**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 5](#_Toc515004263)

[1. АНАЛИЗ ПРОТОТИПОВ, ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ И ФОРМИРОВАНИЕ ТРЕБОВАНИЙ К ПРОЕКТИРУЕМОМУ ПРОГРАММНОМУ СРЕДСТВУ 6](#_Toc515004264)

[1.1. Подсчет тестового балла 6](#_Toc515004265)

[1.2. Анализ существующих аналогов 8](#_Toc515004266)

[1.3. Постановка задачи 13](#_Toc515004267)

[2. АНАЛИЗ ТРЕБОВАНИЙ К ПРОГРАММНОМУ СРЕДСТВУ И РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ ТРЕБОВАНИЙ 14](#_Toc515004268)

[2.1. Описание функциональности ПС. 14](#_Toc515004269)

[2.2. Спецификация функциональных требований. 16](#_Toc515004270)

[3. ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА 17](#_Toc515004271)

[3.1. Разработка алгоритма генерации бланка 17](#_Toc515004272)

[3.2. Разработка алгоритма подсчета правильных ответов на задание 20](#_Toc515004273)

[3.3. Разработка алгоритма вычисления весовых коэффициентов заданий 21](#_Toc515004274)

[3.4. Разработка алгоритма создания бланков абитуриентов 22](#_Toc515004275)

[3.5. Разработка алгоритма подсчета итогового балла 23](#_Toc515004276)

[3.6. Динамический список 25](#_Toc515004277)

[4. СОЗДАНИЕ (КОНСТРУИРОВАНИЕ) ПРОГРАММНОГО СРЕДСТВА 27](#_Toc515004278)

[4.1. Формы в программе 27](#_Toc515004279)

[4.2. Структура модулей программы 28](#_Toc515004280)

[4.3. Описание модуля MyUnit 28](#_Toc515004281)

[4.4. Описание модуля Unit30zad 30](#_Toc515004282)

[4.5. Описание модуля Unit40zad 31](#_Toc515004283)

[4.6. Описание модуля Unit50zad 32](#_Toc515004284)

[4.7. Описание модуля UnitResult 33](#_Toc515004285)

[5. ТЕСТИРОВАНИЕ, ПРОВЕРКА РАБОТОСПОСОБНОСТИ И АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ 34](#_Toc515004286)

[5.1. Тестирование подсчета баллов 34](#_Toc515004287)

[5.2. Тестирование подсчета количества абитуриентов, правильно выполнивших задание 35](#_Toc515004288)

[5.3. Тестирование вычисления коэффициентов 36](#_Toc515004289)

[5.4. Тестирование производительности программы 37](#_Toc515004290)

[5.5. Тестирование сохранения результатов в историю 38](#_Toc515004291)

[5.6. Тестирование функционала окон и кнопок приложения 39](#_Toc515004292)

[5.7. Итоги тестирования 39](#_Toc515004293)

[6. РУКОВОДСТВО ПО УСТАНОВКЕ И ИСПОЛЬЗОВАНИЮ 40](#_Toc515004294)

[ЗАКЛЮЧЕНИЕ 45](#_Toc515004295)

[СПИСОК ЛИТЕРАТУРЫ 46](#_Toc515004296)

[ПРИЛОЖЕНИЕ 1 47](#_Toc515004297)

[ПРИЛОЖЕНИЕ 2 48](#_Toc515004298)

# ВВЕДЕНИЕ

Каждый гражданин имеет право на получение высшего образования в государственных учреждениях на конкурсной основе. Оценка уровня знаний кандидатов по соответствующим дисциплинам проводится на основании результатов централизованного тестирования.

Централизованное тестирование - форма вступительных испытаний, организованная на основе педагогических тестов, обработки, анализа и представления результатов, используемая для проведения конкурса при поступлении в учреждения высшего, среднего специального и профессионально-технического образования.[1]

Данная курсовая работа посвящена разработке программного средства централизованного тестирования. Программа имитирует процесс создания тестовых бланков и осуществляет автоматическую проверку результатов.

