**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
федеральное государственное автономное образовательное   
учреждение высшего образования   
«Санкт-Петербургский политехнический университет Петра Великого»  
ИНСТИТУТ СРЕДНЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

|  |  |
| --- | --- |
|  | **Допустить к защите** Заместитель директора  по УМР \_\_\_\_\_\_\_\_\_\_\_\_ Е.Г. Конакина\_  (Подпись) (ФИО)  «\_\_\_»\_\_\_\_\_\_\_\_\_\_ 2023 г. |
|  |

**курсовой проект**Тема: «Программа для игры «угадай букву»»

специальность 09.02.07 группа 32919/11

Студент (ка) Буров С. В.   
 (подпись) (ФИО)  
Преподаватель Молькова Л.Ю.

(подпись) (ФИО)

Санкт-Петербург  
2023

Содержание

[ВВЕДЕНИЕ 3](#_Toc154112130)

[1. ТЕОРЕТИЧЕСКИЕ ОСНОВЫ РАЗРАБОТКИ 4](#_Toc154112131)

[1.1. Описание предметной области 4](#_Toc154112132)

[1.2. Анализ методов решения 6](#_Toc154112133)

[1.3. Обзор средств программирования 7](#_Toc154112134)

[1.4. Описание языка C# 9](#_Toc154112135)

[2. ПРАКТИЧЕСКАЯ ЧАСТЬ 11](#_Toc154112136)

[2.1. Постановка задачи 11](#_Toc154112137)

[2.2. Проектирование приложения 11](#_Toc154112140)

[2.3. Текст программы 15](#_Toc154112145)

[2.4. Описание программы 16](#_Toc154112146)

[2.5. Руководство оператора 17](#_Toc154112152)

[2.6. Программа и методика испытаний 18](#_Toc154112155)

[2.7. Протокол испытаний 22](#_Toc154112162)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 25](#_Toc154112163)

[ПРИЛОЖЕНИЕ А 2](#_Toc154112164)

[ПРИЛОЖЕНИЕ Б 8](#_Toc154112168)

[ПРИЛОЖЕНИЕ В 10](#_Toc154112169)

# ВВЕДЕНИЕ

Современные компьютеры широко применяются в различных областях, включая науку, промышленность, медицину и образование. В системе образования они оказывают значительную помощь преподавателям, облегчая проверку знаний учащихся и подготовку учебных материалов. Кроме того, использование компьютеров способствует расширению интереса к технологиям через разнообразные игровые программы, которые не только развлекают, но и развивают когнитивные навыки.

Наблюдая за постоянным развитием компьютерных технологий, становится ясно, что каждый новый шаг в этой области открывает перед нами еще больше возможностей. Будущее, наполненное инновациями в мире программного обеспечения, дает нам шанс взглянуть на мир с новой перспективы и создать нечто удивительное.

Созданная в рамках данного курсового проекта программа предназначена для развлечения и возможности релаксации.

# 1. ТЕОРЕТИЧЕСКИЕ ОСНОВЫ РАЗРАБОТКИ

## 1.1. Описание предметной области

Компьютерная игра́ (видеоигра) — представляет собой мультимедийное программное обеспечение, созданное для виртуального взаимодействия пользователя с виртуальной средой через графические, звуковые и другие компьютерные элементы.

Она является формой развлечения, которая обычно содержит определенные характеристики и атрибуты.

Компьютерные игры классифицируются по нескольким признакам:

1. **Жанр:** Компьютерные игры могут быть разделены на различные жанры, такие как экшн, стратегии, ролевые игры (RPG), приключения, головоломки, спортивные и т. д.

Каждый жанр имеет свои уникальные правила и цели, определяющие игровой процесс.

2. **Количество игроков и способ взаимодействия:** Игры могут быть одиночными (где игрок играет сам) или многопользовательскими (где несколько игроков могут играть одновременно). Многопользовательские игры могут быть кооперативными (совместное выполнение целей) или соревновательными (борьба друг с другом).

3. **Визуальное представление:** Графическое исполнение игры может быть разнообразным: от двумерных пиксельных игр до трехмерных реалистичных симуляций с высоким разрешением.

4. **Платформа:** Игры могут быть созданы для различных платформ, таких как персональные компьютеры, игровые консоли, мобильные устройства или веб-браузеры. Каждая платформа имеет свои уникальные характеристики и возможности.

Компьютерные игры являются уникальной формой развлечения, объединяющей технологию, дизайн и интерактивность для создания виртуального мира, где пользователи могут погрузиться и наслаждаться различными видами игрового опыта.

