Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Основы алгоритмизации и программирования (ОАиП)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

на тему:

«ИГРА ФУТБОЛ»

БГУИР КП 1-40 01 01 03 014 ПЗ

Студент: гр. 551005 Коваленко И.А.

Руководитель: асс. Болтак С.В.

Минск 2016

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

УТВЕРЖДАЮ

Заведующий кафедрой ПОИТ

––––––––––––––––––––––––

(подпись)

Лапицкая Н.В. 2016 г.

ЗАДАНИЕ

по курсовому проектированию

Студенту Коваленко Илье Андреевичу

1. Тема работы Игра Футбол

2. Срок сдачи студентом законченной работы 20.12.2016

3. Исходные данные к работе Среда программирования RAD Studio 10.1 Berlin, модули стандартной библиотеки Delphi

4. Содержание расчётно-пояснительной записки (перечень вопросов, которые подлежат разработке)

*Введение.*

*1. Аналитический обзор литературы и существующих аналогов;*

*2. Разработка алгоритма;*

*3. Разработка программного средства;*

*4. Обоснование технических приемов программирования;*

*5. Тестирование, экспериментальные исследования и анализ полученных результатов;*

*6. Руководство пользователя программы;*

*Заключение, список литературы, ведомость, приложения.*

5. Перечень графического материала (с точным обозначением обязательных чертежей и графиков)

1. Схема программы

6. Консультант по курсовому проекту Болтак С.В.

7. Дата выдачи задания 10.09.2016 г.

8. Календарный график работы над проектом на весь период проектирования (с обозначением сроков выполнения и процентом от общего объёма работы):

раздел 1, введение к 02.03.2016 – 10 % готовности работы;

разделы 2 к 15.03.2016 – 30 % готовности работы;

разделы 3,4 к 15.04.2016 – 60 % готовности работы;

раздел 5, 6 к 05.05.2016 – 90 % готовности работы;

оформление пояснительной записки и графического материала к 20.05.2016 – 100 % готовности работы.

Защита курсового проекта с 23.05 по 12.06 2016 г.

РУКОВОДИТЕЛЬ С.В. Болтак

(подпись)

Задание принял к исполнению –––\_\_\_\_––     15.02.2016 г.

(дата и подпись студента)

**СОДЕРЖАНИЕ**

ВВЕДЕНИЕ 5

1 АНАЛИЗ ИГРЫ ФУТБОЛ И АНАЛОГОВ ПО 6

1.1 Общие сведения о футболе и правила игры 6

1.2 Краткий анализ уже созданного ПО 10

1.3 Постановка требований к разрабатываемому ПО 13

2 РАЗРАБОТКА АЛГОРИТМА 12

3 РАЗРАБОТКА ПРОГРАММНОГО СРЕДСТВА 14

3.1 Выбор структур данных 14  
 3.2 Работа с файлами 15

4 ОБОСНОВАНИЕ ТЕХНИЧЕСКИХ ПРИЕМОВ   
ПРОГРАММИРОВАНИЯ 17

5 ТЕСТИРОВАНИЕ 19

6 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ ПРОГРАММЫ 24

6.1 Создание теста 24

6.2 Редактирование теста 25

6.3 Прохождение теста 25

ЗАКЛЮЧЕНИЕ 28

СПИСОК ЛИТЕРАТУРЫ 29

ПРИЛОЖЕНИЕ А. ТЕКСТ МОДУЛЯ UnitMain 30

ПРИЛОЖЕНИЕ Б. ТЕКСТ МОДУЛЯ UnitModel 32

ПРИЛОЖЕНИЕ В. ТЕКСТ МОДУЛЯ UnitUser 42

ПРИЛОЖЕНИЕ Г. ТЕКСТ МОДУЛЯ UnitAdmin 50

**ВВЕДЕНИЕ**

В игры с мячом играли во многих странах. В Китае такая разновидность называлась Чжу-Кэ. В древней Спарте игра называлась «Эпискирос», а в Древнем Риме «Харпастум». Где-то в Новое время в Брянских землях проводились игры, инвентарём которых был кожаный мяч размером с человеческую голову, набитый перьями. Эти состязания наименовались как «шалыга» и «кила». Примерно в XIV веке итальянцы изобрели игру «Кальчо». Именно они завезли эту игру на Британские острова.

В XIX веке футбол в Англии приобрёл популярность, сравнимую с крикетом. В него играли в основном в колледжах. Но в некоторых колледжах правила разрешали ведение и передачу мяча руками, а в других напротив, запрещалось. Первая попытка создать единые правила была предпринята в 1846 году, когда встретились представители нескольких колледжей. Они установили первый свод правил. В 1855 году был основан первый специализированный футбольный клуб — «Шеффилд». В 1863 году после долгих переговоров был принят свод правил Футбольной Ассоциации Англии. Также были приняты размеры поля и ворот. А в 1871 году был основан Кубок Англии — старейший футбольный турнир в мире. В 1891 году было принято правило о пенальти. Но сначала пенальти билось не с точки, а с линии, которая также как и сейчас находилась на расстоянии 11 метров от ворот.

Планируется создать игру футбол, в которой пользователь будет играть против компьютера командами 3х3. При этом, игра будет с элементами развития своей команды.

**В ходе курсового проектирования были выделены следующие разделы:**

1. **Анализ принципов тестирования и аналогов ПО. В данном разделе описаны основные принципы проведения тестирования.**
2. **Разработка алгоритма. Здесь описан процесс разработки алгоритма.**
3. **Разработка программного средства. Раздел посвящен полной разработке приложения, выбору структур данных.**
4. **Обоснование технических приемов программирования. Здесь были описаны основные модули и функции.**
5. **Тестирования. Тут было проведено полное тестирование приложения.**
6. **Руководство пользователя. Краткое руководство о том, как пользоваться приложением.**

**1 АНАЛИЗ ИГРЫ ФУТБОЛ И АНАЛОГОВ ПО**

* 1. Основные сведения о футболе и правила игры

Футбол (от англ. *foot* — ступня, *ball* — мяч) — командный вид спорта, в котором целью является забить мяч в ворота соперника ногами или другими частями тела (кроме рук) большее количество раз, чем команда соперника. В настоящее время самый популярный и массовый вид спорта в мире.

Правила игры:

Есть 17 официальных правил игры, каждое из которых содержит список оговорок и руководящих принципов. Эти правила предназначены для применения на всех уровнях футбола, хотя есть некоторые изменения для таких групп, как юниоры, взрослые, женщины и люди с ограниченными физическими возможностями. Законы очень часто формулировались в общих чертах, которые позволяют упростить их применения в зависимости от характера игры. Правила игры публикуются в ФИФА, но поддерживаются Международным советом футбольных ассоциаций (IFAB).

