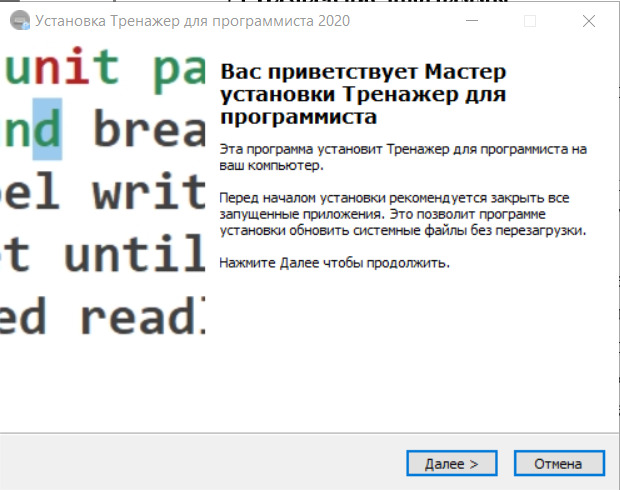


Рисунок 5.1 – Инсталлятор программы

.

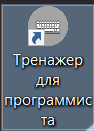


Рисунок 5.2 – Ярлык программы на рабочем столе

* + 1. Выполнение программы

После открытия программы, в запускаемом по умолчанию главном окне в левой части окна находится полоса меню, которое открывается по нажатию на полосу. Свернутое меню представлено на рисунке 5.3, а развернутое меню показано на рисунке 5.4.

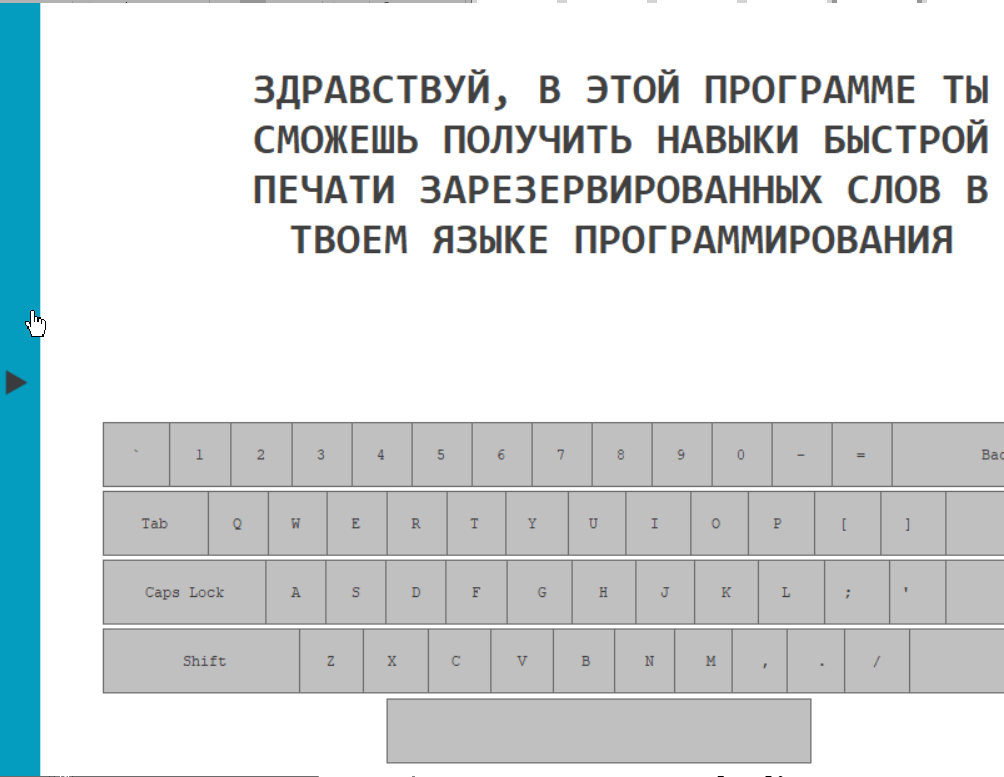


Рисунок 5.3 – Меню в свернутом состоянии



Рисунок 5.4 – Меню в развернутом состоянии

Для того, чтобы начать выполнение заданий, необходимо выбрать язык программирования, на котором Пользователь хочет выполнять задания (см. рис. 5.5). В выпадающем списке представлены такие языки программирования, как C#, Pascal, JavaScript, Python, PL/SQL.

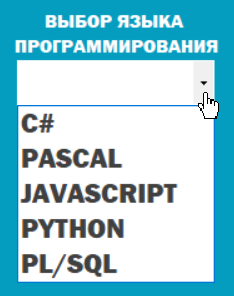


Рисунок 5.5 – Выбор языка программирования

Далее для выбора задания необходимо нажать на кнопку «Задания» и будет развернут список заданий с различным уровнем сложности (см. рис. 5.6). Задание 1 состоит из ключевых слов с минимальным количеством символов в слове, а задание 5 соответственно состоит из ключевых слов с максимальным количеством символов в слове.



Рисунок 5.6 – Развернутый список заданий

Если же язык программирования выбран не будет при нажатии на кнопку «Задания» будет отображено информационное окно с сообщением, представленное на рисунке 5.7.

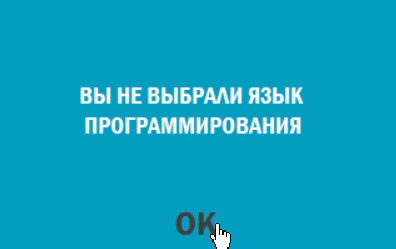


Рисунок 5.7 – Информационное окно о невыбранном ЯП

Перед началом задания также будет отображено информационное окно с объяснением прохождения задания (см. рис. 5.8).