Жанр определяется целью игры. Выделяют следующие жанры:

* *Приключенческая игра* (Adventure) — Этот жанр подчеркивает историю и разгадывание загадок. Игроки исследуют миры, взаимодействуют с персонажами и решают головоломки для продвижения по сюжету. Примеры: "Broken Sword", "Life is Strange".
* *Боевик* (Action) — Этот жанр часто характеризуется высокой скоростью игры, быстрыми реакциями и интенсивными боями. Здесь важна активность игрока, его способности к стрельбе, сражениям или выполнению задач в реальном времени. Примеры: "Call of Duty", "Devil May Cry".
* *Ролевая игра* (RPG — англ. *Role Playing Game*) — Этот жанр предлагает игрокам вживаться в роль фантастических персонажей, развивать их навыки, исследовать миры и выполнять квесты. Часто присутствует система прокачки персонажа. Примеры: "The Elder Scrolls", "Final Fantasy".
* *Стратегии* (Strategy) — В играх этого жанра акцент делается на планировании, стратегическом мышлении и управлении ресурсами. Игроки часто принимают решения, влияющие на развитие игрового процесса. Примеры: "Civilization", "StarCraft".
* *Головоломка* (Puzzle) — Здесь акцент делается на разрешении логических задач, головоломок и испытаний интеллекта. Часто требуется изобретательность и умение мыслить логически. Примеры: "Tetris", "Portal".
* *Спортивные* (Sports) *—* Этот жанр имитирует спортивные виды деятельности, предлагая игрокам участвовать в соревнованиях различных видов спорта. Примеры: "FIFA", "NBA 2K".
* *Шутеры* (Shooter) — Шутеры сосредотачиваются на стрельбе и боях с использованием различного вооружения. Могут быть от первого или третьего лица. Примеры: "Counter-Strike", "Destiny".

Каждый из этих жанров имеет свои особенности и привлекает разные аудитории, предлагая уникальный игровой опыт.

## 1.2. Анализ методов решения

В данном случае к моей игре можно присвоить жанр головоломок или логических игр, с элементами словесных игр. Это обычно относится к поджанру головоломок, связанных со словами или буквами. Игрок взаимодействует с машиной, отгадывая слова на основе букв, которые машина «задумывает». Игра предоставляет определенные ограничения для ввода слов, связанных со словарем, и предлагает решение на основе логических предложений о наличии определенной буквы в слове.

Это также может быть интерпретировано как комбинация головоломок и обучающих игр, поскольку игрок расширяет свой словарный запас и логически мыслит, отгадывая слова на основе информации, предоставленной машиной.

Программа для игры «Угадай букву» может быть реализована следующими способами:

1. Консольное приложение: Создание программы, работающей в командной строке, где игрок взаимодействует с игрой через текстовый интерфейс. Это позволит игроку вводить слова, а машина давать ответы в текстовом формате.
2. Графический интерфейс пользователя (GUI): Разработка приложения с использованием графического интерфейса, где игрок взаимодействует с игрой через окна, кнопки и текстовые поля. Это делает процесс более интуитивно понятным и визуально привлекательным.
3. Веб-приложение: Создание игры, доступной через веб-браузер. Это позволяет играть в игру в онлайн-режиме, обеспечивает доступ к игре с разных устройств и обновление словаря через интернет.
4. Мобильное приложение: Разработка игры для мобильных устройств с помощью специальных SDK и инструментов разработки. Это предоставляет возможность играть в любом месте с помощью мобильного устройства.

В нашем случае мы рассмотрим консольный способ.

## 1.3. Обзор средств программирования

Рассмотрим несколько языков программирования и определимся с наиболее подходящим языком для нашей программы:

1. **Python:** это высокоуровневый, интерпретируемый язык программирования с динамической типизацией. Он известен своим чистым и читаемым синтаксисом, что делает его привлекательным для начинающих и опытных разработчиков. Python имеет богатую стандартную библиотеку, а также множество сторонних библиотек для различных целей: от веб-разработки и научных вычислений до машинного обучения и искусственного интеллекта.
2. **JavaScript:** это язык программирования, который широко используется для создания интерактивных веб-страниц и веб-приложений. Он позволяет добавлять динамическое поведение к веб-страницам, работать с анимацией, обрабатывать события пользователя и многое другое. JavaScript используется во фронтенд-разработке в сочетании с HTML и CSS.
3. **Java:** это объектно-ориентированный язык программирования, который известен своей платформонезависимостью. Программы на Java могут работать на различных платформах, таких как Windows, macOS, Linux и других, благодаря виртуальной машине Java (JVM). Он широко используется для создания приложений для мобильных устройств, веб-приложений, корпоративных систем и игр.
4. **C++:** это высокопроизводительный язык программирования, который предоставляет возможность близкого управления памятью и низкоуровневыми ресурсами компьютера. Он широко используется в разработке игр, системного программирования, операционных систем и других приложений, где важна производительность.
5. **Swift:** это язык программирования, разработанный для создания приложений для устройств Apple, таких как iPhone, iPad, Mac и других устройств от компании Apple. Он предлагает безопасность, производительность и чистый синтаксис, обеспечивая множество современных функций для разработки под платформу Apple.
6. **PHP:** это язык программирования, который преимущественно используется для создания динамических веб-страниц и веб-приложений. Он широко применяется в разработке сайтов благодаря своей способности взаимодействовать с HTML и управлять содержимым веб-страниц.
7. **C#:** это язык программирования, разработанный Microsoft, основанный на языке C++. C# используется для создания приложений под платформу .NET, включая Windows-приложения, веб-сервисы и игры на популярном игровом движке Unity. Является объектно-ориентированным и обладает широкими возможностями разработки
8. **Rudy:** это динамический, интерпретируемый язык программирования, известный своим фреймворком Ruby on Rails для веб-разработки. Он поддерживает простой и чистый синтаксис, что делает его популярным среди разработчиков.

