*Министерство образования и молодежной политики   
Свердловской области*

*ГАПОУ СО «Асбестовский политехникум»*

*Специальность 09.02.07 «Информационные системы и программирование»*

*ТЕМА КУРСОВОГО*

*ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА КОмпЬЮТЕРНОЙ ИГРЫ «ВИСЕЛИЦА»*

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

*КП ИС 2025.01 ПЗ*

|  |  |
| --- | --- |
| *Руководитель проекта*  *Петрова Е.С.*  *\_\_\_\_\_\_\_.2025* | *Студент группы ИС-22*  *Бадретдинов К.Д.*  *\_\_\_\_\_\_\_.2025* |

*2025*

*СОДЕРЖАНИЕ*

[введение 3](#_Toc190695510)

[1 постановка и проектирование задачи 5](#_Toc190695511)

[1.1 Описание программы 5](#_Toc190695512)

[1.2 Функциональные возможности 5](#_Toc190695513)

[1.3 Прототипы интерфейса 6](#_Toc190695514)

[2 кодирование программного продукта 8](#_Toc190695515)

[2.1 Проверка введенного слова\буквы на совпадения с секретным словом 8](#_Toc190695516)

[2.2 Вывод совпавших букв в соответствующее поле на форме 9](#_Toc190695517)

[2.3 Вывод не совпавших букв в соответствующее поле на   
форме 10](#_Toc190695518)

[Заключение 11](#_Toc190695519)

[Список литературы 12](#_Toc190695520)

# введение

Актуальность данного проекта обусловлена его высокой образовательной ценностью. Разработка игры "Виселица" на C# в среде Visual Studio предоставляет отличную возможность для практического применения знаний в области программирования. Проект охватывает ключевые аспекты, такие как работа со строками, массивами, условными операторами и циклами, а также позволяет изучить принципы объектно-ориентированного программирования. Создание пользовательского интерфейса с использованием Windows Forms способствует развитию навыков разработки интерактивных приложений. В условиях постоянно растущего спроса на квалифицированных программистов, этот проект является эффективным инструментом для обучения и повышения компетенций.

Объект исследования: процесс разработки и написания компьютерной игры “Виселица”.

Предмет исследования: компьютерная игра.

Цель проекта: разработка и написание кода для компьютерной игры “Виселица” в среде программирования Visual Studio на языке C#.

Для достижения поставленной цели сформулированы следующие задачи:

* анализ предметной области;
* определение правила игры в “Виселица”;
* определение необходимых функций программы по ранее определенным правилам;
* разработка функций для программы;
* верстка и кодирование программного продукта;
* тестирование и отладка программного продукта.

Анализ аналогов компьютерной игры «Виселица» был проведен на следующих сайтах: startgamer.ru, igroutka.ru, rebuskids.ru.

Были выявлены следующие общие черты:

* все три сайта предлагают классическую реализацию игры "Виселица" с угадыванием слова по буквам и ограничением по количеству ошибок;
* все игры представлены на русском языке;
* ни один из сайтов не предлагает дополнительных функций, таких как выбор тем, уровней сложности, подсказок или статистики;
* все три игры имеют достаточно простые и понятные интерфейсы, без сложных элементов управления или навигации;
* все три игры доступны для бесплатного использования;
* все три игры используют изображение виселицы, на которой появляется человечек при неправильных ответах;
* управление осуществляется выбором букв.

# постановка и проектирование задачи

## Описание программы

Область применения: программный продукт предназначен для развлекательных и развивающих целей.

Игроку необходимо угадать и ввести слово в специальное поле в течение одиннадцати попыток. Пользователь может вводить как буквы, так и целые слова. Если введенные буквы совпадают с буквами в секретном слове, они отображаются в поле совпавших букв. Буквы, которые введены, но отсутствуют в секретном слове, помещаются в поле не совпавших букв. Если же введенное слово или буква не содержат хотя бы одной совпадающей буквы, игрок получает одну ошибку. Игра заканчивается при достижении 11 ошибок.

## Функциональные возможности

В приложении необходимо реализовать следующий функционал:

* выбор случайного слова;
* ввод от пользователя;
* отображение угаданных букв в соответствующем поле;
* отображение не угаданных букв в соответствующем поле;
* отображение загаданного слова в виде символов нижнего подчеркивания “\_” (один символ нижнего подчеркивания эквивалентен одной букве загаданного слова).