Каждая команда состоит максимум из одиннадцати игроков (без учета запасных), один из которых должен быть вратарём. Правила неофициальных соревнований могут уменьшить количество игроков, максимум до 7. Вратари являются единственными игроками, которым позволено играть руками при условии: они делают это в пределах штрафной площади у своих собственных ворот. Хотя есть различные позиции на поле, эти позиции не обязательны.

Отдельная футбольная игра называется матч, который в свою очередь состоит из двух таймов по 45 минут. Пауза между первым и вторым таймами составляет 15 минут, в течение которой команды отдыхают, а по её окончании меняются воротами.

Цель игры — забить мяч в ворота противника, сделать это как можно большее количество раз и постараться не допустить гола в свои ворота. Матч выигрывает команда, забившая большее количество голов.

В случае, если в течение двух таймов команды забили одинаковое количество голов, то или фиксируется ничья, или победитель выявляется согласно установленному регламенту матча. В этом случае может быть назначено дополнительное время — ещё два тайма по 15 минут каждый. Как правило, между основным и дополнительным временем матча командам предоставляется перерыв. Между дополнительными таймами командам даётся лишь время на смену сторон. Одно время в футболе существовало правило, по которому победителем объявлялась команда, первой забившая гол (правило «золотого гола») или выигрывавшая по окончании любого из дополнительных таймов (правило «серебряного гола»). В настоящий момент дополнительное время либо не играется вовсе, либо играется в полном объёме (2 тайма по 15 минут). Если в течение дополнительного времени победителя выявить не удаётся, проводится серия послематчевых пенальти, не являющихся частью матча: по воротам противника с расстояния 11 метров пробивается по пять ударов разными игроками. Если количество забитых пенальти у обеих команд будет равным, тогда пробиваются по одной паре пенальти, пока не будет выявлен победитель.

Поле:

Матчи могут проводиться как на полях с естественным, так и на полях с искусственным покрытием. Согласно официальным правилам игры в футбол искусственное покрытие должно быть зелёного цвета. Поле для игры имеет форму прямоугольника. Боковая линия обязательно должна быть длиннее линии ворот. Принималось решение о том, что размер поля должен составлять 100—110 м (110—120 ярдов) в длину и минимум 64—75 м (70—80 ярдов) в ширину, однако затем обязательность этого требования была приостановлена.

Разметка:

Ширина разметки

Разметка поля делается линиями шириной не более 12 см (5 дюймов); эти линии входят в площади, которые они ограничивают. Все линии должны быть одинаковой ширины.

Название линий поля

Две длинные линии, ограничивающие поле для игры, называются боковыми линиями; две короткие линии — лицевыми линиями или линиями ворот, так как на них располагаются ворота.

Средняя линия

Поле делится на две половины с помощью средней линии, соединяющей середины боковых линий. Посередине средней линии делается отметка центра поля — сплошной круг диаметром 0,3 м (1 фут). Вокруг центра поля проводится окружность радиусом 9,15 м (10 ярдов). С отметки центра поля в начале каждого из таймов основного и дополнительного времени, а также после каждого забитого гола, выполняется начальный удар. При исполнении начального удара все игроки должны находиться на своих половинах поля, а соперники выполняющей удар команды — и за пределами центрального круга.

Площадь ворот

На каждой половине поля размечается *площадь ворот* — зона, из пределов которой выполняется удар от ворот.

Из точек на расстоянии 5,5 м (6 ярдов) от внутренней стороны каждой стойки ворот, под прямым углом к линии ворот, вглубь поля проводятся две линии. На расстоянии 5,5 м (6 ярдов) эти линии соединяются другой линией, параллельной линии ворот. Таким образом, размеры площади ворот — 18,32 м (20 ярдов) на 5,5 м (6 ярдов).

Штрафная площадь

На каждой половине поля размечается штрафная площадь — зона, в которой вратарь может играть руками, а в ворота команды, совершившей в своей штрафной площади нарушение, наказуемое штрафным ударом, будет назначен 11-метровый удар.

Из точек на расстоянии 16,5 м (18 ярдов) от внутренней стороны каждой стойки ворот, под прямым углом к линии ворот, вглубь поля проводятся две линии. На расстоянии 16,5 м (18 ярдов) эти линии соединены другой линией, параллельной линии ворот. Размеры штрафной площади, таким образом — 40,32 м (44 ярда) на 16,5 м (18 ярдов). В пределах штрафной площади, по центру линии ворот и на расстоянии 11 м (12 ярдов) от неё, наносится одиннадцатиметровая отметка — сплошной круг диаметром 0,3 м (1 фут). За пределами штрафной площади проводится дуга окружности радиусом 9,15 м (10 ярдов), центр которой находится на одиннадцатиметровой отметке. Данная дуга используется для расположения игроков команд при пробитии одиннадцатиметрового удара.

Угловые сектора

В каждом из четырёх углов поля проводится дуга радиусом 1 м (или 1 ярд) с центром в углу поля, ограничивающая сектор для исполнения угловых ударов.

На расстоянии 9,15 м (10 ярдов) от границ угловых секторов у боковых линий и линий ворот могут быть нанесены отметки (с внешней стороны линий, примыкающие к ним под прямым углом), используемые для определения расстояния, на котором находятся игроки при исполнении углового.

Также у каждого угла поля обязательно ставятся флаги на флагштоках высотой не менее 1,5 метра (5 футов), не имеющих сверху заострений.

Мяч:

Футбольный мяч должен быть сферической формы, используемый для игры в **футбол**, параметры которого регламентируются Правилом 2 Правил игры в футбол.

Мяч состоит из 3 частей: покрышки, подкладки и камеры.

Покрышка — верхняя оболочка мяча, по которой наносятся удары.

Подкладка — средняя оболочка мяча, от толщины которой зависит прочность мяча (чем толще, чем прочнее)

Камера — центральная оболочка мяча, в которую закачивается воздух для более удобного нанесения ударов.

## Стандартные положения:

В футболе бывают стандартные положения. Стандартными положениями в футболе являются штрафные, свободные, угловые и прочие удары, которые наносятся по сигналу судьи.