В нашем примере было упомянуто множество различных языков программирования.

Я даже рассматривал Python, благодаря его простому и читаемому синтаксису, обширной библиотеке для обработки данных, а также для создания игр, развлекательных проектов и его лаконичности.

Однако, выбор в итоге пал на C#, поскольку он обеспечивает:

- Тесную интеграцию с платформой .NET, что позволяет создавать разнообразные приложения для Windows и веб-сервисы.

- Мощные инструменты разработки в среде Visual Studio, ускоряющие процесс создания приложений.

Эти преимущества и возможности C# послужили основной причиной окончательного выбора этого языка для реализации проекта.

Таким образом, средством программирования данного курсового проекта был выбран язык С#, среда разработки Visual Studio 2017.

## 1.4. Описание языка C#

C# – это многоцелевой язык программирования. От настольных приложений до мобильных приложений, от веб-разработки до разработки игр - C# подходит для всего.

C# был разработан Microsoft и является основным языком для платформы .NET. Это означает, что мы получаем доступ к таким крутым инструментам Microsoft, как Visual Studio - очень мощная среда для разработки, её аналог Visual Studio Code, которая является своеобразным легковесным конструктором и ей пользуются многие разработчики JavaScript, PHP, Java, Go, Python и других языков.

Асинхронное программирование может быть головной болью, но в C# есть ключевые слова async и await, которые позволяют вам переложить всю сложную работу на комплятор. Это помогает писать более эффективные и отзывчивые приложения, не путаясь в сложных концепциях и хитросплетениях потоков.

Ссылка на электронный источник, откуда была взята информация: https://habr.com/ru/companies/otus/articles/723454/

**Ключевые слова C#:**

|  |  |  |  |
| --- | --- | --- | --- |
| abstract | extern | null | struct |
| as | false | object | switch |
| base | finally | operator | this |
| bool | fixed | out | throw |
| break | float | override | true |
| byte | for | params | try |
| case | foreach | private | typeof |
| catch | goto | protected | uint |
| const | if | public | ulong |
| continue | implicit | readonly | unchecked |
| decimal | in | ref | unsafe |
| default | int | return | ushort |
| delegate | interface | sbyte | using |
| do | internal | sealed | virtual |
| double | is | short | void |
| else | lock | sizeof | while |
| enum | long | stackalloc |  |
| event | namespace | static |  |
| explicit | new | string |  |

**Контекстные ключевые слова C#:**

|  |  |  |  |
| --- | --- | --- | --- |
| add | equals | join | set |
| ascending | from | let | value |
| async | get | on | var |
| await | global | orderby | where |
| by | group | partial | yield |
| descending | in | remove |  |
| dynamic | into | select |  |