## Прототипы интерфейса

Прототип интерфейса стартового экрана представлен на рисунке 1.

Прототип интерфейса основного экрана, предназначенного для игры представлен на рисунке 2.



Рисунок 1 – Прототип интерфейса стартового экрана

На рисунке 1 предоставлен экран после запуска приложения, на котором расположены 3 кнопки:

* Старт – при нажатии на кнопку вам откроется экран с игрой;
* Правила – при нажатии на кнопку вам откроется экран с правилами игры;
* Выход – при нажатии на кнопку программа завершится.

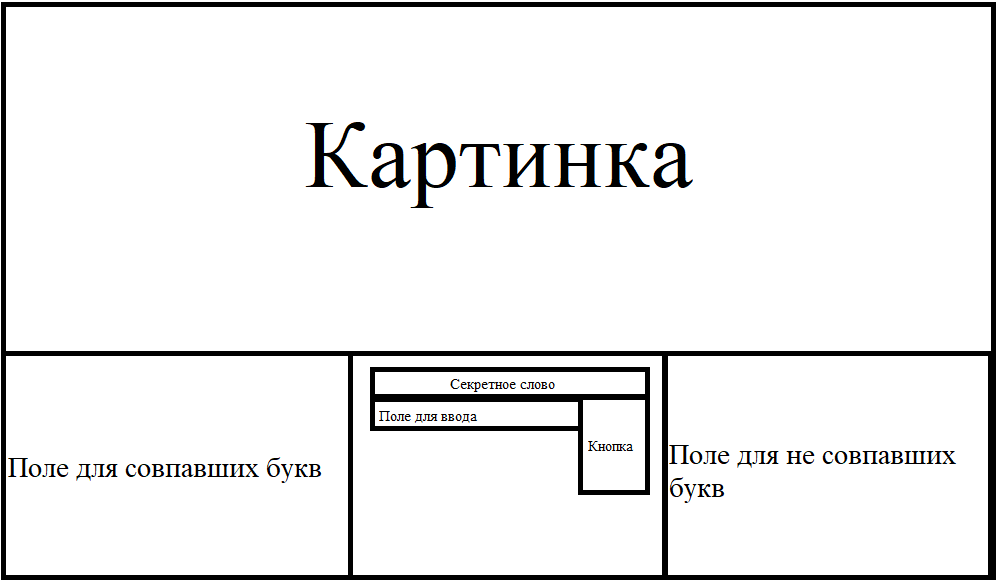


Рисунок 2 - Прототип интерфейса игрового экрана

На рисунке 2 предоставлен экран после нажатия на кнопку «играть» с экрана на рисунке 1. При открытии игрового экрана в разделе «Секретное слово» будет показано загаданное слово, но вместо букв вы увидите символы нижнего подчеркивания. Чтобы начать игру, нужно ввести букву или слово в «Поле для ввода» и нажать «Кнопку». Если в результате проверки совпавших букв не окажется, вам будет начислена ошибка, и все несовпадающие буквы отобразятся в «Поле для не совпавших букв». Однако, если совпавшие буквы найдены, ошибка начислена не будет, и буквы будут перенесены в «Поле для совпавших букв».

# кодирование программного продукта

## Проверка введенного слова\буквы на совпадения с секретным словом

public void Check(string word)

{

char[] secretWordChar = SecretWord.Text.ToLower().Trim().ToCharArray();

char[] inputWordChar = word.ToLower().Trim().ToCharArray();

// Внешний цикл проходит по каждому символу inputWordChar

for (int j = 0; j < inputWordChar.Length; j++)

{

// Флаг для отслеживания совпадений

bool matchFound = false;

for (int i = 0; i < secretWordChar.Length; i++)

{

if (inputWordChar[j] == secretWordChar[i])

{

// Если совпадение найдено и буква еще не добавлена

// Добавляем и проверяем уникальность

if (matchedLettersSet.Add(inputWordChar[j]))

{

// Устанавливаем флаг, если совпадение найдено

matchFound = true;

}

}

}

// Если совпадений не найдено, добавляем буквы во множество несовпадений

if (!matchFound)

{

errorInAttempt = true;