Стандартными положениями являются:

* Начальный удар. Наносится в начале каждого тайма, а также — после каждого забитого мяча. Назначается с центральной точки поля (в центральном круге).
* Вбрасывание мяча (аут). Бросается руками из-за боковой линии. Назначается после того, как мяч эту самую боковую линию пересёк. При этом аут бросает соперник игрока, которого мяч коснулся последним перед уходом за боковую линию.
* Удар от ворот. Наносится вратарём, после того, как мяч полностью пересёк линию ворот (вне территории ворот) от игрока нападавшей команды.
* Угловой удар. Наносится игроком нападающей команды из углового сектора. Назначается в случае, если мяч полностью пересекает линию ворот (вне территории ворот) от игрока защищавшейся команды.
* Свободный удар. Назначается в случае опасной игры против соперника (несостоявшееся нарушение) в ворота команды, которая совершила опасную игру. Пробивается с точки, где произошёл момент опасной игры. Гол, забитый прямым ударом со свободного не засчитывается.
* Штрафной удар. Назначается в случае нарушения правил в ворота команды, нарушившей правила. Может быть назначен только за пределами штрафной площадки команды-нарушителя (в случае, если фол произошёл в пределах штрафной, назначается пенальти). Штрафной также, как и свободный удар пробивается с точки нарушения. Гол, забитый прямым ударом со штрафного засчитывается.
* Пенальти. Наносится со специальной отметки, расположенной в 11 метрах от ворот. Назначается в случае, если игрок нарушает правила в своей штрафной.
* Спорный мяч. Совершается судьёй, скидывающим мяч между двумя игроками-соперниками. Назначается в случае, если игра была остановлена в ситуации, не связанной с правилами. Напоминает хоккейное вбрасывание.
  1. Краткий анализ уже созданного ПО

В футбольном менеджере «11х11», и надеемся, что игра доставит вам удовольствие! Жанр футбольного менеджера предполагает, что участник выступает в качестве главного тренера, его основная задача — подобрать наиболее эффективную тактику против конкретного соперника, правильно выставить настройки на игру и успешно выступить в турнире, умело распределив силы футболистов на серию матчей.

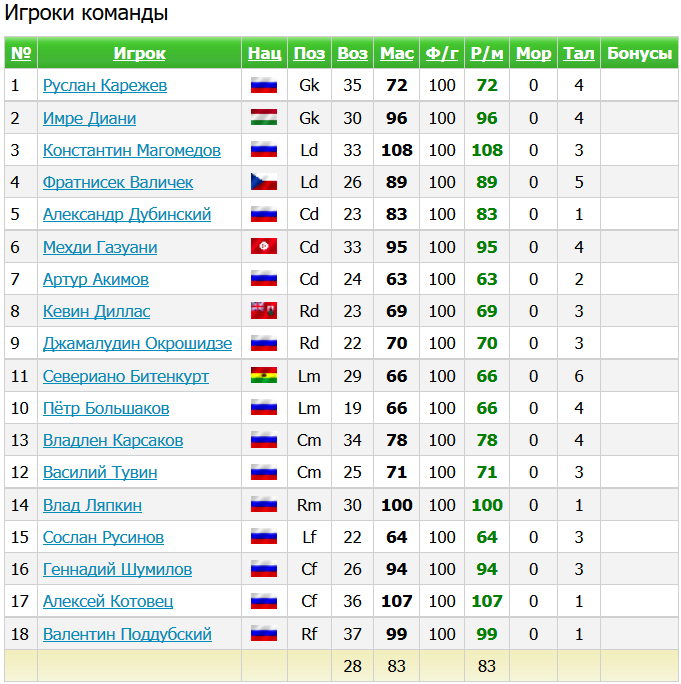


Рисунок 1.2.1 – Список игроков 11х11

Ваша цель в игре — одержать победу в наибольшем количестве матчей, набрать максимальное количество баллов в призовых турнирах и занять достойное место в рейтинге.



Рисунок 1.2.2 – Перед началом матча 11х11

После регистрации в игре Вы получаете команду, состоящую из 18 футболистов, каждый из которых уже имеет специализацию в соответствии с его позицией на поле. Перейдя в список игроков, Вы можете посмотреть их характеристики.

Для того, чтобы сыграть товарищеский матч, следует либо подать заявку на странице Матчи — Товарищеские матчи, либо там же вызвать соперника, чья заявка уже находится в списке.

После подтверждения заявки открывается окно настройки тактики (билдер) команды. Вы можете выбрать расстановку из списка стандартных или создать собственный вариант, перетаскивая мышкой игроков на поле. При несоответствии позиции и амплуа футболиста, обозначение амплуа подсвечивается красным цветом. Замены могут производиться автоматически, при получении травмы футболист заменяется запасным с тем же амплуа или же, если такого нет, другим полевым игроком. Чтобы замена была возможна, запасных игроков следует поместить на "скамейку запасных" - места сразу под зеленым полем в окне настройки-расстановки состава на матч. По ссылке "Замены" можно выбрать пары футболистов и выставить время замен (число в минутах от начала матча), и тогда они будут происходить во время ближайшей остановки игры после наступления времени замены.  
Тактика на матч настраивается путем выбора баланса нападение-защита, а также выбирается манера игры из стандартного набора инструкций.  
Дополнительно могут быть определены исполнители во время стандартных положений - угловой, штрафной, пенальти.

FIFA — серия симуляторов футбола, которая разрабатывается студией EA Canada, входящей в состав корпорации Electronic Arts. Игры издаются под брендом EA Sports™, каждый год выходит новое издание, включающее в себя изменения, произошедшие в футбольном мире за год. Electronic Arts владеет множеством лицензий на использование в игре футбольных лиг разных стран и игроков, выступающих в этих чемпионатах.



Рисунок 1.2.3 – Скриншот из игры FIFA

Особенности FIFA:

* Переработанные стандарты
* Активная система интеллекта
* Переработанная физическая модель
* Атакующие возможности
* Кроссплатформенный движок FROSTBITE
  1. Постановка требований к разрабатываемому ПО

Создать интересную игру с интуитивно понятным и красивым интерфейсом, в которой будет присутствовать довольно сильный искусственный интеллект. Так же, необходимо создать возможность прокачивать свою команду и зарабатывать на победах.

Создать:

* удобный и интуитивно понятный интерфейс для пользователя, играющего в футбол
* реализовать систему прокачки команды, каждого игрока в отдельности, своего игрового аккаунта
* создать систему статистики и механизм определения места игрока в ней
* создать понятное описание игры и цен на прокачку своего аккаунта и команды

**2 РАЗРАБОТКА АЛГОРИТМА**

Изначально было решено хранить тесты в текстовых файлах. Но файлы могут быть изменены без помощи приложения и защитить их средствами ОС практически невозможно. Было решено реализовать механизм проверки файлов при помощи хеширования перед их открытием. После проверки файла при помощи хеширования можно открыть при помощи функции OpenTest, алгоритм которой представлен ниже, а сохранить тест при помощи функции SaveTest.



а

Схема 2.1 – Алгоритм чтения теста из файла и хеширования данных:

а – алгоритм загрузки теста из файла, б – алгоритм получения хеша, в – алгоритм получения соли

Была создана схема работы программы.



