Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

**ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ**

**УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)**

Кафедра автоматизации обработки информации (АОИ)

**Игра «Придумай слова»**

Отчет к курсовой работе

по дисциплине

«Объектно-ориентированное программирование»

Выполнилстудент гр. 427-1

\_\_\_\_\_\_\_\_\_\_\_\_\_Петрова Е.А.

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2019 г.

Руководитель

Доцент каф. АОИ, к.т.н.

\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_ Морозова Ю. В.

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

Томск 2019

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

**ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ**

**УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)**

УТВЕРЖДАЮ

Зав. кафедрой АОИ

Профессор

\_\_\_\_\_\_Ю. П. Ехлаков

\_\_\_\_\_\_\_\_\_\_\_ 2019 г.

Кафедра автоматизации обработки информации (АОИ)

ЗАДАНИЕ

на курсовую работу по дисциплине

«Объектно-ориентированное программирование»

Студент: Петрова Елена Александровна

Группа: 427-1 Факультет: систем управления

Тема проекта: Объектно-ориентированное программирование

Исходные данные к проекту: методическое пособие; дополнительные информационные ресурсы.

Срок сдачи студентом законченного проекта: 27.05.2019

Перечень подлежащих разработке вопросов: анализ предметной области; анализ существующих аналогов; обоснование выбора среды разработки приложения; программная реализация; тестирование приложения; описание интерфейса приложения.

Дата выдачи задания: 11.02.2019

Руководитель:

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

Доцент каф. АОИ ТУСУР  Морозова Ю.В.

Задание принял к исполнению 28.02.2019 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оглавление

[Введение 4](#_Toc65160608)

[1. Теоретические основы 5](#_Toc65160609)

[1.1 Общие сведения 5](#_Toc65160610)

[1.2 Язык программирование и среда разработки 5](#_Toc65160611)

[2. Программная реализация 7](#_Toc65160612)

[2.1 Диаграмма классов 7](#_Toc65160613)

[3. Тесты программы 9](#_Toc65160614)

[3.1 Unit-тесты 9](#_Toc65160615)

[3.2 Покрытие кода 10](#_Toc65160616)

[4. Интерфейс программы 11](#_Toc65160617)

[4.1 Общий интерфейс программы 11](#_Toc65160618)

[4.2 Интерфейс игры «Придумай слова» на примере 12](#_Toc65160619)

[Заключение 14](#_Toc65160620)

[Список источников 15](#_Toc65160621)

[Приложение А 16](#_Toc65160622)

[Код программы 16](#_Toc65160623)

[Приложение Б 21](#_Toc65160624)

[Unit - тесты 21](#_Toc65160625)

Введение

На данный период времени разработка является актуальной и имеет перспективы для улучшения. Актуальность темы разработки обусловлена малым количеством аналогов игры «Придумай слова», способствующей развитию внимания, активизированию мышления и пополнению словарного запаса.

Основной целью данной работы является закрепление знаний и навыков по дисциплине Объектно-ориентированное программирование и анализ. Для этого была разработана компьютерная игра «Придумай слова», позволяющая развивать логику и мышление.

При выполнении работы необходимо было решить следующие задачи:

* выбрать язык для программирования и среду разработки;
* реализовать поиск из списка слов, которые могут быть получены из слова – образца;
* реализовать «дружественный» графический интерфейс;
* произвести тестирование созданного приложения.

1. Теоретические основы
   1. **Общие сведения**

Эта программа является игрой «Придумай слова», предназначена для познавательных задач, при работе с которой улучшаются знания слов русского языка. Программа имеет русский язык и интуитивно понятный интерфейс, с которым легко и приятно работать. В игре «Найди слово» пользователь загадывает слово-образец, после чего он должен найти все возможное слова из загаданного слова. Если пользователь правильно отгадал слово, то оно окрашивается в зелёный, в противном случае – в красный. Играющий имеет право в любой момент начать новую игру.