//Если буква есть в совпавших добавление не произойдет

if (!matchedLettersSet.Contains(inputWordChar[j]))

{

// Добавляем букву из вводимого слова

noMatchedLettersSet.Add(inputWordChar[j]);

}

}

}

if(errorInAttempt)

{

error++;

imgUpdate(error);

errorInAttempt = false;

}

// Если длины разные, слова не идентичны

if (inputWordChar.Length == secretWordChar.Length)

{

int lenght = 0;

// Сравниваем символы по позициям

for (int i = 0; i < inputWordChar.Length; i++)

{

if (inputWordChar[i] == secretWordChar[i])

{

lenght++;

if (lenght == secretWordChar.Length)

{

FormWin formWin = new FormWin();

FormMain formMain = new FormMain();

formMain.Hide();

Refresh();

Thread.Sleep(500);

formWin.Show();

}

}

}

}

// Обновление меток с использованием методов OutMatchingLetters и OutNoMatchingLetters

OutMatchingLetters(matchedLettersSet);

OutNoMatchingLetters(noMatchedLettersSet);

}

## Вывод совпавших букв в соответствующее поле на форме

public void OutMatchingLetters(HashSet<char> matchedLettersSet)

{

// Очистка текстового поля перед обновлением

labelOutMatchingLetters.Text = string.Empty;

// Выводим буквы в текстовое поле для совпадающих букв

foreach (char letter in matchedLettersSet)

{

labelOutMatchingLetters.Text += letter + ", ";

}

// Убираем последний символ ", " если есть буквы

if (matchedLettersSet.Count > 0)

{

labelOutMatchingLetters.Text = labelOutMatchingLetters.Text.TrimEnd(',', ' ');

}

}

## Вывод не совпавших букв в соответствующее поле на форме

public void OutNoMatchingLetters(HashSet<char> noMatchedLettersSet)

{

// Очистка текстового поля перед обновлением

labelOutNoMatchingLetters.Text = string.Empty;

// Выводим буквы в текстовое поле для несовпадающих букв

foreach (char letter in noMatchedLettersSet)

{

labelOutNoMatchingLetters.Text += letter + ", ";

}

// Убираем последний символ ", " если есть буквы

if (noMatchedLettersSet.Count > 0)

{

labelOutNoMatchingLetters.Text = labelOutNoMatchingLetters.Text.TrimEnd(',', ' ');

}

}

# Заключение

В программе реализованы следующий функционал:

* выбор случайного слова из XML файла;
* подсчет ошибок;
* простой и интуитивно понятный интерфейс;
* обновление изображение при начислении ошибки;
* ввод от пользователя;
* проверка вводимого слова на идентичность секретному слову;
* отображение угаданных букв в соответствующем поле;
* отображение не угаданных букв в соответствующем поле;
* отображение загаданного слова в виде символов нижнего подчеркивания “\_” (один символ нижнего подчеркивания эквивалентен одной букве загаданного слова).

Для выполнения работы использовалась среда разработки Visual Studio и тип проекта Windows Forms. Данная среда программирования была выбрана из-за ее мощных инструментов и удобного интерфейса, а также было интересно попробовать создать на ней курсовой проект.

В программе хорошо получилось реализовать проверку вводимого слова на наличие совпадений и последующий вывод совпавших и не совпавших букв в соответствующие поля на форме. Бесспорно, в программе есть недостатки. Например, можно оптимизировать процесс проверки вводимого слова на совпадения, более лаконично и логически правильно написать сам код программы.

Для улучшения программного продукта можно добавить возможность выбора сложности, длины и темы слов.

# Список литературы

1. Справочник и самоучитель C, C++, C# [Электронный ресурс] – Электрон. текст. дан. – Евгений Попов, 2011 - по сей день. – Режим доступа: https://metanit.com, свободный. – Загл. с экрана.
2. Справочник и самоучитель C# [Электронный ресурс] – Электрон. текст. дан. –Корпорация Microsoft, 2019 - по сей день. – Режим доступа: https://learn.microsoft.com свободный. – Загл. с экрана.
3. Справочник и самоучитель C# [Электронный ресурс] – Электрон. текст. дан. – Дэн Шефер, 2001 - по сей день. – Режим доступа: https://learn.microsoft.com свободный. – Загл. с экрана.