В основу методики положена 100-бальная шкала модифицированных первичных баллов, которая позволяет принять во внимание не только количество верно выполненных заданий теста, но также учесть сложность каждого выполненного или невыполненного задания, произвести корректировку тестового балла в зависимости от сложности выполненного варианта теста.[2]

Итоговый балл абитуриента за тест зависит от первичного балла задания, первичного тестового балла абитуриента и весового коэффициента сложности каждого задания.

В пояснительной записке отображены следующие этапы написания курсовой работы:

1. Анализ прототипов, литературных источников и формирование требований к проектируемому программному средству;
2. Анализ требований к программному средству и разработка функциональных требований;
3. Проектирование программного средства;
4. Создание (конструирование) программного средства;
5. Тестирование, проверка работоспособности и анализ полученных результатов;
6. Руководство по установке и использованию.

# 1. **АНАЛИЗ ПРОТОТИПОВ, ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ И ФОРМИРОВАНИЕ ТРЕБОВАНИЙ К ПРОЕКТИРУЕМОМУ ПРОГРАММНОМУ СРЕДСТВУ**

## Подсчет тестового балла

Для подсчёта баллов на централизованном тестировании используется специальная методика, которая помогает объективно оценить уровень знаний и подготовки абитуриентов.

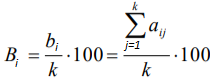
Для тестов с единственным правильным ответом: результат выполнения каждого задания оценивается по следующему (бинарному) принципу: ставится единица, если задание выполнено верно, и ноль, если задание выполнено неверно. Множество таких нулей и единиц образует некоторую матрицу ответов A = (aij) .

Количество абитуриентов, которые приняли участие в централизованном тестировании по предмету – n: i =1, 2,…,n. Педагогический тест состоит из k заданий различной трудности: j =1, 2,…,k.

Исходной информацией для обработки результатов тестирования служат матрицы AL , размером nL × k , где L = 1, 2, ..., m – номер варианта теста, m – количество вариантов теста, nL  – количество абитуриентов, выполнявших L - ый вариант теста.

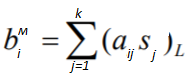
Сумма элементов матрицы AL по каждой строке bi = aij)L является суммой верно выполненных заданий i -м абитуриентом и называется первичным баллом i -го абитуриента. При сложении элементов по каждому столбцу (cj)L = aij)L получится число абитуриентов, верно выполнивших j -е задание, и эта величина называется первичным баллом j -го задания.

Отношение первичного балла абитуриента к числу заданий в тесте – это и есть процентная шкала распределения первичных баллов. Тестовый балл участника выражается соотношением:[2]

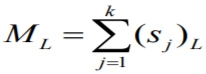


Весовой коэффициент – величина, характеризующая относительную сложность каждого задания в тесте:[3]

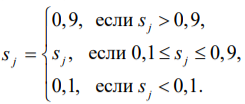
При умножении соответствующих элементов матрицы ответов AL на весовые коэффициенты ( sj)L , получаются элементы матрицы (aijsj)L значения которых (если i -ый абитуриент верно выполнил j -е задание) зависят от количества участников, не справившихся с данным тестовым заданием. Сумма элементов такой модифицированной матрицы ответов по строке называется модифицированным первичным баллом i - го абитуриента:



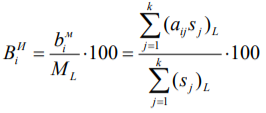
Абитуриент может набрать ML модифицированных первичных баллов:



Для ограничения достаточно больших различий в сложности отдельных заданий используется следующее ограничение, задаваемое априорно по схеме:

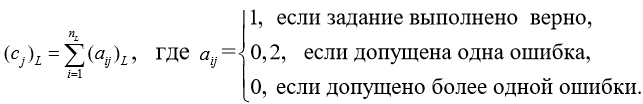


Формула для подсчета итогового тестового балла i -го абитуриента:



Округление итогового тестового балла до целого числа производится по правилам математического округления.

Подсчет первичного балла задания с несколькими вариантами ответов (для русского и белорусского языка) происходит следующим образом:



Дальнейшие этапы определения результата осуществляются аналогично случаю с единственно верным ответом. [2]

## Анализ существующих аналогов

В качестве аналогов я рассматриваю веб- и мобильные приложения.