**Основные алгоритмические конструкции C#:**

|  |  |
| --- | --- |
| Оператор присваивания | int a; // Объявление переменной "a"  a = 5; // Присваивание переменной "a" значения 5  int b = 10; // Объявление переменной "b" и присваивание ей значения 10 в одной строке  string message = "Привет, мир!"; // Присваивание строковой переменной "message" текстового значения |
| Условный оператор | *if (условие) {(действие) } else {(альтернатива)} ;*  int x = 10; // Объявление переменной x и присвоение ей значения 10  if (x > 5) // Если x больше 5  {  Console.WriteLine("x больше 5"); // Вывести сообщение "x больше 5" в консоль  }  else if (x == 5) // Иначе, если x равен 5  {  Console.WriteLine("x равен 5"); // Вывести сообщение "x равен 5" в консоль  }  else // Иначе (если x меньше 5)  {  Console.WriteLine("x меньше 5"); // Вывести сообщение "x меньше 5" в консоль  } |
| Оператор множественного ветвления | *Конструкция switch/case аналогична конструкции if/else, так как позволяет обработать сразу несколько условий:*  int day = 3; // Переменная, которая будет использоваться в выражении switch  switch (day)  {  case 1:  Console.WriteLine("Понедельник");  break;  case 2:  Console.WriteLine("Вторник");  break;  case 3:  Console.WriteLine("Среда");  break;  case 4:  Console.WriteLine("Четверг");  break;  case 5:  Console.WriteLine("Пятница");  break;  default:  Console.WriteLine("Выходные");  break;  }*После ключевого слова switch в скобках идет сравниваемое выражение. Значение этого выражения последовательно сравнивается со значениями, помещенными после оператора сase. И если совпадение будет найдено, то будет выполняться определенный блок сase.*  *В конце блока сase ставится оператор break, чтобы избежать выполнения других блоков.*  *Если мы хотим также обработать ситуацию, когда совпадения не будет найдено, то можно добавить блок default,.* |
| Арифметический цикл  **(**применяется, когда известно количество повторений цикла) | for *([инициализация счетчика]; [условие]; [изменение счетчика])*  // Цикл от 1 до 10 с шагом 2  for (int i = 1; i <= 10; i += 2)  {  Console.WriteLine(i); // Вывести значение i в консоль  } |
| Цикл с предусловием  **(**применяется, когда неизвестно количество повторений цикла) | *while (условие) { тело цикла };*  Этот цикл будет выполняться до тех пор, пока истинно *условие* (логическое выражение, возвращающее значение типа **Boolean**). При этом если это выражение сразу равно **false**, *тело цикла* не будет выполнено ни разу. Нужно очень внимательно следить за написанием *условия* и контролем завершения цикла, так как в результате ошибки цикл **while** будет повторяться бесконечное количество раз, что приведёт к "зацикливанию" и "зависанию" программы. |
| Цикл с постусловием  **(**применяется, когда неизвестно количество повторений цикла) | *do { тело цикла } while (условие);*  Повторения сначала выполняет *тело цикла*, а затем уже проверяет выполнение *условия*: Таким образом, этот вариант цикла гарантирует, что *тело цикла* будет выполнен по крайней мере один раз. И будет выполняться до тех пор, пока *условие* не станет истинным (т.е. **true**). |

# 2. ПРАКТИЧЕСКАЯ ЧАСТЬ

## 2.1. Постановка задачи

### 2.1.1. Основания для разработки

Разработка ведётся на основании задания к курсовому проекту по профессиональному модулю ПМ.01 «Разработка программных модулей программного обеспечения для компьютерных систем» МДК 01.01 «Разработка программных модулей» и утверждена Институтом среднего профессионального образования.

### 2.1.2. Назначение программы

Человек играет против машины; машина «задумывает» букву; человек вводит с клавиатуры разные слова, и машина сообщает, есть ли задуманная буква в этом слове.

Полный текст технического задания приведен в приложении А.

## 2.2. Проектирование приложения

На этапе проектирования были разработаны диаграмма прецедентов, диаграмма последовательностей, диаграмма активности и блок схема.

### 2.2.1. Диаграмма прецедентов

Диаграммы прецедентов частично описывает use case – прецедент использования проектируемой системы, давая частичное описание частичного применения системы с точки зрения условного внешнего обозревателя. (рисунок 1).

Изображение выглядит как текст, снимок экрана, диаграмма, дизайн

Автоматически созданное описание

Рисунок 1 Диаграмма прецедентов

### 2.2.2. Диаграмма последовательностей

Диаграмма последовательности отражает поток событий, происходящих в рамках варианта использования.

Все действующие лица показаны в верхней части диаграммы. Стрелки соответствуют сообщениям, передаваемым между действующим лицом и объектом или между объектами для выполнения требуемых функций. На диаграмме последовательности объект изображается в виде прямоугольника, от которого вниз проведена пунктирная вертикальная линия. Эта линия называется линией жизни (lifeline) объекта. Она представляет собой фрагмент жизненного цикла объекта в процессе взаимодействия.

Изображение выглядит как диаграмма, текст, снимок экрана, План

Автоматически созданное описание

Рисунок 2 Диаграмма последовательностей

### 2.2.3. Диаграмма активности

На рисунке 3 показана диаграмма активности (деятельности) во время сеанса игры.

Диаграмма активности – это UML-диаграмма, на которой показаны действия, состояния. Под деятельностью понимается спецификация исполняемого поведения в виде координированного последовательного и параллельного выполнения подчинённых элементов – вложенных видов деятельности и отдельных действий action, соединённых между собой потоками, которые идут от выходов одного узла ко входам другого.

Изображение выглядит как текст, снимок экрана, диаграмма, астрономия

Автоматически созданное описание

Рисунок 3 Диаграмма активности сеанса игры

### 2.2.4. Блок схема

Блок-схема представляет собой визуальное изображение логики работы программы, соединенных линиями, показывающими поток данных или действий. Она служит важным инструментом для понимания структуры и последовательности операций программы.

Изображение выглядит как зарисовка, диаграмма, рисунок, линия

Автоматически созданное описание

## 2.3. Текст программы

Текст программы в соответствии с ГОСТ 19.101-77 (СТ СЭВ 1626-79) и ГОСТ 19.401-79 (СТ СЭВ 3746-82) представляет собой запись программы на исходном языке программирования с необходимыми комментариями. Текст программы представляет собой документ, выполненный машинным способом, и приведен в приложении В.