Схема 2.2 – Схема работы программы

**3 РАЗРАБОТКА ПРОГРАММНОГО СРЕДСТВА**

**3.1** Выбор структур данных

Для работы разрабатываемого приложения необходимо использовать динамические структуры данных. Первым делом, необходимо выбрать способ представления теста в памяти. Были рассмотрены варианты: представление массивом и списком. После анализа двух способов хранения было решено выбрать двунаправленные списки, так как по сравнению с массивом они обладают рядом преимуществ, таких как более быстрое удаление элементов, более простое добавление новых заданий. Список выбран двунаправленным для того, чтобы была возможность перемещаться по списку в обоих направлениях и вносить либо ответы (со стороны пользователя), либо вносить изменения в задания (со стороны администратора). Для хранения вариантов ответов использовались однонаправленные списки, которые присоединялись к элементам списка с заданиями. Так как по вариантам ответа нет необходимости перемещаться в обе стороны, использование однонаправленного списка желательно (позволяет реализовать более простое добавление ответов).

Были созданы структуры данных типа Record TQuestion и TOption для хранения заданий и вариантов ответов соответственно.

TQuest и TOpt – указатели на структуры TQuestion и TOption соответственно.

Структура данных типа запись TQuestion включает следующие поля:

* ID
* Title
* Time
* TOQ
* Options
* Mark
* Answers
* Solved
* PrevQuest
* NextQuest

ID – поле типа integer, содержащее номер вопроса. Используется при прохождении теста пользователем.

Title – поле типа string, содержащее заголовок вопроса. Первый элемент списка, содержащего задания, содержит название теста.

Time – поле типа integer, содержащее время в секундах, отведенное на тест. Хранится только в первом элементе теста.

TOQ – поле типа string, содержащее тип вопроса: checkbox либо radio.

Options – поле типа TOpt, содержащее указатель на начало списка, состоящего из вариантов ответов на данный вопрос.

Mark – поле типа integer, содержащее количество баллов, которое будет добавлено к конечному результату в случае если пользователь успешно ответит на данный вопрос.

Answers – поле типа string, содержащее ответы пользователя в виде ‘245’

Solved – поле типа boolean, содержащее флаг: верно или неверно пользователь ответил на заданный вопрос.

PrevQuest – поле типа TQuest, содержащее указатель на предыдущий элемент списка, состоящего из вопросов теста.

NextQuest – поле типа TQuest, содержащее указатель на следующий элемент списка, состоящего из вопросов теста.

Структура данных типа запись TOption включает следующие поля:

* ID
* Info
* NextOpt
* Answer

ID – поле типа integer, содержащее порядковый номер данного варианта.

Info – поле типа string, содержащее вариант ответа. Был выбран тип string для создания более универсальной структуры данных.

NextOpt – поле типа TOpt, содержащее указатель на следующий элемент списка, состоящего из вариантов ответов на данный вопрос.

Answer – поле типа boolean, содержащее флаг, показывающий, является ли данный ответ верным ответом на заданный вопрос.

**3.2** Работа с файлами

Программа должна поддерживать функции прохождения тестирования, создания и редактирования тестов. Но создавать каждый раз новый тест и хранить его в оперативной памяти бессмысленно, поэтому было принято решение хранить файлы с тестами в текстовых файлах в директории /Tests/, находящейся рядом с исполняемым файлом. Названия тестов, дабы избежать использования зарезервированных слов и других ограничений было принято использовать вида “testN.txt”, где N – натуральное число.

После прохождения тестирования необходимо сохранять результаты прохождения теста в файл. После того, как пользователь ответит на вопросы (или не ответит), он вводит фамилию и получает результаты в удобном виде как в самом приложении, так и в виде текстового файла. Хранить результаты пользователей было решено в директории /Results/, находящейся рядом с исполняемым файлом. Если пользователь с заданной фамилией еще не проходил тестирование, то будет создана директория, название которой – фамилия пользователя, и в нее будет записан файл с результатом. Если уже проходил, то в нее просто запишется файл с текущим результатом. К названиям файлом с результатом добавляется хеш, позволяющий избежать коллизий при записи файлов в одну директорию.

Если какие-либо директории отсутствуют (например, /Tests/, /Results/…), то они будут созданы приложением автоматически. Но возникает еще одна проблема – это возможность изменять файлы. А если приложение начнет разносить по списку измененные данные, то это может привести к серьезным проблемам в виде падения программы и др. Было решено при создании и редактировании теста генерировать хеш с использованием соли таким, что при открытии файла его можно будет легко проверить. Таким образом, первая строка теста составляет некоторый числовой хеш. При открытии файла происходит построчное считывание всего файла и генерация хеша на его основе. Затем к нему добавляется соль и полученное значение сравнивается с хешем, находящимся на первой строке файла с тестом. Если хеши совпали, то файл редактировался/создавался с использованием оригинального приложения. Имея лишь скомпилированное приложение нельзя в общем случае сгененировать вручную верный хеш для файла с тестом, так как соль находится в .exe файле (но нельзя исключать факт ее просмотра при помощи дизассемблера). В итоге: при изменении любого символа файла с тестом хеш, сгенерированный приложением на лету и хеш, записанный в файл на этапе редактирования, совпадать не будут, а значит приложение не будет пытаться создать из данного файла тест.

**4 ОБОСНОВАНИЕ ТЕХНИЧЕСКИХ ПРИЕМОВ**

**ПРОГРАММИРОВАНИЯ**

Изначально было решено отделить логику работы с содержимым мультисписка от логики работы с элементами графического интерфейса.

В модуль UnitModel была вынесена основная логика работы приложения. Для облегчения процесса разработки и дальнейшего поддержания кода было решено разрабатывать процедуру/функцию для каждой отдельной задачи.

Для открытия и загрузки данных из файла с тестом было решено создать функцию, принимающую в качестве аргумента имя файла и возвращающая указатель типа TQuest на мультисписок, содержащий тестовые данные, загруженные из текстового файла. Проверка теста на существование производится в модулях работы с формами. Для проверки файла с тестом на правильность производится генерация хеша с добавлением соли на основе всех строк файла, а затем и проверка на совпадение его с тем, который записан в первой строке текстового файла. Функционал сверки хешей реализован в функции IsTestValid, принимающей в качестве аргумента название файла с тестом и возвращаю true/false в зависимости от результатов проверки. Генерация хеша на основе файла происходит в функции GetFileHash, принимающей строку с названием файла и возвращающая целочисленное значение хеша. В GetFileHash используется функция GetHash, которая вычисляет хеш от переданной строки. Соль представляет собой функцию GetSalt, возвращающую некоторое уникальное заданное значение. Перед сохранением теста в файл генерируется такой хеш, аналогичный которому может быть получен и при построчном его создании из файла. Это выполняет функция GetQuestHash, которая принимает заголовок на мультисписок и возвращает целочисленное значение хеша.

Реализованные функции NextQuest и PrevQuest возвращают значения предыдущих и следующих значений соответственно элемента TQuest.