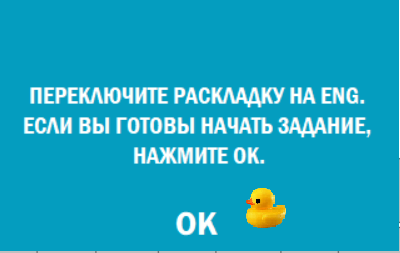


Рисунок 5.8 – Информационное окно о невыбранном ЯП

После нажатия кнопки «Ок» в окне, представленном на рисунке 5.8 начнется прохождение задания. Раскладка клавиатуры должна быть переключена на английский язык. Выполнение задания состоит в том, что Пользователь должен печатать текст, показанный над клавиатурой в окне, смотря на символ, который далее следует или только на текст задания.

При совершении ошибки, то есть нажатие клавиши на клавиатуре не соответствующей текущему символу текста прозвучит системный звук ошибки. Демонстрация прохождения задания представлена на рисунке 5.9.

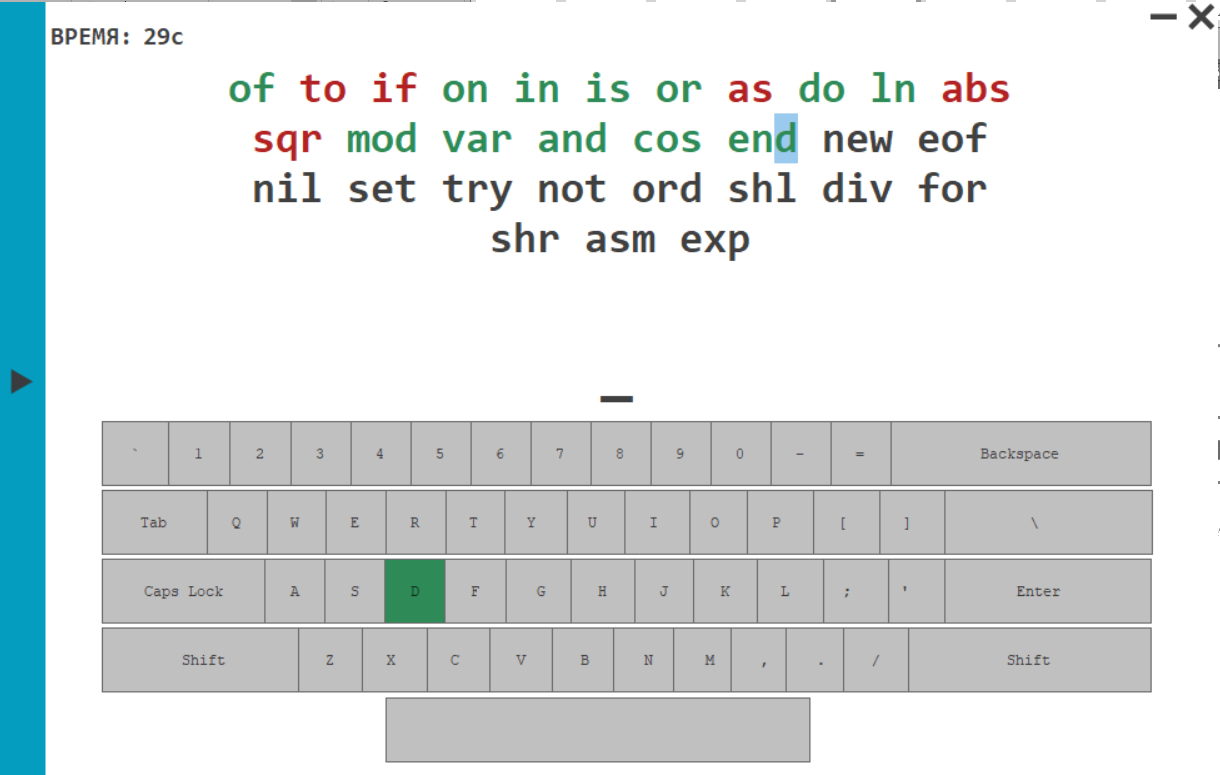


Рисунок 5.9 – Прохождение задания

По итогу выполнения задания, то есть при окончании печати всего текста задания отображается окно с результатами выполнения, такими как общее количество символов, допущенные ошибки, итоговое время и скорость печати, выраженная в символах в минуту (см. рис. 5.10).

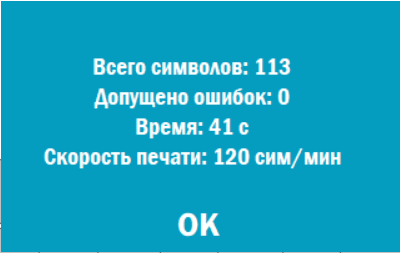


Рисунок 5.10 – Окно с результатами выполнения

Для того, чтобы сменить цветовую тему программы нужно нажать на флажок под надписью «Темная тема» в нижней части меню. При активированном флажке тема сменяется на темную, при деактивированном флажке тема меняется на светлую. Демонстрация смены тем представлена на рисунках 5.11 и 5.12.