## 2.4. Описание программы

### 2.4.1. Общие сведения

Игровая Человек играет против машины; машина «задумывает» букву; человек вводит с клавиатуры разные слова, и машина сообщает, есть ли задуманная буква в этом слове.

### 2.4.2. Функциональное назначение

Основное назначение программного продукта заключается в организации игры «Угадай букву»: человек играет против машины; машина «задумывает» букву; человек вводит с клавиатуры разные слова, и машина сообщает, есть ли задуманная буква в этом слове.

### 2.4.3. Описание логической структуры системы

Программа использует функции следующих библиотек среды С#:

System, System.Collections.Generic,System.ComponentModel, System.Linq, System.IO.

Исполняемый файл программы создан средствами среды C#, имеет имя   
guess\_letter.sln.

### 2.4.4. Используемые технические и программные средства

Для нормального функционирования данной информационной системы необходим компьютер(ноутбук), клавиатура, мышь и следующие технические средства:

- процессор Apple M1 (или другой совместимый);

- объем свободной оперативной памяти ~500 Кб;

- объем необходимой памяти на жестком диске ~3Мб;

- стандартный VGA-монитор или совместимый;

- стандартная клавиатура;

- манипулятор «мышь».

### 2.4.5. Вызов и загрузка

Программа может быть загружена как с диска, так и с жесткого диска. В последнем случае требуется предварительно переписать программу с диска на жесткий диск.

Исполняемым файлом программы является файл guess\_letter.sln. Для его запуска необходимо дважды щелкнуть по исполняемому файлу левой кнопкой мышки.

## 2.5. Руководство оператора

### 2.5.1. Назначение программы

Основное назначение программного продукта заключается в организации игры «угадай букву»: Человек играет против машины; машина «задумывает» букву; человек вводит с клавиатуры разные слова, и машина сообщает, есть ли задуманная буква в этом слове.

### 2.5.2. Выполнение программы и сообщения оператору

Для запуска программы дважды щелкните левой кнопкой мыши по исполняемому файлу guess\_letter.sln.

Процесс игры, этапы работы приложения показаны в виде иллюстраций в приложении Б.

При запуске работы приложения загружается терминал, а в нем инструкции для пользователя, в которых сообщается: Добро пожаловать в игру 'Угадай букву'! Введите 'add' для добавления слова, 'view' для просмотра словаря, 'delete' для удаления слова, 'exit' для выхода или слово для угадывания буквы. При вводе той или иной команды выполняются соответствующие действия. При вводе команды Add, программа выдает текст: «Введите слово для добавления:» - вы вводите слово, которое хотите добавить в словарь, при наличии того, что такого же слова еще не было в словаре, в противном случае программа выдает текст: «Такое слово уже есть в словаре». При вводе команды View, вам открывается весь список слов, содержащийся в словаре. При вводе команды delete, программа выдает текст: «Введите слово для удаления:» - вы вводите слово из словаря, которое хотите удалить и которое есть в словаре, при отсутствии слова в словаре, программа выдает текст: «Такого слова нет в словаре». При вводе команды exit, программа завершает свою работу. Так же после удачного написания слова, программа сообщает что в этом слове содержится загаданная буква, цикл обновляется и программа загадывает новую букву.

## 2.6. Программа и методика испытаний

### 2.6.1. Объект испытаний

Объектом испытаний является игровая программа guess\_letter.sln. Человек играет против машины; машина «задумывает» букву; человек вводит с клавиатуры разные слова, и машина сообщает, есть ли задуманная буква в этом слове.

### 2.6.2. Цель испытаний

Целью испытаний является проверка соответствия программы требованиям Технического Задания.

### 2.6.3. Требования к программе

В процессе испытаний подлежат проверке следующие требования к программе:

#### 2.6.3.1. Требования к функциональным характеристикам

**1 версия:**

* каждое слово должно быть на отдельной строке;
* словарь хранится в текстовом файле;
* файл содержит примерно 20 000 слов;
* должна быть предусмотрена возможность добавления слов в словарь;

**2 версия:**

* добавление новых слов;
* словарь хранится в файле
* каждое слово на отдельной строке;
* файл содержит примерно 20 000 слов;
* должна быть предусмотрена работа со словарем;
* просмотр словаря в алфавитном порядке, начиная с заданной буквы или слога(слова)
* удаление слов

#### 2.6.3.2. Требования к информационной и программной совместимости

Для полноценного функционирования данной системы необходимо наличие операционной системы выше Mac OC X Snow Leopard (10.6) или совместимой. Язык интерфейса – русский.

#### 2.6.3.3. Требования к маркировке и упаковке

Программа должна поставляться в виде проекта, исполняемого (sln) файла, установщика и документации.

#### 2.6.3.4. Требования к транспортировке и хранению

Программа распространяется в электронном виде. Требования к транспортировке и хранению не предъявляются.