После того, как пользователь ответит на все вопросы необходимо их проверить и узнать количество баллов, набранное пользователем. Для этого реализована функция GetMark, которая принимает на вход единственный параметр – заголовок мультисписка и возвращает количество набранных пользователем баллов. Для получения максимальной оценки разработана функция GetMaxMark. Принцип работы GetMark довольно прост – если вопрос выполнен верно (Solved = true), то к конечному результату добавляется балл за текущий вопрос. Перед вызовом функции GetMark вызывается процедура CheckQuest с один параметром – заголовком мультисписка. CheckQuest сверяет верные ответы с ответами пользователя и устанавливает в элементе соответствующий флаг Solved true/false.

В данном модуле также имеются функции и процедуры, необходимые для работы от лица администратора. Это процедура CreateTest, создающая заголовок мультисписка, процедура FillFirstQuest, принимающая в качестве параметра указатель на заголовок мультисписка, название теста и время проведения теста, которая необходима для инициализации заголовочного элемента начальными данными. В процессе работы используются процедуры AddQuest, DeleteQuest, соответственно добавляющие и удаляющие переданные им элементы. Для работы с вариантами ответов используются: FillFirstOption (в качестве параметра принимает заголовок мультисписка) которая заполняет поле Options указателем на будущий однонаправленный список вариантов ответов. ClearOptions (принимает в качестве параметра указатель на элемент TQuest) необходима для очистки вариантов ответов у данного вопроса. А также функция AddOption, реализующая добавление новых вариантов ответов.

Имеется ряд вспомогательных функций в данном модуле, которые выполняют буквально несколько простых операций и перечисление их не имеет смысла.

В модуле UnitMain реализован простой интерфейс выбора Пользователь либо Администратор.

Для большей наглядности интерфейса использовалась библиотека с элементами AlphaControls. Из данной библиотеки использовались только элементы, аналогичные элементам Delphi, а, значит, они не могут никоем образом изменить логику работы программы.

В модулях UnitUser и UnitAdmin содержатся функции и процедуры, обеспечивающее взаимодействие пользователя с графическим интерфейсом. Изначально было решено ввести две глобальные переменные в обоих модулях – FQuest и CQ (первая отвечает за хранение указателя на мультисписок, вторая – за хранение указателя на текущий элемент). Так как в приложении реализована возможность перемещаться от вопроса к вопросу Вперед и Назад, данный вариант хранения двух указателей является наиболее оптимальным. Соответственно, устанавливая видимость кнопок (Visible := true/false) мы можем разрешать или запрещать перемещаться в обе стороны по тесту. Были созданы функции установки видимости кнопок, вывода содержимого текущего элемента на форму, функция сохранения данных с формы, а также функция вывода результата.

**5** **ТЕСТИРОВАНИЕ**

Испытания на представительном множестве примеров с выполнением проверок правильности результатов работы, которые программа позволяет получать, показали корректность работы приложения.

Сперва протестируем интерфейс администратора:

Заходим в приложение. Нажимаем кнопку «Администратор», выбираем тест двойным нажатие и переходим в окно редактирования названия теста и таймера. Если ввести данные неверно, то появится сообщение об ошибке. Ввести символы в поле с таймером не удалось. Ввести слишком длинную последовательность символов в поля названия и таймера не удалось.

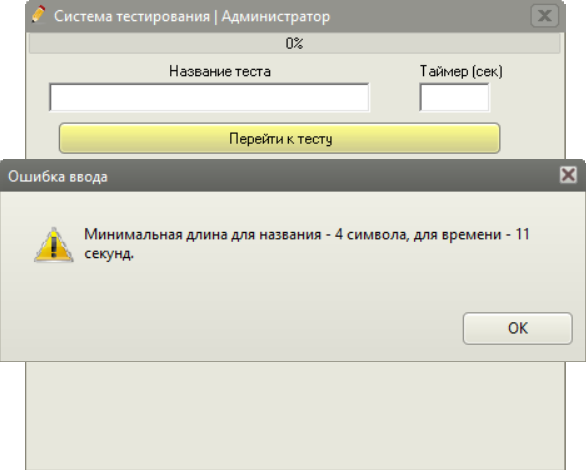


Рисунок 5.1 – Ошибка ввода

Заполняем поля верно и переходим в интерфейс редактирования вопросов.

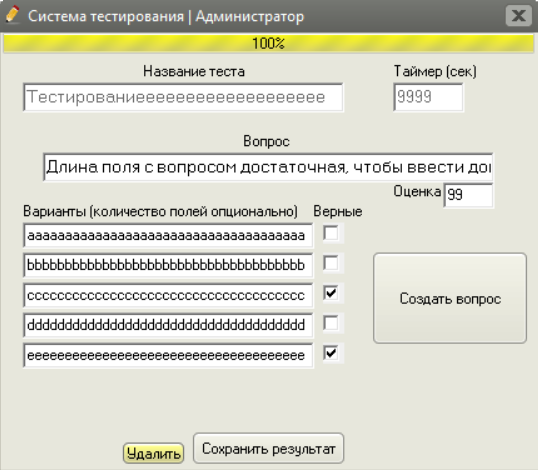


Рисунок 5.2 – Корректное заполнение полей программы

Для тестирования уберем все верные варианты ответа и затем при клике по любой кнопке получаем ошибку как на рисунке 5.3:

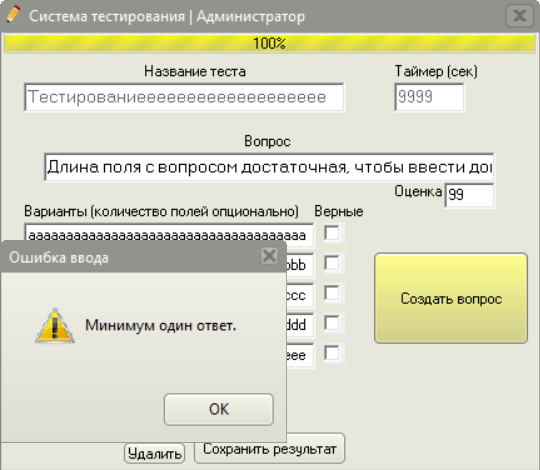


Рисунок 5.3 – Ошибка: нет верного варианта ответа

Вводить символы, отличные от цифр в поле Оценка невозможно. Длины строк, вводимых в поля, ограничены.