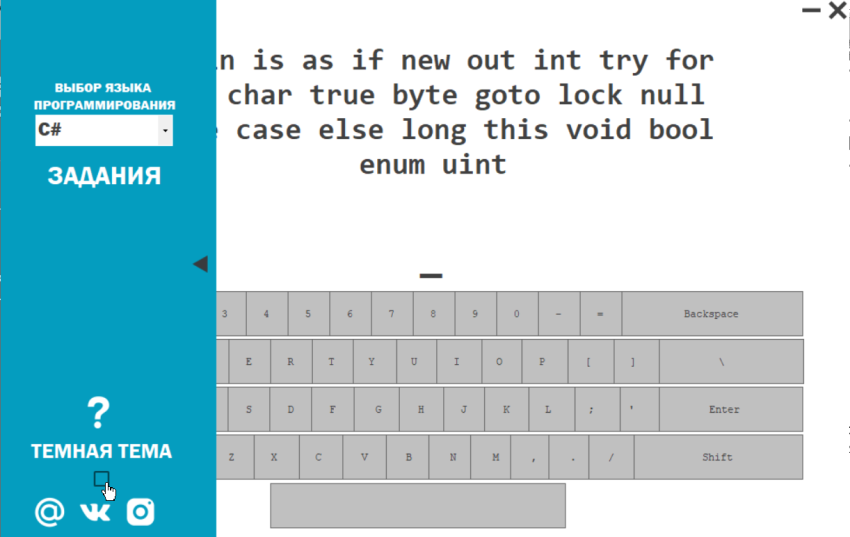


Рисунок 5.11 – Светлая тема

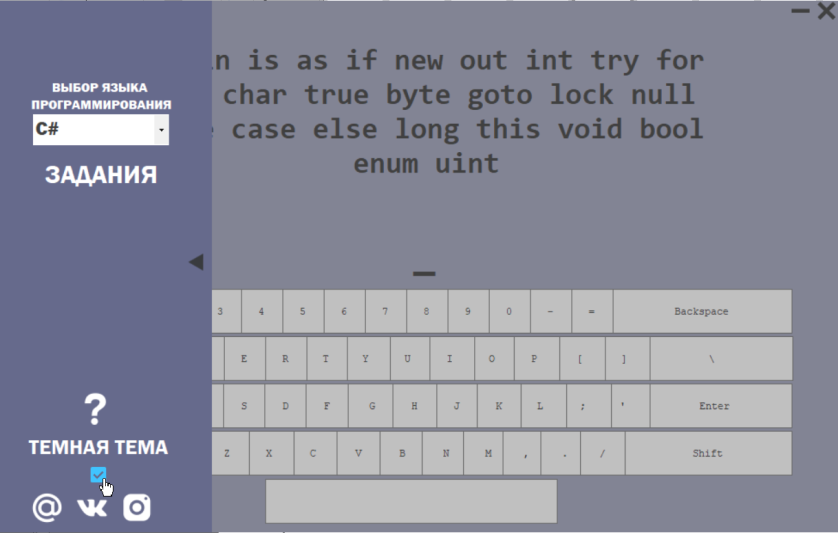


Рисунок 5.12 – Темная тема

При возникновении необходимости в обучении работы с программой нужно нажать на кнопку с знаком вопроса, которая откроет справку (см. рис. 5.13).

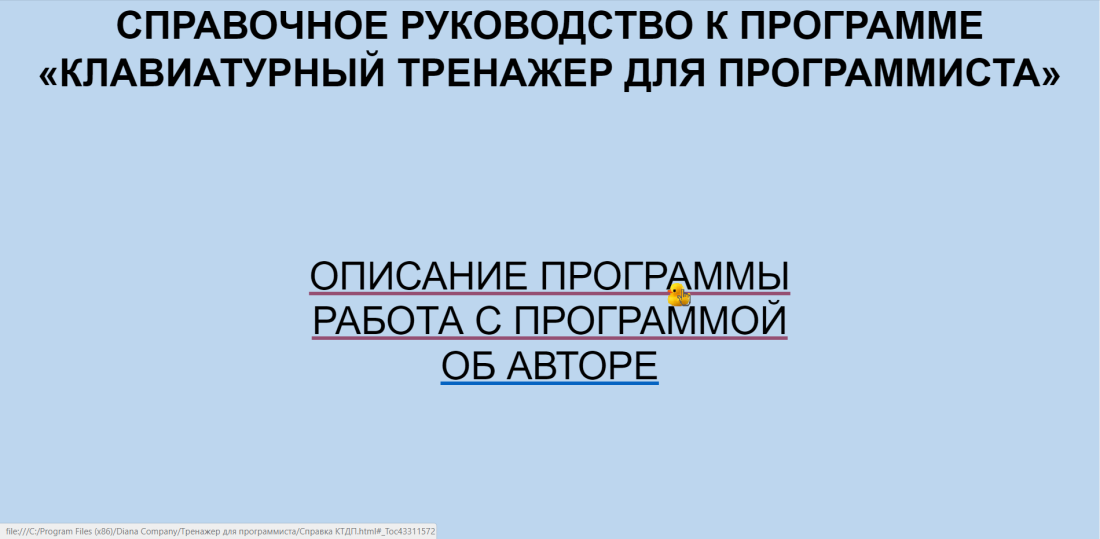


Рисунок 5.13 – Справка в виде файла html

Также при нахождении ошибки в программе или необходимости связи с разработчиком можно нажать на соответствующие кнопки внизу меню программы, которая откроет почтовый клиент или страницы разработчика в социальных сетях (см. рис. 5.14).



Рисунок 5.14 – Кнопки для связи с разработчиком

* + 1. Завершение работы программы

По окончанию работы с программой можно ее свернуть в панель задач или же завершить с ней работу. Для того, чтобы свернуть программу необходимо нажать на кнопку «Свернуть» в виде «–» в верхнем правом углу окна.

Для завершения работы с программой нужно нажать кнопку «х» также находящуюся в верхнем правом углу программы (см. рис. 5.14) или же закрыть программу на панели задач также нажав на кнопку «х» (см. рис. 5.15).



Рисунок 5.14 – Кнопки сворачивания и закрытия программы

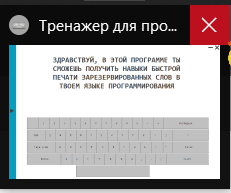


Рисунок 5.15 – Кнопка закрытия программы на панели задач

* 1. **Сообщения оператору**

К сообщениям оператору будут относится информационные окна, представленные на рисунках выше 5.7, 5.8 и 5.10.

Сообщение, представленное на рисунке 5.7, отображается при нажатии на кнопку «Задания», но при невыбранном языке программирования в выпадающем списке.