**ЦТ помощник** – приложение для подготовки к централизованному тестированию по математике.

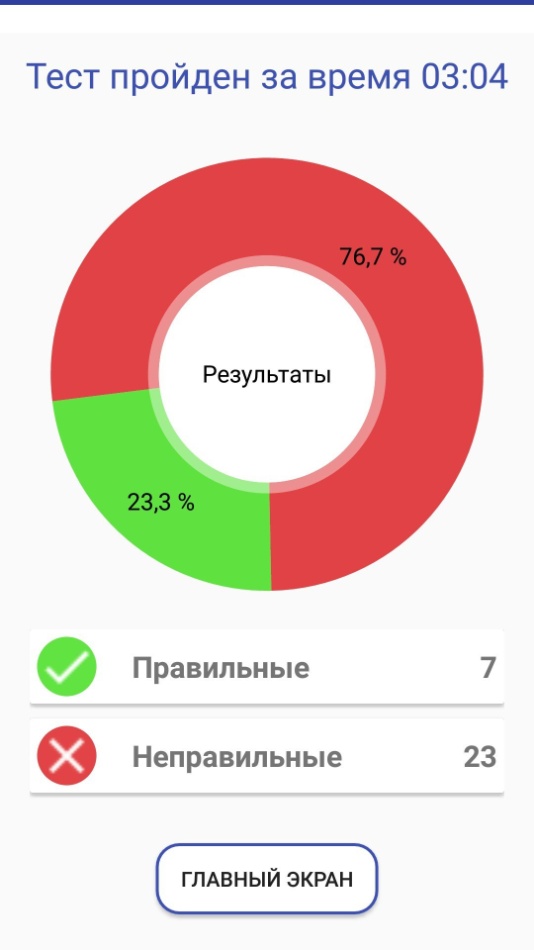
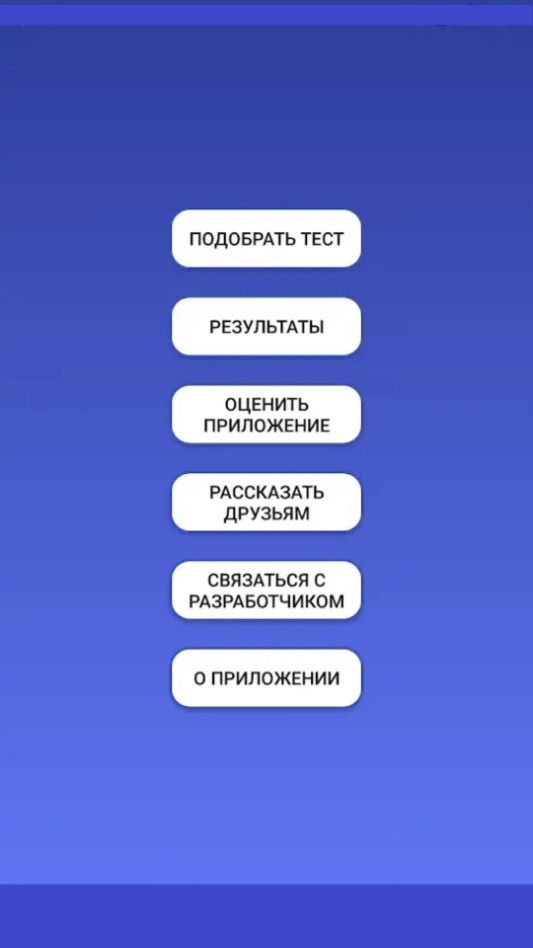
****

Рисунок 1.1 – Скриншот ЦТ

Достоинства:

* Простой пользовательский интерфейс.
* Можно делать заметки прямо в задании.
* Предоставлено несколько вариантов теста.
* Предоставляется бесплатно.

Недостатки:

* Из тестов по предметам доступна только математика.
* После прохождения теста нет итогового балла (выводится количество правильных и неправильных ответов и их процентное соотношение).
* Не отображаются номера заданий, которые были решены неправильно.