### 2.6.4. Требования к программной документации

На испытания должны быть представлены следующие программные документы:

* техническое задание
* текст программы
* описание программы
* руководство оператора
* описание языка

### 2.6.5. Средства и порядок испытаний

Для проведения испытаний необходимы следующие технические средства:

* процессор Apple M1 или другой совместимый;
* объем свободной оперативной памяти ~500 Кб;
* объем необходимой памяти на жестком диске ~3Мб;
* стандартный VGA-монитор или совместимый;
* стандартная клавиатура;
* манипулятор «мышь».

Для проведения испытаний необходимы следующие программные средства:

Операционная система Mac OS X Snow Leopard (10.6) или выше.

Испытания проводятся в следующем порядке:

1) проверяется наличие и комплектность программной документации (п.2.6.4)

2) проверяется соответствие требованиям к маркировке и упаковке (п.2.6.3.3)

3) проверяется соответствие требованиям к функциональным характеристикам (п.2.6.3.1)

4) проверяется соответствие требованиям к информационной и программной совместимости (п.2.6.3.2)

### 2.6.6. Методы испытаний

#### 2.6.6.1. Для проверки способности программы обеспечивать добавление нового слова в словарь, необходимо:

* запустить программу
* выбрать команду add для добавления нового слова в словарь
* ввести новое слово
* проверить наличие добавленного слова в словаре программы.

#### 2.6.6.2. Для проверки способности программы обеспечивать удаление слова из словаря, необходимо:

* запустить программу
* выбрать команду delete для удаления слова из словаря
* ввести слово для удаления
* проверить отсутствие удаленного слова из словаря программы.

#### 2.6.6.3. Для проверки способности программы обеспечивать благоприятный поиск буквы в слове, необходимо:

* запустить программу
* начать игру
* ввести слово
* проверить результаты поиска загаданной буквы в веденном слове

#### 2.6.6.4. Для проверки способности программы обеспечивать обработку некорректного ввода, необходимо:

* запустить программу
* просмотреть какие слова содержаться в словаре
* выбрать команду add для добавления нового слова в словарь
* написать слово, которое уже есть в словаре
* проверить наличие сообщение о том, что данное слово уже есть в словаре

## 2.7. Протокол испытаний

Результаты испытаний программы представлены в таблице 1, рисунки приведены в приложении Б.

**Результаты испытаний программы**

*Таблица 1*

| **№** | **Проверяемые требования** | **Сообщения программы и вводимые значения** | **Ожидаемые результаты** | **Результат** |
| --- | --- | --- | --- | --- |
| 1 | Способность программы обеспечивать добавление нового слова в словарь. | * запустить программу * выбрать команду add для добавления нового слова в словарь * ввести новое слово * проверить наличие добавленного слова в словаре программы | Новое слово было успешно добавлено в словарь программы | Пройден  Рис. Б1 |
| 2 | Способность программы обеспечивать удаление слова из словаря, необходимо | * запустить программу * выбрать команду delete для удаления слова из словаря * ввести слово для удаления * проверить отсутствие удаленного слова из словаря программы | Слово было успешно удалено из словаря программы | Пройден  Рис. Б2 |
| 3 | Способность программы обеспечивать благоприятный поиск буквы в слове | * запустить программу * начать игру * ввести слово * проверить результаты поиска загаданной буквы в веденном слове | Программа успешно нашла букву в веденном слове | Пройден  Рис. Б3 |
| 4 | Способность программы обеспечивать обработку некорректного ввода | * запустить программу * просмотреть какие слова содержаться в словаре * выбрать команду add для добавления нового слова в словарь * написать слово, которое уже есть в словаре * проверить наличие сообщения о том, что данное слово уже есть в словаре | Программа успешно выдала некорректность в вводе слова и выдала соответствующий текст | Пройден  Рис. Б4 - Б5 |

ЗАКЛЮЧЕНИЕ

Программа, разработанная в рамках курсового проекта, успешно соответствует всем требованиям технического задания, что было подтверждено результатами проведенных испытаний.

Эта программа предоставляет возможность использования в развлекательных целях, позволяя пользователям наслаждаться отдыхом и релаксацией.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. А. Хейлсберг, М. Торгерсен, С. Вилтамут, П. Голд ., Язык программирования C#, 4-е издание, СПб:-ПИТЕР, 2020
2. <https://habr.com/ru/companies/otus/articles/723454/https://app.diagrams.net>
3. <https://pylot.me/article/20-pochemu-python-luchshij-pervij-yazik-programmirovaniya/>
4. <https://habr.com/ru/companies/otus/articles/723460/#:~:text=Python,-1.&text=Python%20известен%20своей%20читаемостью%20и,Python%2C%20который%20является%20интерпретируемым%20языком>.
5. https://gb.ru/blog/zhanry-kompjuternyh-igr/https://ru.wikipedia.org/wiki/Программирование\_игр
6. Стандарты Единой Системы Программной Документации:

ГОСТ 19.105-78 Общие требования к программным документам

ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом

ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению

ГОСТ 19.301-78 Программа и методика испытаний. Требования к содержанию и оформлению