Сообщение, представленное на рисунке 5.8, отображается перед началом выполнения задания с просьбой переключения раскладки и возможностью подготовки комфортной окружающей среды Пользователя для выполнения задания.

Сообщение, представленное на рисунке 5.10, отображается по окончанию выполнения задания с результирующими данными такими как общее количество напечатанных символом, количество допущенных ошибок, итоговое время выполнения и скорость печати, выраженная в символах в минуту.

# ЗАКЛЮЧЕНИЕ

Качество и скорость разработки программного обеспечения зависит от степени знания зарезервированных слов языка программирования. Правильность составления инструкций зависит от знания ключевых слов конкретного языка программирования. Для запоминания и увеличения скорости набора и была создана данная прикладная программа. Данная программа может применяться начинающими программистами, желающими улучшить навыки в конкретном языке программирования. Программа предоставляет возможность выбора языка программирования для выполнения заданий среди перечисленных: C#, Pascal, Python, JavaScript, PL/SQL.

Цель курсового проекта – создание прикладной программы «Клавиатурный тренажер для программиста» – была достигнута. Средой разработки данного программного обеспечения был выбрана Visual Studio и язык программирования C#, а также шаблон Windows Forms. Целевой рабочей средой является .NET Framework 4.7.2, то есть предполагается выполнение программы на операционных системах выше Windows 7.

Были решены следующие задачи, в соответствии с целью:

1) подробна изучена литература по разработке приложений на шаблоне Windows Forms, найдены, рассмотрены и изучены материалы, связанные с темой разработки программы;

2) получены навыки в разработке программного обеспечения на языке программирования C# на шаблоне Windows Forms, задействована работа с внешними библиотеками и дополнительные расширения, была изучена работа с Git – системой управления версиями;

3) детально создан дизайн в одной стилистике и обеспечена интуитивная навигация по программе.

4) был подробно проанализирована задача, спроектирован, спрототипирован, закодирован, протестирован и отлажен рабочий проект и реализована стабильная версия программы версии 1.0 с датированным наименованием версии 2020. В процессе тестирования были исправлены ошибки при работе с программой, созданы окна подсказок, предусмотрены меры, чтобы уберечь Пользователя от некорректных действий;

5) были изучены правила написания технической документации на программные средства. Создана интуитивно понятная справочная система.

Достоинствами программы являются:

1) возможность выполнения заданий на различных языках программирования.

2) наличие уровней сложности заданий, начинающиеся с ключевых слов с минимальным количеством символов и заканчивающийся ключевыми словами с максимальным количеством символов.

3) невысокие системные требования.

4) привлекательный и оригинальный дизайн программы.

6) простота и понятность использования программы.

Программа отлажена, протестирована и готова для демонстрации и практического использования. Пояснительная записка написана в соответствии с ГОСТами и требованиями, предъявляемыми к технической документации.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

***Нормативная документация:***

1. ГОСТ 19.102-77 – Единая система программной документации. Стадии разработки.
2. ГОСТ 19.104-78 – Единая система программной документации. Основные надписи.
3. ГОСТ 19.105-78 – Единая система программной документации. Общие требования к программным документам.
4. ГОСТ 19.106-78. Единая система программной документации. Требования к программным документам, выполненным печатным способом.
5. ГОСТ 19.201-78 – Единая система программной документации. Техническое задание. Требования к содержанию и оформлению.
6. ГОСТ 19.301-2000 – Единая система программной документации. Программа и методика испытаний.
7. ГОСТ 19.502-78 – Единая система программной документации. Описание применения. Требования к содержанию и оформлению.
8. ГОСТ 19.505-79 – Единая система программной документации. Руководство оператора.
9. ГОСТ 19.701-90 – Единая система программной документации. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения.
10. ГОСТ 19.781-90 – Единая система программной документации. Термины и определения.
11. ГОСТ 2.105-95 – Единая система конструкторской документации. Общие требования к текстовым документам.
12. ГОСТ 34.603-92 – Информационная технология. Виды испытаний автоматизированных систем.
13. ГОСТ 19.402-2000 Единая система конструкторской документации. Описание программы.
14. ГОСТ 19.701-90 Единая система конструкторской документации. Схемы алгоритмов, программ, данных и систем.
15. ISO/IEC/IEEE 29119-1:2013 Software and systems engineering – Software testing – Part 1: Concepts and definitions.
16. ISO/IEC/IEEE 29119-2:2013 Software and systems engineering – Software testing – Part 2: Test processes.
17. ISO/IEC/IEEE 29119-3:2013 Software and systems engineering – Software testing – Part 3: Test documentation.

***Основная документация:***

1. Буч Градди Объектно-ориентированный анализ и проектирование с примерами приложений, 3-е изд. / Буч Градди, Максимчук Роберт А., Энгл Майкл У., Янг Бобби Дж., Коналлен Джим, Хьюстон Келли А.: Пер с англ. – М.: ООО «И.Д. Вильямс», 2010. – 720 с.
2. Вигерс К. Разработка требований к программному обеспечению. 3-е изд., дополненное. Пер. с англ. / К. Вигерс. – М.: Издательство «Русская редакция»; СПб.: БХВ-Петербург, 2014. – 736 стр.
3. Скит Джон.С# для профессионалов: тонкости программирования, 3-е изд. : Пер. с англ. – М. : ООО “И.Д. Вильямс”, 2014. – 608 с.: ил. – Парал. тит. англ.
4. Чакон Скотт, Страуб Бен Git для профессионального программиста. – СПб.: Питер, 2016. – 496 с.: ил.