* 1. **Язык программирование и среда разработки**

Для решения поставленной задачи был выбран язык программирования C#. Этот язык удобен и прост для создания данной игры, а также является объектно-ориентированным языком, соответствуя требованию курсовой. С# подходит для разработки продукта на Windows, т.к. поддерживается Microsoft и постоянно развивается, благодаря этой компании. Язык имеет конструкции, которые созданы для облегчения написания и понимания кода, что упрощает работу с продуктом. Немаловажным фактором является наличие большого количества литературы для изучения данного языка, что способствует успешному изучению языка и написанию курсовой работы.

Средой разработки я выбрала Visual Studio. Это обусловлено следующими факторами:

* интуитивный стиль кодирования. По умолчанию Visual Studio форматирует код по мере его ввода, автоматически вставляя необходимые отступы и применяя цветовое кодирование для выделения элементов типа комментариев. Такие незначительные отличия делают код более удобным для чтения и менее подверженным ошибкам;
* более высокая скорость разработки. Многие из функциональных возможностей Visual Studio направлены на то, чтобы помогать разработчику делать свою работу как можно быстрее. Удобные функции, вроде функции IntelliSense (которая умеет перехватывать ошибки и предлагать правильные варианты), функции поиска и замены (которая позволяет отыскивать ключевые слова как в одном файле, так и во всем проекте) позволяют разработчику работать быстро и эффективно;
* возможности отладки. Предлагаемые в Visual Studio инструменты отладки являются наилучшим средством для отслеживания загадочных ошибок и диагностирования странного поведения.

1. Программная реализация
   1. **Диаграмма классов**

Диаграмма классов показывает классы и их иерархию, тем самым представляя логическую сторону проекта. Диаграмма классов занимает центральное место в проектировании объектно-ориентированной системы. Нотация классов используется на разных этапах проектирования и строится с различной степенью детализации. Язык UML применяется не только для проектирования, но и с целью документирования проекта.

Основными элементами являются классы и связи между ними. Классы характеризуются при помощи атрибутов и операций

**Виды связей:**

* ассоциация — это структурная связь между элементами модели, которая описывает набор связей, существующих между объектами.   
  Ассоциация показывает, что объекты одной сущности (класса) связаны с объектами другой сущности таким образом, что можно перемещаться от объектов одного класса к другому;
* агрегация– особая разновидность ассоциации, представляющая структурную связь целого с его частями;
* наследование– это связь, при которой специализированный элемент (потомок) строится по спецификациям обобщенного элемента (родителя).

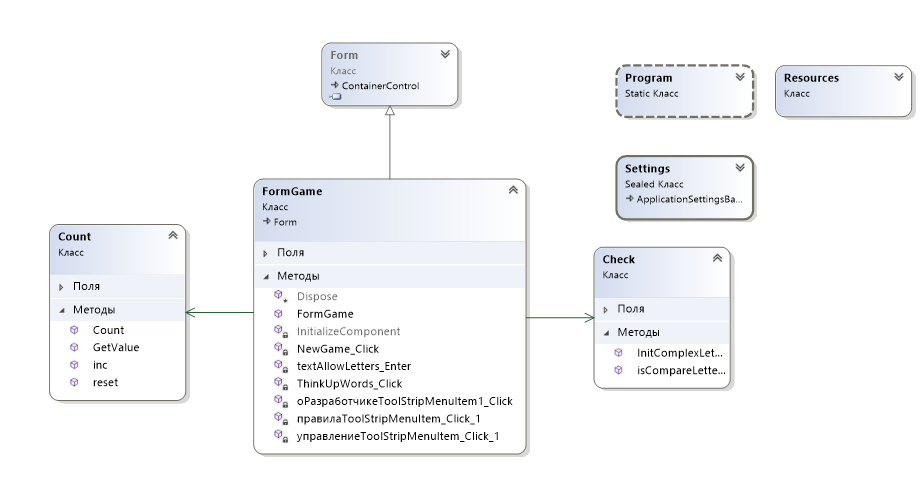


Рисунок 1 – Диаграмма классов

Программа курсовой работы содержит 7 классов: Form, FormGame, Count, Check, Program, Setting, Resourses. Класс Form является родителем класса FormGame (связаны наследованием), класс FormGame связан с Count и Check с помощью ассоциации, используя методы этих классов.