ГОСТ 19.401-78 Текст программы. Требования к содержанию и оформлению

ГОСТ 19.402-78 Описание программы. Требования к содержанию и оформлению

ГОСТ 19.505-79 Руководство оператора. Требования к содержанию и оформлению

ГОСТ 19.701-90 Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения

Министерство ОБРАЗОВАНИЯ И науки Российской Федерации  
Федеральное автономное образовательное учреждение высшего профессионального образования  
«Санкт-Петербургский государственный политехнический университет Петра Великого»  
(ФГАОУ ВО «СПбПУ)  
**Институт среднего профессионального образования**

УТВЕРЖДАЮ  
Председатель ПЦК   
\_\_\_\_\_\_\_\_\_\_\_ Андреев В.А.  
\_\_ .\_\_. 2023

Программа для игры «угадай букву»  
**Техническое задание**  
Листов 11

ПРОВЕРИЛ  
Преподаватель   
\_\_\_\_\_\_\_\_\_\_\_ Молькова Л.Ю.  
23.11.2023

ВЫПОЛНИЛ  
Студент группы 32919/11   
\_\_\_\_\_\_\_\_\_\_\_Буров С. В.   
23.11.2023

2023

# ВВЕДЕНИЕ

* 1. Полное наименование программной разработки: «ПРОГРАММА ДЛЯ ИГРЫ «УГАДАЙ БУКВУ».
  2. Игровая программа «УГАДАЙ БУКВУ»: Человек играет против машины; машина «задумывает» букву; человек вводит с клавиатуры разные слова, и машина сообщает, есть ли задуманная буква в этом слове.
  3. В соответствии с заданием программный продукт состоит из теоретической и практической частей. Теоретическая часть включает описание предметной области задачи, анализ методов её решения, обзор и обоснование выбора средств программирования. Практическая часть включает подробное описание проектирования программного продукта, разработку и реализацию на языке программирования C# с использованием среды программирования Visual Studio программных модулей программного продукта.

# ОСНОВАНИЕ ДЛЯ РАЗРАБОТКИ

* 1. Разработка ведётся на основании задания к курсовому проекту по профессиональному модулю ПМ.01 «Разработка программных модулей программного обеспечения для компьютерных систем» МДК 01.01 «Разработка программных модулей» и утверждена Институтом среднего профессионального образования.

# НАЗНАЧЕНИЕ РАЗРАБОТКИ

* 1. Основное назначение программного продукта: программа для игры в жанре «логическая игра с элементами словесных игр».
  2. Эксплуатационное назначение программного продукта: программа предназначена для широкого круга пользователей, без ограничения по возрасту, не требует внесения денежных средств или использования платёжных систем для игры, предназначена для развлекательных целей.

4. ТРЕБОВАНИЯ К РАЗРАБОТКЕ

4.1. Требования к функциональным характеристикам:

**1 версия:**

* каждое слово должно быть на отдельной строке;
* словарь хранится в текстовом файле;
* файл содержит примерно 20 000 слов;
* должна быть предусмотрена возможность добавления слов в словарь;

**2 версия:**

* добавление новых слов;
* словарь хранится в файле
* каждое слово на отдельной строке;
* файл содержит примерно 20 000 слов;
* должна быть предусмотрена работа со словарем;
* просмотр словаря в алфавитном порядке, начиная с заданной буквы или слога(слова)
* удаление слов

4.2. Требования к надежности:

* использование лицензированного программного обеспечения;
* проверка программы на наличие вирусов;
* организация бесперебойного питания.

4.3. Требования к составу и параметрам технических средств

Для нормального функционирования данной информационной системы необходим компьютер(ноутбук), клавиатура, мышь и следующие технические средства:

- процессор Apple M1 или другой совместимый;

- объем свободной оперативной памяти ~500 Кб;

- объем необходимой памяти на жестком диске ~20Мб;

- стандартный VGA-монитор или совместимый;

- стандартная клавиатура;

- манипулятор «мышь».

4.4. Требования к информационной и программной совместимости

Для полноценного функционирования данной системы необходимо наличие операционной системы выше MAC OC X Snow Leopard (10.6) или совместимой. Язык интерфейса – русский.

4.5. Требования к маркировке и упаковке

Программа должна поставляться в виде проекта, исполняемого (sln) файла, установщика и документации.

4.6. Требования к транспортировке и хранению

Программа распространяется в электронном виде. Требования к транспортировке и хранению не предъявляются.

4.7. Специальные требования

Теоретическая часть включает описание предметной области задачи, анализ методов её решения, обзор и обоснование выбора средств программирования. Практическая часть включает подробное описание проектирования программного продукта, разработку и реализацию на языке программирования C# с использованием среды программирования Visual Studio программных модулей программного продукта.

5. ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ

5.1. Предварительный состав программной документации:

- «Техническое задание»;

- разрабатываемые программные модули должны быть самодокументированы, т.е. тексты программ должны содержать все необходимые комментарии;

- разрабатываемое программное обеспечение должно включать справочную систему.