**Адукар ЦТ** – интернет-каталог образовательных программ.

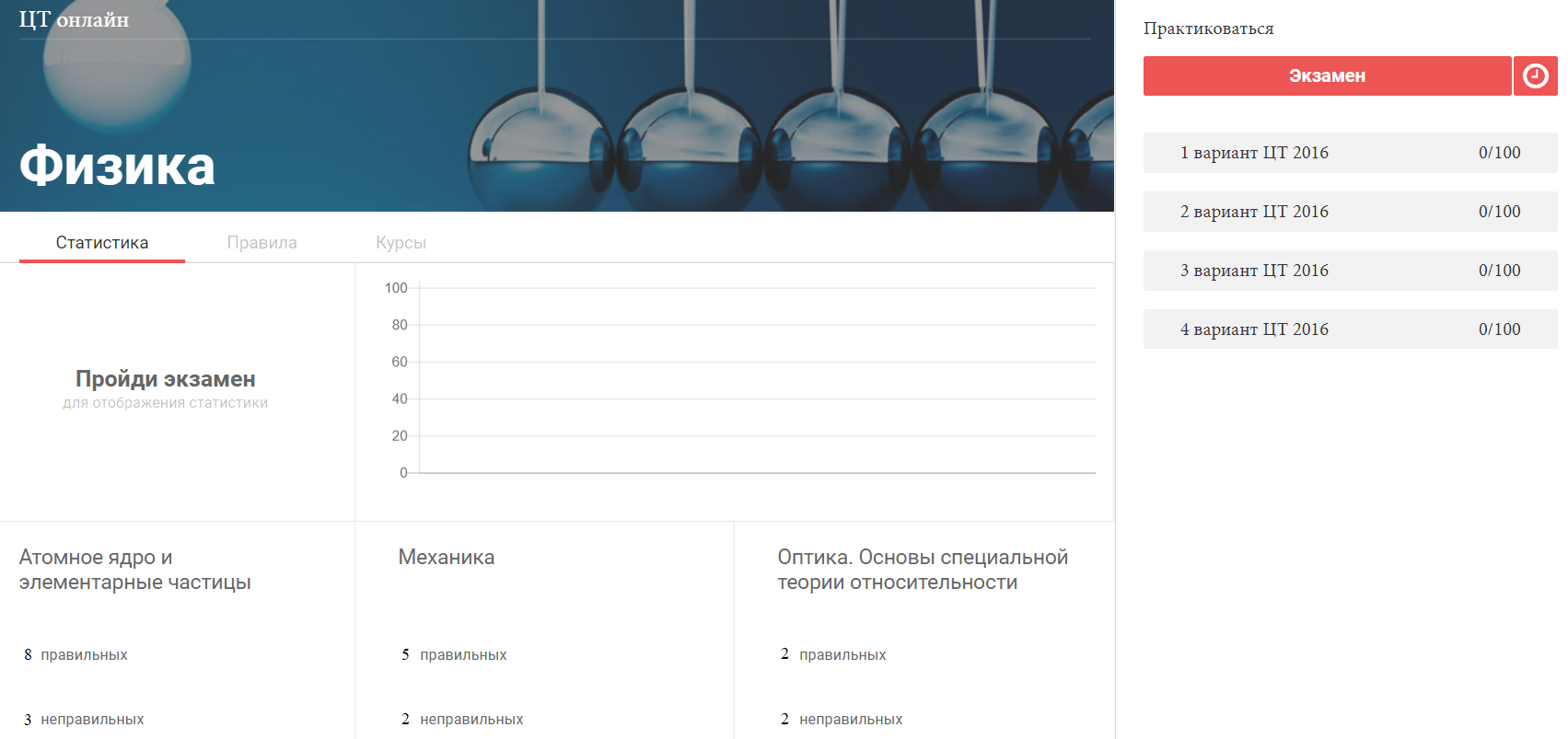


Рисунок 1.2 – Скриншот Адукар ЦТ

Достоинства:

* Представлено 9 предметов для прохождения тестирования.
* Несколько вариантов для каждого теста.
* Вместе с отображением неправильных ответов указывается тема предмета, по которой пользователь совершил ошибку.
* При повторном прохождении теста статистика сохраняется.
* Пользователь всегда может продолжить тест с того места, где он остановился.
* После ответа на вопрос, можно сразу узнать правильный вариант.
* Предоставляется бесплатно.