Удаление единственного вопроса в тесте вызывает ошибку:

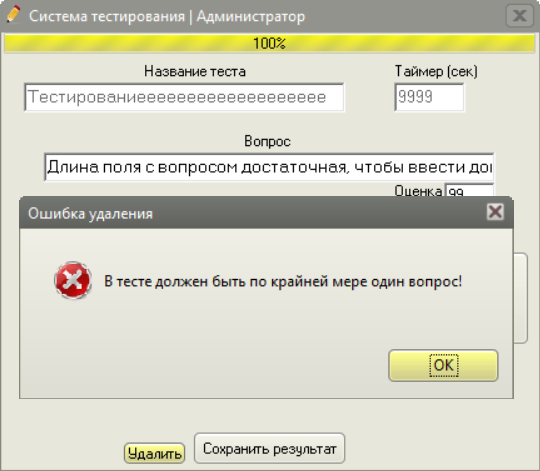


Рисунок 5.4 – Ошибка: единственность вопроса

При сохранении теста: если директория, в которую необходимо записать файл, она будет создана автоматически, а в нее будет корректно помещен файл.

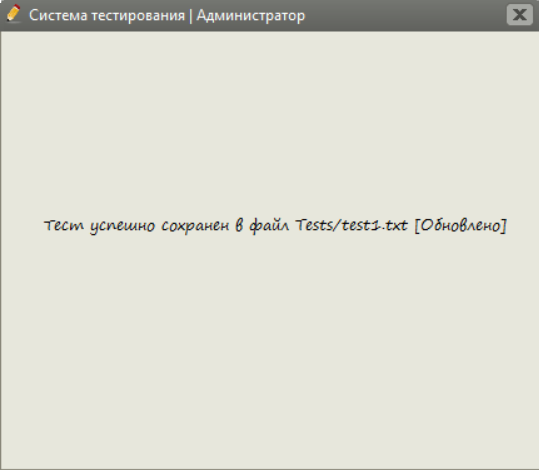


Рисунок 5.5 – Автоматическое создание директории

Приложение в своей работе использует файлы. Если файлов с тестами нет, нет директории, в которой они находятся, нет файлов с правильными хешами, то приложение продолжит стабильную работу.

Для проверки целостности файлов используется хеширование, алгоритм которого описан выше, поэтому если в файл внести какие-либо изменения, то приложение откажет в работе с ним.

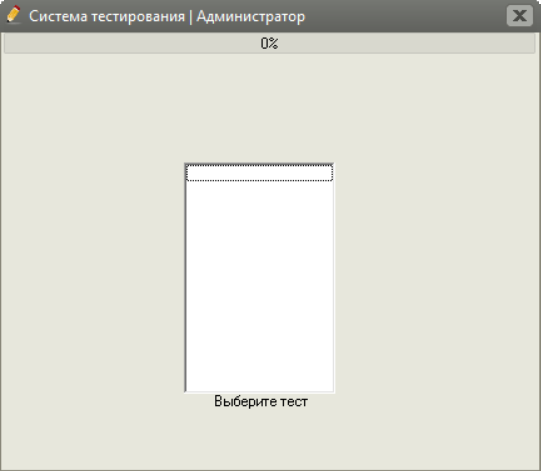


Рисунок 5.6 – Отсутствие файлов с тестами

Перейдем к тестированию пользовательской части приложения

Заходим в приложение, выбираем «Пользователь» двойным нажатием тест (если тестов нет, либо они невалидны, то будет пустой список – протестировано выше). Нажимаем единственную кнопку «Начать ТЕСТ!». Далее запускается таймер и открываются вопросы, на которые нужно отвечать.

В приложении не появляется ошибок, если не отвечать на вопросы. Нельзя выбрать более одного варианта, если только это не предусмотрено вопросом, где необходимо выбрать несколько вариантов ответов.

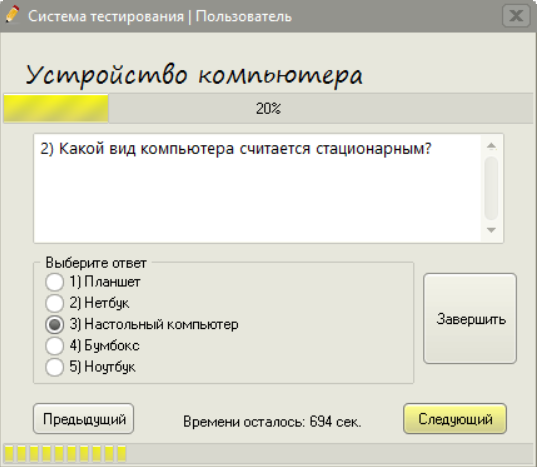


Рисунок 5.7 – Корректная работа пользовательской части

Пользователь может досрочно завершить выполнение теста, нажав кнопку «Завершить». Выполнение теста завершится автоматически по прошествии времени, отведенного на тест. Кнопка «Завершить» появляется, если пользователь достиг последнего вопроса. Выйти во время выполнения теста из приложения без нажатия кнопки Завершить или пока не закончится время, нельзя.

После завершения выполнения теста (нажатие по кнопке или окончание времени выполнения) предлагается ввести фамилию тестируемого, которая может состоять только из символов русского и английского алфавитов, а так же знака «-». Иные символы ввести нельзя, минимальная длина – 3 символа.

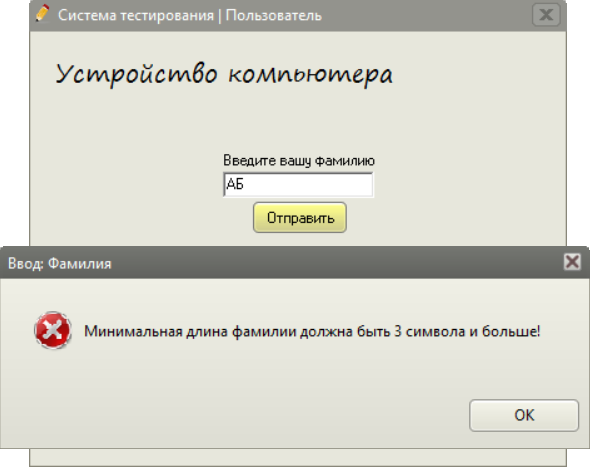


Рисунок 5.8 – Ошибка: длина фамилии

Если ввести фамилию верно, то результаты сохраняются в файл в папке /Results/Введенная\_фамилия/ (если такой не существует, то она создается и корректно выводятся на форму).