1. Тесты программы
   1. **Unit-тесты**

После программной реализации курсовой работы проводятся тесты для выявления критических ошибок программы и возможных недочетов.

Первым тестом проведен Unit - тест на выполнения метода [[2]](#приложение_б), в результате которого можно определить: выполняется ли метод и время выполнения:

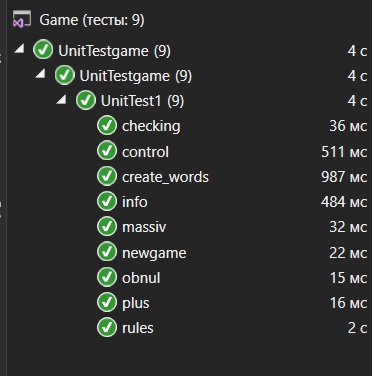


Рисунок 2 – Результат Unit- теста

* 1. **Покрытие кода**

Покрытие кода — мера, используемая при [тестировании программного обеспечения](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F). Она показывает процент [исходного кода](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4) программы, который был выполнен в процессе тестирования.

Вторым тестом проведен тест на покрытие кода, который показал 80.94%.

Таблица 1 – Покрытие кода

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Helen\_LAPTOP-00JDQSM0 | 73 | 19.06% | 310 | 80.94% |
| Game | 68 | 19.21% | 286 | 80.79% |
| Check | 11 | 57.89% | 8 | 42.11% |
| Count | 0 | 0.00% | 7 | 100.00% |
| FormGame | 52 | 16.10% | 271 | 83.90% |
| Program | 5 | 100.00% | 0 | 0.00% |

1. Интерфейс программы
   1. **Общий интерфейс программы**

Программа реализована в одном окне. Оно состоит из поля ввода слова-образца (TextBox), поля ввода найденных слов (RichTextBox), кнопки «Новая игра», которая очищает все поля, кнопки «Проверить слова» Так же на окне реализован вывод текущих очков игрока и меню «Об игре», содержащее информацию о игре и разработчике.

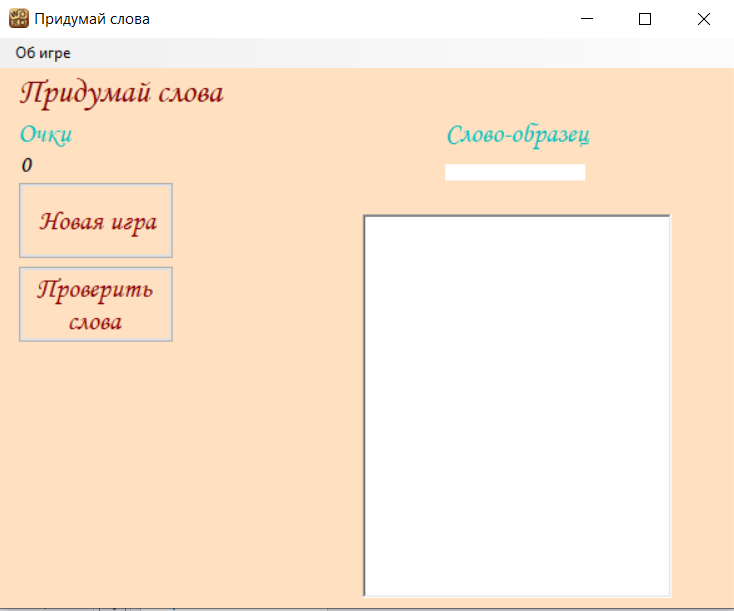


Рисунок 3 – Интерфейс игры «Придумай слова»