Недостатки:

* Итоговый бал подсчитывается с погрешностью, так как результат должен зависеть от того, как ответили другие абитуриенты. В этой программе используются усредненные коэффициенты.
* Много всплывающих окон-подсказок. Пользователю приходится постоянно закрывать.
* Прохождение теста доступно только авторизованным лицам.

**Тесты ЦТ** – онлайн тесты по разным предметам за 2012, 2013, 2014 год**.**

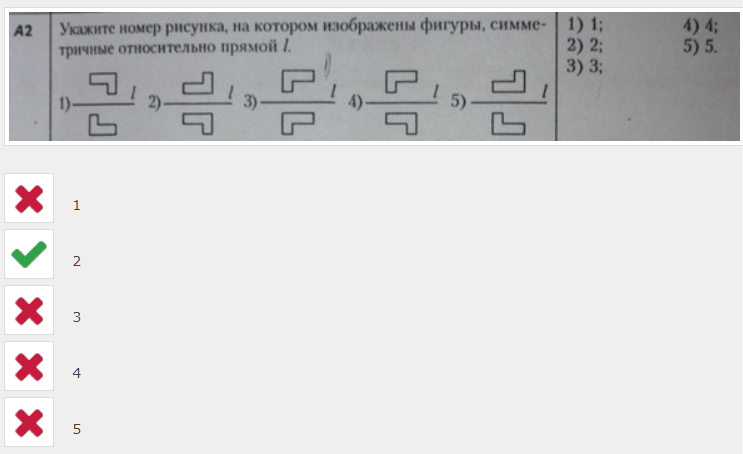
****

Рисунок 1.3 – Скриншот Тесты ЦТ

Достоинства:

* Представлено 8 предметов для прохождения тестирования.
* Варианты тестов за 2012, 2014, 2016 год.
* Использование предоставляются бесплатно.
* Для прохождения тестирования авторизация не обязательна.
* После выполнения теста можно посмотреть правильные ответы для каждого задания.

Недостатки:

* Результат представляется в процентах.
* Условие вопроса в каждом задании теста представлено фотографией (плохое качество).
* Низкая скорость работы программы.
* Неудобный пользовательский интерфейс.

**Testirovanie.org –** сайт централизованного тестирования. Содержит онлайн-тесты для подготовки к ЦТ. Предоставляет возможность пройти тестирование на Testy.by.