***Интернет-источники:***

1. Имена объектов и зарезервированные слова [Электронный ресурс] – Режим доступа: http://plsqlbook.ru/object-names-and-reserved-words/ – Дата доступа: 12.03.2020.
2. Ключевые слова в JavaScript [Электронный ресурс] – Режим доступа: https://alekseygulynin.ru/klyuchevye-slova-v-javascript/ – Дата доступа: 12.03.2020.
3. Ключевые слова C# [Электронный ресурс] – Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/ – Дата доступа: 13.03.2020
4. Ключевые слова, модуль keyword [Электронный ресурс] – Режим доступа: https://pythonworld.ru/osnovy/klyuchevye-slova-modul-keyword.html – Дата доступа: 12.03.2020.
5. Правильно пишем тест-кейсы. Памятка начинающему специалисту по тестированию [Электронный ресурс] – Режим доступа: https://victorz.ru/202001101079 – Дата доступа: 20.05.2020.
6. Руководство по программированию на C# [Электронный ресурс] – Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/csharp/programming-guide/ – Дата доступа: 04.05.2020.
7. Словарь языка Pascal [Электронный ресурс] – Режим доступа: https://pas1.ru/pascaldictionary – Дата доступа: 13.03.2020.

# ПРИЛОЖЕНИЕ А

БЛОК-СХЕМА ПРОГРАММЫ



Рисунок А.1 – Блок-схема программы

# ПРИЛОЖЕНИЕ Б

БЛОК-СХЕМА МЕТОДА CVET



Рисунок А.1 – Блок-схема методы Cvet

# ПРИЛОЖЕНИЕ В

ЛИСТИНГ ПРОГРАММЫ

**Форма Trenazher (Главное окно программы)**

using System;

using System.Drawing;

using System.IO;

using System.Windows.Forms;

using AudioSwitcher.AudioApi.CoreAudio;

using System.Diagnostics;

using System.Drawing.Text;

namespace PodobieKursovogo

{

#region ПОЯСНЕНИЕ К КОМПОНЕНТАМ

/\* При объявлении компонентов использовались сокращенные названия элементов

\* и пояснительное имя элемента.

\* cb - ComboBox, CheckBox

\* flp - FlowLayoutPanel

\* l - Label

\* p - Panel

\* pb - PictureBox

\* rt - RichTextBox

\* t - Timer

\* tb - TextBox

\* zhad\_1, zhad\_2 и т.д. являются Label, используются как кнопки для вызова заданий

\*/

#endregion

public partial class Trenazher : Form

{

private int next = -1; //унарный

private Button knop; //кнопки клавиатуры на p\_klav

private int timer;

private int pravilno;

private int nepravilno;

private bool menu = false;

private bool prover = false;

//всплывающие подсказки

private readonly ToolTip t = new ToolTip();

//управление системным звуком

private readonly CoreAudioDevice zvuk = new CoreAudioController().DefaultPlaybackDevice;

//экземпляр класса-формы с выводом сообщений

readonly MessageBox m = new MessageBox();

public Trenazher()

{

InitializeComponent();

}

private void Trenazher\_Load(object sender, EventArgs e)

//происходит при создании формы

{

svetlaya\_thema(); //вызов светлой темы

rtZhadanie.ForeColor = Color.FromArgb(64, 64, 64);

rtZhadanie.SelectionAlignment = HorizontalAlignment.Center;

menu = false;

tbPole.TabIndex = 0;

rtZhadanie.Text = "ЗДРАВСТВУЙ, В ЭТОЙ ПРОГРАММЕ ТЫ СМОЖЕШЬ ПОЛУЧИТЬ НАВЫКИ БЫСТРОЙ ПЕЧАТИ ЗАРЕЗЕРВИРОВАННЫХ СЛОВ В ТВОЕМ ЯЗЫКЕ ПРОГРАММИРОВАНИЯ";

}

#region метод Chtenie

private string Chtenie(string f) //метод, который считывает информацию из файла с заданием

{

FileStream f1;

StreamReader f2;

f1 = new FileStream(f, FileMode.Open, FileAccess.Read);

f2 = new StreamReader(f1);

string text = f2.ReadToEnd();

f1.Close();

/\*for (int i = 0; i < text.Length; i++)

{

rtZhadanie.ForeColor = Color.FromArgb(64, 64, 64);

}\*/

return text;

}

#endregion

#region метод Otkritie

private void Otkritie(string f) //метод, начинающий задание

{

try

{

knop.BackColor = Color.Silver;

}

catch (Exception ex) { }

next = -1;

timer = 0;

pravilno = 0;

nepravilno = 0;

rtZhadanie.Focus();

tbPole.Clear();

tbPole.Text = Chtenie(f);

m.l\_mes1.Visible = true;

m.l\_mes2.Visible = false;

m.l\_mes3.Visible = false;

m.l\_mes1.Text = "ПЕРЕКЛЮЧИТЕ РАСКЛАДКУ НА ENG.\nЕСЛИ ВЫ ГОТОВЫ НАЧАТЬ ЗАДАНИЕ,\n НАЖМИТЕ ОК.";

m.ShowDialog(); //окно сообщения

t\_vremya.Start();

lVr.Visible = true;

}

#endregion

#region метод Cvet

private void Cvet(Button clav, char[] s, int next, char key)

{ //метод, меняющий цвет нажатых клавиш и выделение цвета в поле печати

try

{

knop.BackColor = Color.Silver;

}

catch (Exception e) { }

try

{

if (s[next + 1] == ' ')

{

lsled.Text = "\_";

}

else

{

lsled.Text = Convert.ToString(s[next + 1]);

}

if (key == s[next])

{

zvuk.Volume = 0;

clav.BackColor = Color.SeaGreen;

knop = clav;

rtZhadanie.SelectionStart = next;

rtZhadanie.SelectionLength = 1;

rtZhadanie.SelectionColor = Color.SeaGreen;

pravilno++;