* 1. **Интерфейс игры «Придумай слова» на примере**

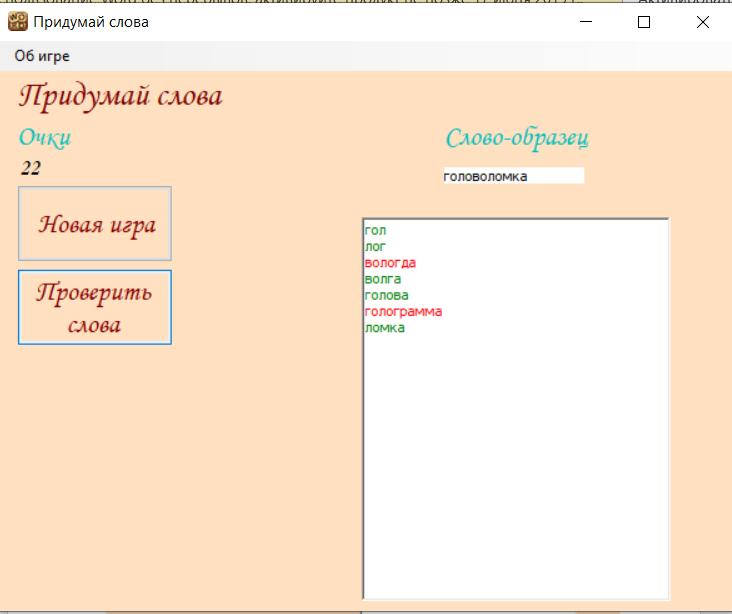


Рисунок 4 – Работа игры «Придумай слова» на примере слова-образца «головоломка»

По нажатию на кнопку «Проверить слова» программа окрашивает подходящие слова зелёным, а не подходящие красным. В случае, если слова не удовлетворяют правилам игры (количество букв меньше трёх и больше, чем в слове-образце), то слова окрашиваются в синий. Также приложение имеет свою иконку.



Рисунок 5 ­– Иконка приложения

Заключение

Результатом выполнения данной работы является реализованная игра «Придумай слова», работающая под управлением Windows, а также закрепление знаний и навыков по дисциплине Объектно-ориентированное программирование и анализ.

В процессе выполнения работы были выполнены следующие задачи:

* выбран язык программирования и среда разработки;
* реализован поиск из списка слов, которые могут быть получены из слова – образца;
* реализован «дружественный» графический интерфейс;
* произведено тестирование созданного приложения.

Список источников

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. (дата обращения 16.05.2019)
2. Диаграммы классов – Блог программиста [электронный ресурс]: <https://pro-prof.com/archives/3212> (дата обращения 16.05.2019)
3. Visual Studio (среда разработки) [электронный ресурс] – Режим доступа: <https://visualstudio.microsoft.com/ru/downloads/?rr=https%3A%2F%2Fwww.google.com%2F> (дата обращения 12.02.2019).
4. С# для начинающих – Пахомов Б.И. [электронный курс] – Режим доступа: <http://lesmatveev.narod.ru/knigi_proect/ci_sharp.pdf> (период обращения с 21.03.2019 по 01.05.2019).
5. Руководство по .NET [электронный ресурс] – Режим доступа: <https://docs.microsoft.com/ru-ru/dotnet/standard/> (период обращения с 21.03.2019 по 10.05.2019).

Приложение А

Код программы

FormGame.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Game

{

public partial class FormGame : Form

{

Check check;

Count count;

public FormGame()

{

InitializeComponent();

check = new Check();

count = new Count();

}

//Меню "Об игре"

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

{

MessageBox.Show("Необходимо составлять слова из показанного на экране слова. " +

"Слово должно быть нарицательным именем существительным в единственном числе. " +

"Минимальная длина слова - 3 буквы.", "Правила");

}

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

{

MessageBox.Show("Чтобы начать игру ведите слово-образец в соответствующее окно. Введите найденные вами слова и нажмите на кнопку 'Проверить слова'. Верные слова станут зелеными, а ошибочные - красными.", "Управление");

}

private void оРазработчикеToolStripMenuItem1\_Click(object sender, EventArgs e)

{

MessageBox.Show("Петрова Елена Александровна, 427-1, курсовая работа", "О разработчике");

}

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

{

string PatternWord = textAllowLetters.Text;

if (PatternWord != "")

{

string[] lines = richTextDone.Text.Split(new char[] { '\_', '\r', '\n', '|', ' ' }, StringSplitOptions.RemoveEmptyEntries);//считывание с Richtextbox

if (lines.Length == 0)//проверка на пустоту Richtextbox

{

MessageBox.Show("Введите слова для проверки.", "Предупреждение");