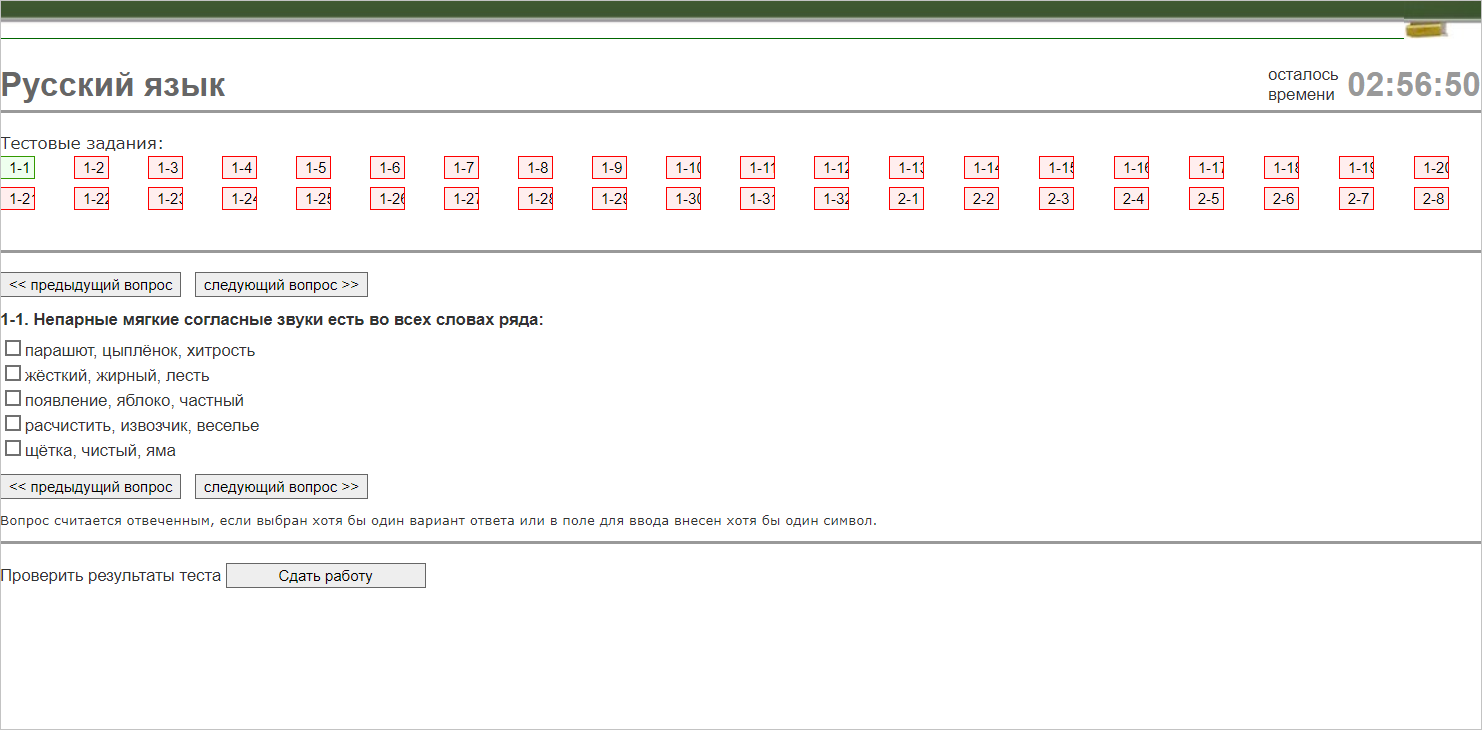


Рисунок 1.4 – Скриншот Testirovanie.org.

Достоинства:

* Представлено 11 предметов для прохождения тестирования.
* Использование предоставляются бесплатно.
* Показывает статистику.
* Для прохождения тестирования авторизация не обязательна.
* Итоговый результат в баллах.

Недостатки:

* Балл выставляется по усредненным коэффициентам.
* Низкая скорость работы программы.
* Неудобный пользовательский интерфейс.

## Постановка задачи

Программное средство должно включать в себя несколько основных предметов тестирования. Для русского и белорусского языка предусмотреть задания с несколькими вариантами ответов в части А. Подсчет итогового тестового бала необходимо выполнять с помощью весовых коэффициентов сложности каждого задания. Данная программа должна самостоятельно формировать бланки абитуриентов в заданном количестве. Для создания пользовательского бланка сделать выбор: сформировать программно или загрузить готовый. Ответы на задания для соответствующих предметов могут быть представлены в виде чисел или слов. Все данные о бланках, коэффициентах, ответах, количестве правильно выполненных заданий следует сохранить в файлы для возможности их просмотра и анализа.

Для разработки программного средства ЦТ необходимо реализовать следующие функции:

* Программное формирование ответов для всех заданий теста.
* Создание бланков абитуриентов.
* Подсчет количества абитуриентов, правильно выполнивших задание.
* Определение весовых коэффициентов заданий.
* Сохранение всех данных в текстовый файл.
* Загрузка данных из текстового файла.
* Определение балла за задание с несколькими вариантами ответов.
* Автоматическая проверка результатов.
* Сохранение истории прохождения теста в файл.

Я выбрал в качестве языка программирования язык Delphi потому, что в ходе выполнения лабораторных работ я хорошо его освоил.

# **АНАЛИЗ ТРЕБОВАНИЙ К ПРОГРАММНОМУ СРЕДСТВУ И РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ ТРЕБОВАНИЙ**

## 2.1. Описание функциональности ПС.

Программное средство должно предоставить пользователю список предметов централизованного тестирования поддерживаемых данной программой.

Предметы:

* Математика
* Русский язык
* Белорусский язык
* Биология
* Физика
* Химия
* История Беларуси

Предметы имеют разное количество заданий как в части А теста, так и в части В.

Некоторые предметы можно объединить в группы по числу заданий:

* Математика и физика:
* 30 заданий (А часть – 18, В часть – 12).
* Русский язык и белорусский язык:
* 40 заданий (А часть – 30, В часть – 10).
* Химия, биология и история Беларуси:
* 50 заданий (А часть – 38, В часть – 12).