}

else

{

zvuk.Volume = 20;

clav.BackColor = Color.Firebrick;

knop = clav;

rtZhadanie.SelectionStart = next;

rtZhadanie.SelectionLength = 1;

rtZhadanie.SelectionColor = Color.Firebrick;

nepravilno++;

}

}

catch (Exception e) { }

}

#endregion

#region метод KeyPress

private void rtZhadanie\_KeyPress(object sender, KeyPressEventArgs e)

//отслеживает нажатие клавиш и производит подсчет результата

{

char key = e.KeyChar;

char[] s = tbPole.Text.ToCharArray(); // s-символ

next++; //ничего не оптимизируй, все работает

for (int i = 0; i < p\_klav.Controls.Count; i++)

{

if (p\_klav.Controls[i].Name.Substring(0, 2).ToLower().CompareTo("b" + key.ToString().ToLower())

== 0 && p\_klav.Controls[i].Name.Length <= 2)

Cvet((Button)p\_klav.Controls[i], s, next, key);

}

if (key == 32) Cvet(bspacebar, s, next, key);

/\*if (key == 33) Cvet(b1, s, next, key);

if (key == 64) Cvet(b2, s, next, key);

if (key == 35) Cvet(b3, s, next, key);

if (key == 36) Cvet(b4, s, next, key);

if (key == 37) Cvet(b5, s, next, key);

if (key == 94) Cvet(b6, s, next, key);

if (key == 38) Cvet(b7, s, next, key);

if (key == 42) Cvet(b8, s, next, key);

if (key == 40) Cvet(b9, s, next, key);

if (key == 41) Cvet(b0, s, next, key);

if (key == 13) Cvet(benter, s, next, key);

\*/

if (s.Length == next + 1)

{

t\_vremya.Stop();

int vsego = pravilno + nepravilno;

double sk = vsego / timer \* 60;

int skorost = (int)sk;

m.l\_mes2.Visible = false;

m.l\_mes1.Visible = false;

m.l\_mes3.Visible = true;

m.l\_mes3.Text="Всего символов: " + vsego + "\nДопущено ошибок: " + nepravilno

+ "\nВремя: " + timer + " c" + "\nСкорость печати: " + skorost + " сим/мин";

m.ShowDialog(); //окно сообщения

}

}

#endregion

#region Отсчет времени

private void t\_vremya\_Tick(object sender, EventArgs e)

//производит отсчет времени при выполнении задания

{

timer += 1;

lVremya.Text = timer.ToString() + "с";

}

#endregion

#region Работа с меню

private void t\_menu\_Tick(object sender, EventArgs e)

//движение меню по таймеру

//Диана, если ты вдруг неудачно отладила...верни начальное положение на -287;0

{

if ((menu) && (pMenu.Left < 0))

pMenu.Left = pMenu.Left + 5;

if ((!menu) && (pMenu.Left > -210))

pMenu.Left = pMenu.Left - 5;

}

private void pb\_ukaz\_Click(object sender, EventArgs e)

//сворачивание, разворачивание меню

{

t\_menu.Start();

menu = !menu;

if (menu)

pb\_ukaz.Image = Properties.Resources.стрелочка\_назад as Bitmap;

else

{

pb\_ukaz.Image = Properties.Resources.стрелочка as Bitmap; //меняем стрелочку

p\_zhadan.Size = p\_zhadan.MinimumSize;

}

}

private void Form1\_MouseMove(object sender, MouseEventArgs e)

//сворачивание меню, если мышь активна на форме, а не на меню

{

if (menu)

pb\_ukaz\_Click(sender, e);

}

#endregion

#region Разворачивание заданий

private void t\_zhadan\_Tick(object sender, EventArgs e)

//работа с выезжающими заданиями по таймеру

{

if (prover)

{

p\_zhadan.Height += 10;

if (p\_zhadan.Size == p\_zhadan.MaximumSize)

{

t\_zhadan.Stop();

prover = false;

}

}

else

{

p\_zhadan.Height -= 10;

if (p\_zhadan.Size == p\_zhadan.MinimumSize)

{

t\_zhadan.Stop();

prover = true;

}

}

}

#endregion

#region Открытие заданий

private void zhad\_1\_Click(object sender, EventArgs e)

//открытие заданий по нажатию задания 1

{

rtZhadanie.SelectionStart = 0;

rtZhadanie.SelectionLength = 1;

pb\_ukaz\_Click(sender, e);

switch (cb\_lang.SelectedIndex) //проверка на выбранный язык

{

case 0:

rtZhadanie.Text = Chtenie("c# 1.txt");

Otkritie("c# 1.txt");

break;

case 1:

rtZhadanie.Text = Chtenie("pas 1.txt");

Otkritie("pas 1.txt");

break;

case 2:

rtZhadanie.Text = Chtenie("js 1.txt");

Otkritie("js 1.txt");

break;

case 3:

rtZhadanie.Text = Chtenie("py 1.txt");

Otkritie("py 1.txt");

break;

case 4:

rtZhadanie.Text = Chtenie("sql 1.txt");

Otkritie("sql 1.txt");

break;

}

}

private void zhad\_2\_Click(object sender, EventArgs e)

//открытие заданий по нажатию задания 2

{

rtZhadanie.SelectionStart = 0;

rtZhadanie.SelectionLength = 1;

pb\_ukaz\_Click(sender, e);

switch (cb\_lang.SelectedIndex)

{

case 0:

rtZhadanie.Text = Chtenie("c# 2.txt");

Otkritie("c# 2.txt");

break;

case 1:

rtZhadanie.Text = Chtenie("pas 2.txt");

Otkritie("pas 2.txt");

break;

case 2:

rtZhadanie.Text = Chtenie("js 2.txt");

Otkritie("js 2.txt");

break;

case 3:

rtZhadanie.Text = Chtenie("py 2.txt");

Otkritie("py 2.txt");

break;

case 4:

rtZhadanie.Text = Chtenie("sql 2.txt");