}

string[] arrayWords = new string[lines.Length];

check.usedLetters = check.InitComplexLetters(PatternWord);

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

{

arrayWords[i] = lines[i];

if ((arrayWords[i].Length > 2) && (arrayWords[i].Length <= PatternWord.Length))

{

if (check.isCompareLetters(arrayWords[i]))

{//в случае true окрасить слово в зелёный

richTextDone.Select(richTextDone.Text.IndexOf(arrayWords[i]), arrayWords[i].Length);

richTextDone.SelectionColor = Color.Green;

for (int k = 0; k < arrayWords[i].Length; k++)

{//увеличить очки

count.inc();

}

}

else

{// в случае false окрасить слово в красный

richTextDone.Select(richTextDone.Text.IndexOf(arrayWords[i]), arrayWords[i].Length);

richTextDone.SelectionColor = Color.Red;

}

}

else

{// если не соответствует правилам, то окрасить в синий

richTextDone.Select(richTextDone.Text.IndexOf(arrayWords[i]), arrayWords[i].Length);

richTextDone.SelectionColor = Color.Blue;

}

}

//вывести очки на экран

label4.Text = count.GetValue();

}

else

{//если не введено слово - образец

MessageBox.Show("Введите слово-образец.","Предупреждение");

textAllowLetters.BackColor = Color.LightCoral;

}

}

private void textAllowLetters\_Enter(object sender, EventArgs e)

{

if (textAllowLetters.Text == "")

{

textAllowLetters.BackColor = Color.White;

textAllowLetters.Text = "";

}

}

//Новая игра

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

{

Count count;

count = new Count();

textAllowLetters.Text = "";

richTextDone.Text = "";

count.reset();

label4.Text = count.GetValue();

}

}

}

Count.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Game

{

class Count

{

private int count;

public Count()

{

count = 0;

}

//геттер счёётчика

public string GetValue()

{

return count.ToString();

}

//увеличение счётчика

public void inc()

{

count++;

}

//обнуление счётчика

public void reset()

{

count = 0;

}

}

}

Check.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Game

{

class Check

{

string abc = "абвгдеёжзийклмнопрстуфхцчшщъыьэюя";

public int[] usedLetters;

//определение массива

public int[] InitComplexLetters(string letters)

{

int[] used = new int[abc.Length];

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

{

used[abc.IndexOf(letters[j])]++;

}

return used;

}

//проверка соответствию

public bool isCompareLetters(string CurrentWord)

{

int[] letters = InitComplexLetters(CurrentWord);

for (int k = 0; k < letters.Length; k++)

{

if (letters[k] > usedLetters[k])

{

return false;

}

}

return true;

}

}

}

Приложение Б

Unit - тесты

using System;

using Microsoft.VisualStudio.TestTools.UnitTesting;

using Game;

namespace UnitTestgame

{

[TestClass]

public class UnitTest1

{

FormGame forgame = new FormGame();

Count schet = new Count();

Check prov = new Check();

[TestMethod]

public void rules()

{

forgame.правилаToolStripMenuItem\_Click\_1(forgame.sender, forgame.e);

}

[TestMethod]

public void control()

{

forgame.управлениеToolStripMenuItem\_Click\_1(forgame.sender, forgame.e);

}

[TestMethod]

public void info()

{

forgame.оРазработчикеToolStripMenuItem1\_Click(forgame.sender, forgame.e);

}

[TestMethod]

public void create\_words()

{

forgame.ThinkUpWords\_Click(forgame.sender, forgame.e);

}

[TestMethod]

public void checking()

{

forgame.textAllowLetters\_Enter(forgame.sender, forgame.e);

}

[TestMethod]

public void newgame()

{

forgame.NewGame\_Click(forgame.sender, forgame.e);

}

[TestMethod]

public void plus()

{

schet.inc();

}

[TestMethod]

public void obnul()

{

schet.reset();

}

[TestMethod]

public void massiv()

{

prov.InitComplexLetters(forgame.textAllowLetters.Text);

}

}

}