Для русского и белорусского языка в части А необходимо сделать два варианта ответа.

С помощью компонента PageControl пользователю предоставлено 3 панели-страницы (tabsheet) для работы с программой. Переход на другую панель осуществляется щелчком мыши в области заголовка страницы. Пользователь может перейти на любую страницу и с помощью соответствующих кнопок на главной странице программного средства.

Программа должна предоставить пользователю следующие действия:

* Выбор предмета тестирования.
* Задание числа абитуриентов, выполнивших тест.
* Выбор способа формирования бланка пользователя (программно или загрузить готовый).
* Возможность внесения изменений в бланк пользователя, если он хочет загрузить бланк для получения результата.
* Получение итогового тестового балла по нажатию мышью на соответствующую кнопку.
* Возможность посмотреть и проанализировать данные о бланке пользователя, бланках абитуриентов, коэффициентах и количестве правильных ответов за каждое задание. Бланк с ответами тоже должен быть доступен для просмотра.

В зависимости от того, какой был выбран предмет для тестирования, программа должна уметь:

* Сформировывать бланки абитуриентов для соответствующего предмета в заданном пользователем количестве.
* Сформировывать программно или загружать готовый бланк пользователя.
* Сохранять все бланки абитуриентов и пользователя.
* Автоматически определять весовые коэффициенты сложности каждого задания и сохранять их.
* Определять балл за задание с несколькими вариантами ответов.
* Подсчитывать количество абитуриентов, правильно выполнивших задание.
* Определять итоговый тестовый балл пользователя.

Программа должна сохранять в файл результаты прохождения тестирования (историю тестирования).

## 2.2. Спецификация функциональных требований.

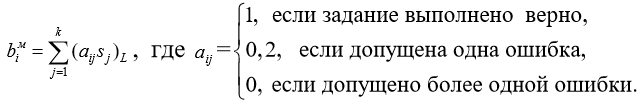
Одним из функциональных требований является «Создание бланков для абитуриентов».

Спецификация формирования бланков для абитуриентов по различным предметам:

* Математика и физика:
* А часть – в числовом виде (один вариант ответа).
* В часть – в числовом виде.
* Русский язык и белорусский язык:
* А часть – только в числовом виде (два варианта ответов)
* В часть – в виде слова, последние 4 задания в виде последовательности букв и цифр (пример - А1Б2В3Г4).
* Биология:
* А часть – в числовом виде (один вариант ответа).
* В часть:
* Ответы В1-В6 – в числовом виде.
* Ответы В7-В9 – слово.
* Ответы В10-В12– последовательность букв и цифр.
* Химия:
* А часть– в числовом виде (один вариант ответа).
* В часть:
* Ответы В1-В9 – в числовом виде.
* Ответы В10-В12 – последовательность букв и цифр.
* История Беларуси:
* А часть – в числовом виде (один вариант ответа).
* В часть:
* Ответы В1-В9 – слово.
* Ответы В10-В12 – последовательность букв и цифр.

Следующее функциональное требование «Определение балла за задание с несколькими вариантами ответов».

Особенность заключается в том, что в заданиях с несколькими вариантами ответов подсчет модифицированного балла i-го абитуриента производится следующим образом:



В случае, если дано более одного неверного ответа, а также дан хотя бы один неверный и одновременно не указан хотя бы один верный ответ, – задание признается выполненным неверно.

Во всех случаях ошибкой считается:

* Указание неверного варианта.
* Неуказанный верный ответ.

# **3. ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА**

В программе используется структура данных – запись. Запись я применяю для хранения информации о бланках с ответами и о таких данных тестирования, как коэффициенты, количество правильных ответов.

Запись состоит из двух полей: str – задание и Num – ответ или коэффициент.

В программном средстве используется однонаправленный динамический список, в который сохраняется история прошлых прохождений тестирования.

## **3.1. Разработка алгоритма генерации бланка**

В программе есть два способа формирования бланков с ответами: когда возможен один вариант ответа и, когда возможно два варианта ответа.

**1) Один вариант ответа:**

Ответы для заданий из части А формируются программно и могут принимать значение от 1 до 4: Random(4)+1. В части В ответы выбираются из соответствующих множеств, которые прописаны заранее. Используется тип записи, где поле Num – типа integer.