Otkritie("sql 2.txt");

break;

}

}

private void zhad\_3\_Click(object sender, EventArgs e)

//открытие заданий по нажатию задания 3

{

rtZhadanie.SelectionStart = 0;

rtZhadanie.SelectionLength = 1;

pb\_ukaz\_Click(sender, e);

switch (cb\_lang.SelectedIndex)

{

case 0:

rtZhadanie.Text = Chtenie("c# 3.txt");

Otkritie("c# 3.txt");

break;

case 1:

rtZhadanie.Text = Chtenie("pas 3.txt");

Otkritie("pas 3.txt");

break;

case 2:

rtZhadanie.Text = Chtenie("js 3.txt");

Otkritie("js 3.txt");

break;

case 3:

rtZhadanie.Text = Chtenie("py 3.txt");

Otkritie("py 3.txt");

break;

case 4:

rtZhadanie.Text = Chtenie("sql 3.txt");

Otkritie("sql 3.txt");

break;

}

}

private void zhad\_4\_Click(object sender, EventArgs e)

//открытие заданий по нажатию задания 4

{

rtZhadanie.SelectionStart = 0;

rtZhadanie.SelectionLength = 1;

pb\_ukaz\_Click(sender, e);

switch (cb\_lang.SelectedIndex)

{

case 0:

rtZhadanie.Text = Chtenie("c# 4.txt");

Otkritie("c# 4.txt");

break;

case 1:

rtZhadanie.Text = Chtenie("pas 4.txt");

Otkritie("pas 4.txt");

break;

case 2:

rtZhadanie.Text = Chtenie("js 4.txt");

Otkritie("js 4.txt");

break;

case 3:

rtZhadanie.Text = Chtenie("py 4.txt");

Otkritie("py 4.txt");

break;

case 4:

rtZhadanie.Text = Chtenie("sql 4.txt");

Otkritie("sql 4.txt");

break;

}

}

private void zhad\_5\_Click(object sender, EventArgs e)

//открытие заданий по нажатию задания 5

{

rtZhadanie.SelectionStart = 0;

rtZhadanie.SelectionLength = 1;

pb\_ukaz\_Click(sender, e);

switch (cb\_lang.SelectedIndex)

{

case 0:

rtZhadanie.Text = Chtenie("c# 5.txt");

Otkritie("c# 5.txt");

break;

case 1:

rtZhadanie.Text = Chtenie("pas 5.txt");

Otkritie("pas 5.txt");

break;

case 2:

rtZhadanie.Text = Chtenie("js 5.txt");

Otkritie("js 5.txt");

break;

case 3:

rtZhadanie.Text = Chtenie("py 5.txt");

Otkritie("py 5.txt");

break;

case 4:

rtZhadanie.Text = Chtenie("sql 5.txt");

Otkritie("sql 5.txt");

break;

}

}

#endregion

#region Жмяканье кнопок

private void pb\_sprav\_Click(object sender, EventArgs e)

//вызов справочки

{

Process.Start("Справка КТДП.html");

}

private void pb\_sprav\_MouseHover(object sender, EventArgs e)

//подсказочка для справочки

{

t.SetToolTip(pb\_sprav, "Открыть справочное руководство");

}

private void l\_zhad\_Click(object sender, EventArgs e)

//открытие панели с заданиями

{

if (cb\_lang.SelectedIndex == -1)

{

m.l\_mes1.Visible = false;

m.l\_mes2.Visible = true;

m.l\_mes3.Visible = false;

m.l\_mes2.Text = "ВЫ НЕ ВЫБРАЛИ ЯЗЫК \nПРОГРАММИРОВАНИЯ";

m.ShowDialog();

cb\_lang.BackColor = Color.FromArgb(255, 128, 128);

}

else

t\_zhadan.Start();

}

private void pb\_email\_Click(object sender, EventArgs e)

//открытие почтового клиента

{

string mail = string.Format("mailto:di.in.it@ya.ru?subject=Клавитурный%20тренажер%20для%20программиста&body=Спасибо%20за%20данную%20программу!!!");

Process.Start(mail);

}

private void pb\_email\_MouseHover(object sender, EventArgs e)

//подсказочка для отрытия емейла

{

t.SetToolTip(pb\_email, "Написать разработчику");

}

private void pb\_inst\_MouseHover(object sender, EventArgs e)

//подсказочка для инстаграма

{

t.SetToolTip(pb\_inst, "Открыть Instagram разработчика");

}

private void pb\_vk\_MouseHover(object sender, EventArgs e)

//подсказочка для вк

{

t.SetToolTip(pb\_vk, "Открыть ВК разработчика");

}

private void pb\_vk\_Click(object sender, EventArgs e)

//открытие веб-страницы вк

{

Process.Start("https://vk.com/d\_44\_p");

}

private void pb\_inst\_Click(object sender, EventArgs e)

//открытие веб-страницы инстаграм

{

Process.Start("https://www.instagram.com/d.44.p/");

}

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

//смена цветовой темы

{

if (cbThema.Checked)

{

temnaya\_thema();

}

else

{

svetlaya\_thema();

}

}

private void pb\_minim\_Click(object sender, EventArgs e)

//сворачивание программы

{

WindowState = FormWindowState.Minimized;

}

private void pb\_exit\_Click(object sender, EventArgs e)

//закрытие программы

{

Application.Exit();

}

#endregion

#region Дезигн (изменение состояние при наведении) Label-ов и PictureBox-ов

/\* MouseMove - происходит при наведении на элемент

\* MouseLeave - происходит при отведении от элемента

\*/

private void l\_zhad\_MouseLeave(object sender, EventArgs e)

{

l\_zhad.ForeColor = Color.White;

}

private void l\_zhad\_MouseMove(object sender, MouseEventArgs e)

{

l\_zhad.ForeColor = Color.FromArgb(64, 64, 64);