5.2. Перечень материалов пояснительной записки



Рисунок 1 – Перечень материалов пояснительной записки

6. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

Технико-экономические показатели не рассчитываются.

7. СТАДИИ И ЭТАПЫ РАЗРАБОТКИ

Таблица 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Содержание стадии** | **Содержание этапа** | **Срок 2023 г.** | | **Форма  отчетности** |
| **начало** | **конец** |
| Техническое задание | Составление технического задания | 23.10 | 26.10 | Техническое задание |
| Эскизный проект | Проектирование программы | 30.02 | 3.11 | UML-диаграммы |
| Рабочий проект | Разработка спецификаций | 09.03 | 13.11 | Спецификации отдельных компонентов |
| Составление программы | 13.03 | 17.11 | Программная документация |
| Приёмо-сдаточные испытания | 17.04 | 20.11 | Протокол испытаний (п. 2.7 пояснительной записки) |
| Приёмка | Защита курсового проекта | 21.11 | 22.11 | Оценка за курсовой проект |

8. ПОРЯДОК КОНТРОЛЯ И ПРИЕМКИ

8.1. Порядок контроля

Контроль выполнения должен осуществляться руководителем курсового проекта (преподавателем) в соответствие с п.7.

8.2. Порядок приемки

Приемка должна осуществляться с участием руководителя после проведения приемо-сдаточных испытаний. В результате защиты курсового проекта должна быть выставлена оценка за курсовой проект.

ПРИЛОЖЕНИЕ Б

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

Рисунок Б1 Добавление нового слова

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

Рисунок Б2 Удаление слова

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок Б3 Поиск буквы в слове

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок Б4 Ввод некорректного слова

ПРИЛОЖЕНИЕ В

**Текст программы**

using System;  
using System.Collections.Generic;  
using System.IO;  
using System.Linq;  
  
class Program  
{  
 static void Main()  
 {  
 *// Путь к файлу со словарём* var dictionaryPath = "../../../dictionary.txt";  
 *// Чтение всех слов из файла и сохранение их в HashSet для быстрого поиска* var dictionary = new HashSet<string>(File.ReadAllLines(dictionaryPath));  
 *// Генерация случайной буквы* char secretLetter = GetRandomLetter();  
  
 Console.WriteLine("Добро пожаловать в игру 'Угадай букву'!");  
 Console.WriteLine("Введите 'add' для добавления слова, 'view' для просмотра словаря, 'delete' для удаления слова, 'exit' для выхода или слово для угадывания буквы.");  
  
 while (true)  
 {  
 Console.Write("> ");  
 var input = Console.ReadLine().ToLower();  
  
 if (input == "exit")  
 break;  
  
 switch (input)  
 {  
 case "add":  
 AddWord(dictionary, dictionaryPath);  
 break;  
 case "view":  
 ViewDictionary(dictionary);  
 break;  
 case "delete":  
 DeleteWord(dictionary, dictionaryPath);  
 break;  
 default:  
 *// Проверка, есть ли слово в словаре* if (dictionary.Contains(input))  
 {  
 *// Проверка, содержит ли слово задуманную букву* if (input.Contains(secretLetter))  
 Console.WriteLine($"Слово содержит букву '{secretLetter}'!");  
 else  
 Console.WriteLine("Попробуйте другое слово.");  
 }  
 else  
 {  
 Console.WriteLine("Такого слова нет в словаре.");  
 }  
 break;  
 }  
 }  
 }  
  
 *// Генерация случайной буквы* static char GetRandomLetter()  
 {  
 var random = new Random();  
 return (char)('а' + random.Next(32)); *// Для русского алфавита* }  
  
 *// Функция добавления слова в словарь* static void AddWord(HashSet<string> dictionary, string path)  
 {  
 Console.Write("Введите слово для добавления: ");  
 var word = Console.ReadLine().ToLower();  
  
 if (dictionary.Add(word))  
 {  
 File.AppendAllText(path, word + Environment.NewLine);  
 Console.WriteLine("Слово добавлено.");  
 }  
 else  
 {  
 Console.WriteLine("Такое слово уже есть в словаре.");  
 }  
 }  
  
 *// Функция для просмотра словаря* static void ViewDictionary(HashSet<string> dictionary)  
 {  
 Console.WriteLine("Словарь:");  
 foreach (var word in dictionary.OrderBy(w => w))  
 {  
 Console.WriteLine(word);  
 }  
 }  
  
 *// Функция удаления слова из словаря* static void DeleteWord(HashSet<string> dictionary, string path)  
 {  
 Console.Write("Введите слово для удаления: ");  
 var word = Console.ReadLine().ToLower();  
  
 if (dictionary.Remove(word))  
 {  
 File.WriteAllLines(path, dictionary);  
 Console.WriteLine("Слово удалено.");  
 }  
 else  
 {  
 Console.WriteLine("Такого слова нет в словаре.");  
 }  
 }  
}