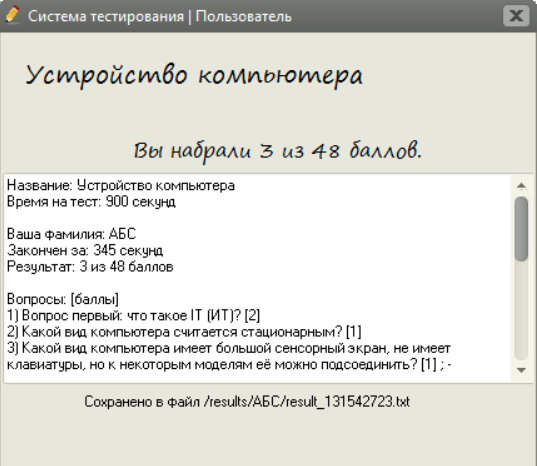
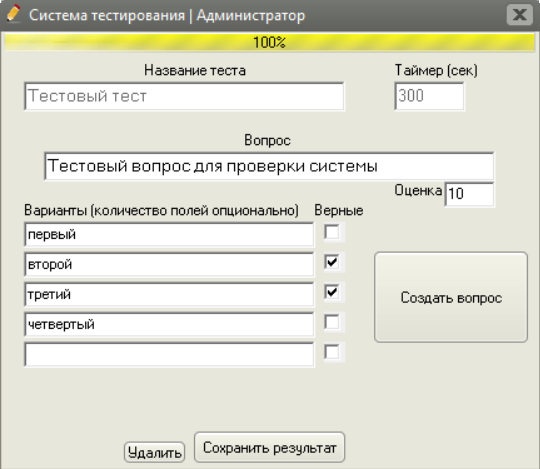
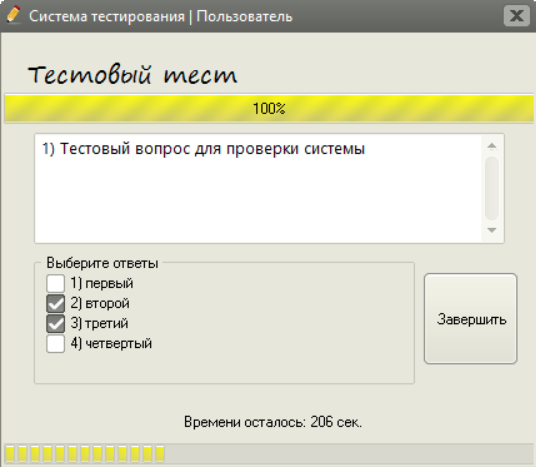
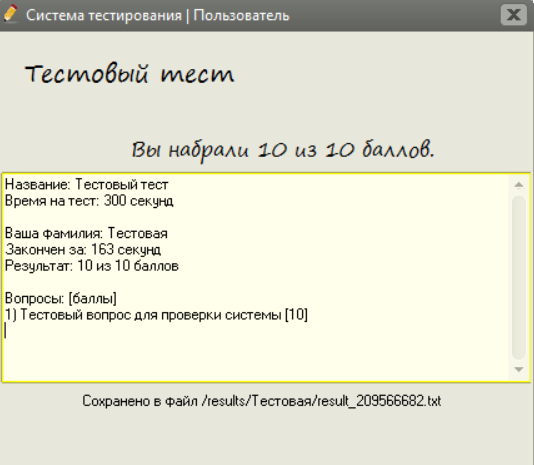


Рисунок 5.9 – Вывод результатов тестирования

Основную задачу – проверку пользовательских ответов на тест приложение выполняет отлично. Создадим тест с одним вопросом и четырьмя вариантами ответа (первый, второй, третий, четвертый). При это второй и третий – верные варианты ответа. Далее пройдем данный тест со стороны пользователя, выберем верными второй и третий варианты и в результате получим, что тест мы прошли верно. Это доказывает тот факт, что система работает отлично.



а б в

Рисунок 5.10 – Корректная выдача результата при прохождении созданного теста

а – создание теста, б – прохождение созданного теста, в – получение результатов

Проведенные испытания программы показывают корректность её работы.

Тестирование показало, что приложение способно корректно работать при любом несанкционированном поведении пользователя. Опыт работы с ней свидетельствует о дружелюбности и высокой наглядности интерфейса, что дает основание рекомендовать данное приложение для применения в образовательной сфере.

**6 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**

**6.1 Создание теста**

Для создания теста необходимо:

1. Запустить программу.
2. Нажать кнопку «Администратор».
3. Нажать кнопку «Создать тест».
4. Ввести название теста в соответствующее поле.
5. Ввести время на тест в секундах в соответствующее поле.
6. Нажать кнопку «Перейти к тесту» (при возникновении ошибок – прочитать ошибку и исправить ее).
7. Открылся интерфейс добавления вопросов, в котором необходимо заполнить все поля (при этом поля вариантов ответов опциональны, но не должно быть менее одного варианта ответа и одного верного варианта ответа).
8. Чтобы добавить новый вопрос нужно нажать кнопку «Создать вопрос».
9. Чтобы удалить вопрос нужно нажать кнопку «Удалить» и произойдет переход либо на предыдущий либо на следующий вопрос, а текущий вопрос удалится.
10. Чтобы вернуться к предыдущему вопросу необходимо нажать кнопку «Предыдущий»
11. Чтобы перейти к следующему вопросу необходимо нажать кнопку «Следующий»
12. Пока необходимо создание/редактирование/удаление вопросов/вариантов ответов нужно повторять шаги 7-9.
13. Если нажать кнопку «Сохранит результат», то тест сохранится в файл с названием, которое отобразится на форме .
14. Для выхода из программы нужно нажать на крестик вверху справа и подтвердить выход

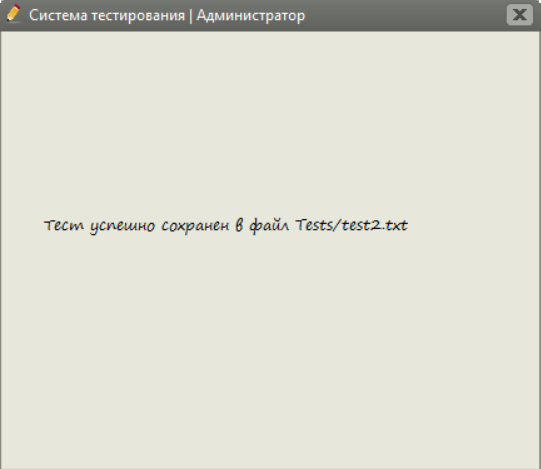


Рисунок 6.1.1 – Сохранение результата в файл

**6.2 Редактирование теста**

Для редактирования теста необходимо:

1. Запустить программу.
2. Нажать кнопку «Администратор».
3. Нажать кнопку «Редактировать тест».
4. Выбрать двойным кликом нужный тест из списка.
5. Отредактировать название теста в соответствующем поле.
6. Отредактировать время на тест в секундах в соответствующем поле
7. Нажать кнопку «Перейти к тесту» (при возникновении ошибок – прочитать ошибку и исправить ее).
8. Открылся интерфейс добавления/редактирования вопросов, в котором необходимо заполнить все поля (при этом поля вариантов ответов опциональны, но не должно быть менее одного варианта ответа и одного верного варианта ответа).
9. Чтобы добавить новый вопрос нужно нажать кнопку «Создать вопрос».
10. Чтобы удалить вопрос нужно нажать кнопку «Удалить» и произойдет переход либо на предыдущий либо на следующий вопрос, а текущий вопрос удалится.
11. Чтобы вернуться к предыдущему вопросу необходимо нажать кнопку «Предыдущий»
12. Чтобы перейти к следующему вопросу необходимо нажать кнопку «Следующий»
13. Пока необходимо создание/редактирование/удаление вопросов/вариантов ответов нужно повторять шаги 7-9.
14. Если нажать кнопку «Сохранит результат», то тест сохранится в файл с прежним названием, которое отобразится на форме.
15. Для выхода из программы нужно нажать на крестик вверху справа и подтвердить выход.