}

private void zhad\_1\_MouseMove(object sender, MouseEventArgs e)

{

zhad\_1.ForeColor = Color.FromArgb(64, 64, 64);

}

private void zhad\_1\_MouseLeave(object sender, EventArgs e)

{

zhad\_1.ForeColor = Color.White;

}

private void zhad\_2\_MouseMove(object sender, MouseEventArgs e)

{

zhad\_2.ForeColor = Color.FromArgb(64, 64, 64);

}

private void zhad\_2\_MouseLeave(object sender, EventArgs e)

{

zhad\_2.ForeColor = Color.White;

}

private void zhad\_3\_MouseMove(object sender, MouseEventArgs e)

{

zhad\_3.ForeColor = Color.FromArgb(64, 64, 64);

}

private void zhad\_3\_MouseLeave(object sender, EventArgs e)

{

zhad\_3.ForeColor = Color.White;

}

private void zhad\_4\_MouseLeave(object sender, EventArgs e)

{

zhad\_4.ForeColor = Color.White;

}

private void zhad\_4\_MouseMove(object sender, MouseEventArgs e)

{

zhad\_4.ForeColor = Color.FromArgb(64, 64, 64);

}

private void zhad\_5\_MouseMove(object sender, MouseEventArgs e)

{

zhad\_5.ForeColor = Color.FromArgb(64, 64, 64);

}

private void zhad\_5\_MouseLeave(object sender, EventArgs e)

{

zhad\_5.ForeColor = Color.White;

}

private void pb\_email\_MouseMove(object sender, MouseEventArgs e)

{

pb\_email.Image = Properties.Resources.письмо1 as Bitmap;

}

private void pb\_email\_MouseLeave(object sender, EventArgs e)

{

pb\_email.Image = Properties.Resources.письмо as Bitmap;

}

private void pb\_sprav\_MouseMove(object sender, MouseEventArgs e)

{

pb\_sprav.Image = Properties.Resources.помощь1 as Bitmap;

}

private void pb\_sprav\_MouseLeave(object sender, EventArgs e)

{

pb\_sprav.Image = Properties.Resources.помощь as Bitmap;

}

private void pb\_vk\_MouseLeave(object sender, EventArgs e)

{

pb\_vk.Image = Properties.Resources.вк as Bitmap;

}

private void pb\_vk\_MouseMove(object sender, MouseEventArgs e)

{

pb\_vk.Image = Properties.Resources.вк1 as Bitmap;

}

private void pb\_inst\_MouseMove(object sender, MouseEventArgs e)

{

pb\_inst.Image = Properties.Resources.инста1 as Bitmap;

}

private void pb\_inst\_MouseLeave(object sender, EventArgs e)

{

pb\_inst.Image = Properties.Resources.инста as Bitmap;

}

private void cb\_lang\_MouseMove(object sender, MouseEventArgs e)

{

cb\_lang.BackColor = Color.White;

}

private void pb\_minim\_MouseMove(object sender, MouseEventArgs e)

{

pb\_minim.Image = Properties.Resources.свернуть1 as Bitmap;

}

private void pb\_minim\_MouseLeave(object sender, EventArgs e)

{

pb\_minim.Image = Properties.Resources.свернуть as Bitmap;

}

private void pb\_exit\_MouseMove(object sender, MouseEventArgs e)

{

pb\_exit.Image = Properties.Resources.закрыть1 as Bitmap;

}

private void pb\_exit\_MouseLeave(object sender, EventArgs e)

{

pb\_exit.Image = Properties.Resources.закрыть as Bitmap;

}

#endregion

#region Смена цветовой темы

private void svetlaya\_thema()

//светлая тема цветовой схемы программы

{

BackColor = Color.White;

p\_klav.BackColor = Color.White;

rtZhadanie.BackColor = Color.White;

pb\_minim.BackColor = Color.White;

pb\_exit.BackColor = Color.White;

lVremya.BackColor = Color.White;

lVr.BackColor = Color.White;

pMenu.BackColor = Color.FromArgb(4, 157, 191);

lThema.BackColor = Color.FromArgb(4, 157, 191);

lvibor.BackColor = Color.FromArgb(4, 157, 191);

m.BackColor = Color.FromArgb(4, 157, 191);

}

private void temnaya\_thema()

//темная тема цветовой схемы программы

{

BackColor = Color.FromArgb(130, 132, 148);

p\_klav.BackColor = Color.FromArgb(130, 132, 148);

rtZhadanie.BackColor = Color.FromArgb(130, 132, 148);

pb\_minim.BackColor = Color.FromArgb(130, 132, 148);

pb\_exit.BackColor = Color.FromArgb(130, 132, 148);

lVremya.BackColor = Color.FromArgb(130, 132, 148);

pMenu.BackColor = Color.FromArgb(102, 106, 140);

lVr.BackColor = Color.FromArgb(130, 132, 148);

lsled.ForeColor = Color.FromArgb(64, 64, 64);

lThema.BackColor = Color.FromArgb(102, 106, 140);

lvibor.BackColor = Color.FromArgb(102, 106, 140);

m.BackColor = Color.FromArgb(102, 106, 140);

}

#endregion

}

}

**Форма MessageBox (Информационное окно)**

using System;

using System.Drawing;

using System.Drawing.Text;

using System.Windows.Forms;

namespace PodobieKursovogo

{

public partial class MessageBox : Form

{

public MessageBox()

{

InitializeComponent();

}

private void l\_ok\_MouseMove(object sender, MouseEventArgs e)

{

l\_ok.ForeColor = Color.FromArgb(64,64,64);

}

private void l\_ok\_MouseLeave(object sender, EventArgs e)

{

l\_ok.ForeColor = Color.White;

}

private void l\_ok\_Click(object sender, EventArgs e)

{

Close();

}

}

}