**6.3 Прохождение теста**

Для прохождения теста необходимо:

1. Запустить программу
2. Нажать кнопку «Пользователь».
3. Выбрать двойным кликом нужный тест из списка, который вы хотите пройти.
4. Нажать кнопку «Начать ТЕСТ!». При этом запускается таймер. После этого вы перейдете к первому вопросу.
5. Выбрать вариант(-ы) ответа.
6. Чтобы вернуться к предыдущему вопросу нужно нажать кнопку «Предыдущий».
7. Чтобы перейти к следующему вопросу нажать кнопку «Следующий».

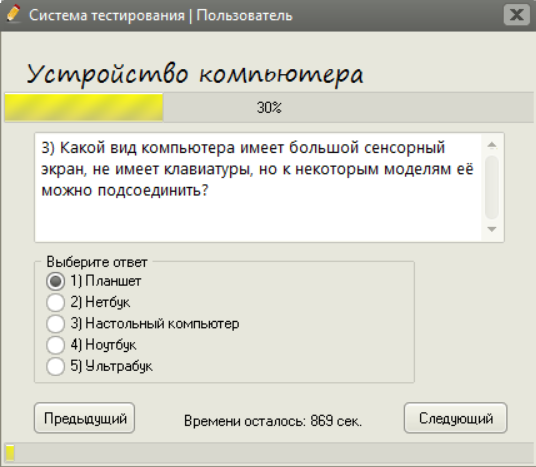


Рисунок 6.3.1 – Вид пользовательской части приложения

1. Работа над тестом завершится, если вы нажмете кнопку «Завершить», либо если закончится время, отведенное на выполнение теста.
2. Выходить из приложения досрочно запрещено в виду наличия таймера, отсчитывающего время, затраченное на тест.
3. После нажатия по кнопке «Завершить» либо если закончилось время вам будет предложено ввести свою фамилию

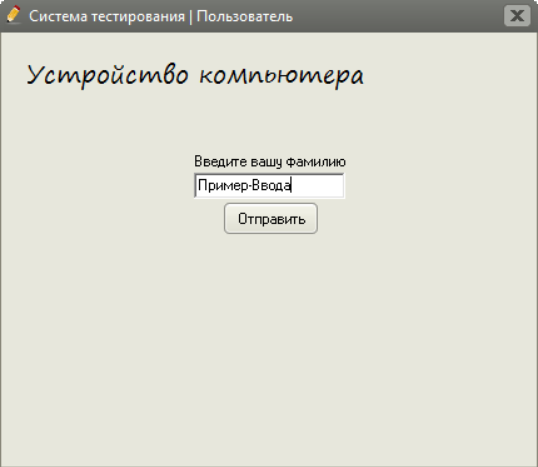


Рисунок 6.3.2 – Вид интерфейса ввода фамилии

1. Нажать кнопку «Отправить» (в случае возникновения ошибки – прочитать ее и исправить)
2. Получить результаты, которые будут также продублированы в файл, путь к которому будет также выведен на форму.

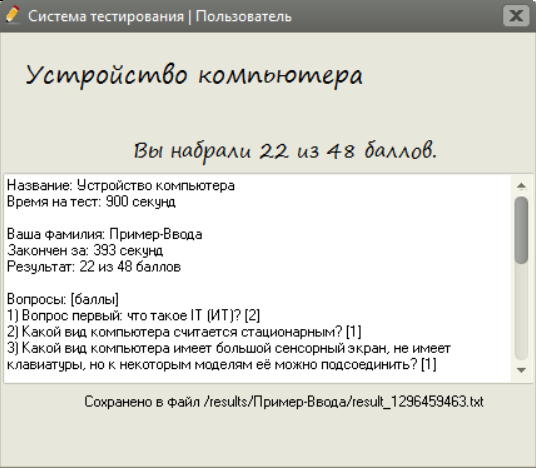


Рисунок6.3.3 –Вывод результатов теста

**ЗАКЛЮЧЕНИЕ**

В результате курсового проектирования было создано приложение, полностью соответствующее заранее определенным требованиям.

Для написания приложения понадобилось знание классических алгоритмов и структур данных.

Были изучены материалы по проведению тестирования, а так же по среде Delphi. В ходе изучения литературы были получены новые знания, как по Delphi, так и по принципам проведения тестирования.

Изначально поставленные задачи достигнуты. Приложения получилось простым и понятным как для пользователя, так и для администратора.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

[1] Википедия https://ru.wikipedia.org/wiki/Футбол

[2] Игра 11х11 http://11x11.ru

[3] Википедия https://ru.wikipedia.org/wiki/FIFA\_(серия\_игр)

[3] Приложение УСАТИК http://www.nestor.minsk.by/kg/2006/29/kg62917.html

[4] Приложение OpenTEST http://opentest.com.ua/kompyuternaya-sistema-testirovaniya-znanij-opentest/

[5] AlphaControls http://www.alphaskins.com/index.php

**ПРИЛОЖЕНИЕ А**

**(обязательное)**

**Текст модуля UnitMain**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Обозначение | | | | Наименование | | | | Дополнительные сведения | | | |
|  | | | | Текстовые документы | | | |  | | | |
|  | | | |  | | | |  | | | |
| БГУИР КП 1–40 01 01 551001 012 ПЗ | | | | Пояснительная записка | | | | 62 с. | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | | Графические документы | | | |  | | | |
|  | | | |  | | | |  | | | |
| ГУИР 551001 012 ПД | | | | Схема программы | | | | Формат А1 | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  |  |  |  |  | БГУИР КП 1-40 01 01 551001 012 Д1 | | | | | | |
|  |  |  |  |  |
|  |  |  |  |  |
| Изм. | Л. | № докум. | Подп. | Дата | Универсальная система тестирования знаний  Ведомость курсового  проекта |  | | | | Лист | Листов |
| Разраб. | | Коваленко И.А. |  | 30.05.16 | Т |  | |  | 62 | 62 |
| Пров. | | Болтак С.В. |  | 06.06.16 | Кафедра ПОИТ  гр. 551001 | | | | | |
|  | |  |  |  |