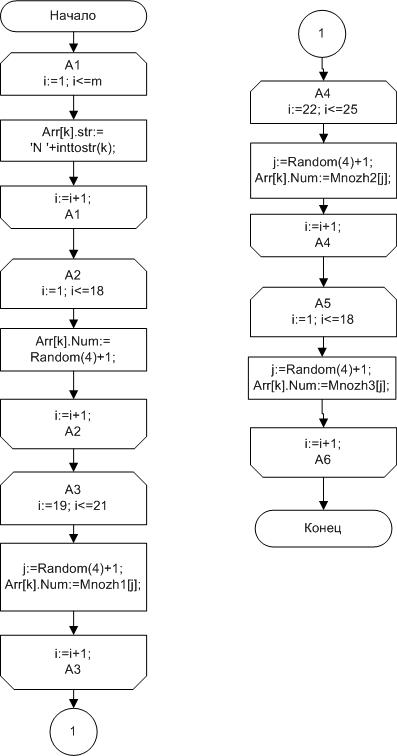


Рисунок 3.1 – схема алгоритма создания бланка с одним вариантом ответа.

**2) Два варианта ответа:**

В части А содержится два числа, которые формируются программно и записываются в строку, между числами пробел. Следовательно, получается строка с ответами, где первый символ строки – 1-ый вариант ответа, а третий символ строки – 2-ой вариант ответа. Используется тип записи, где поле Num – строкового типа. На рисунке 3.2 показан алгоритм формирования ответов части А для русского и белорусского языка.

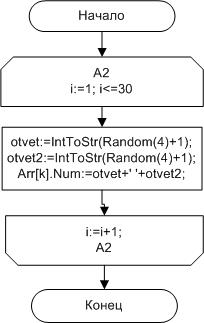


Рисунок 3.2 – схема алгоритма создания в бланке несколько вариантов ответа для заданий части А.

На рисунке 3.3 отображено, как будет выглядеть бланк с одним правильным ответом.

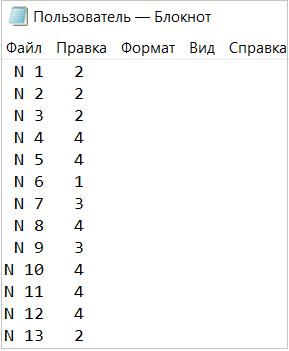


Рисунок 3.3 – скриншот бланка с одним вариантом ответов.

На рисунке 3.4 отображено, как будет выглядеть бланк с двумя правильными ответами.

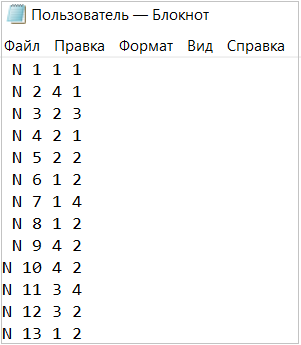


Рисунок 3.4 – скриншот бланка с двумя вариантами ответов.

## **3.2. Разработка алгоритма подсчета правильных ответов на задание**

Т.к. некоторые предметы имеют разное количество вариантов ответов в части А, то в программе должно быть две разновидности подсчета количества верных ответов на задание: когда возможен один вариант ответа и, когда возможно два варианта ответа.

**1) Для одного варианта ответа:**

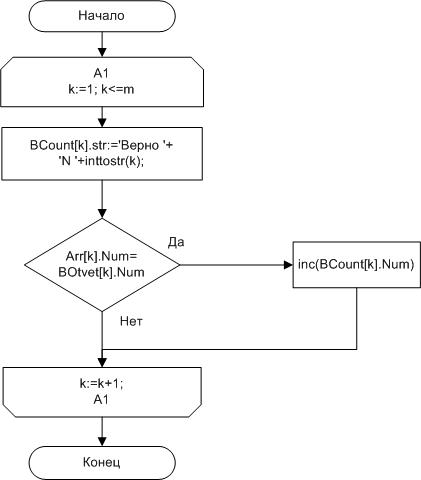


Рисунок 3.5 – схема алгоритма подсчета количества верных ответов.

**2) Для двух вариантов ответа:**

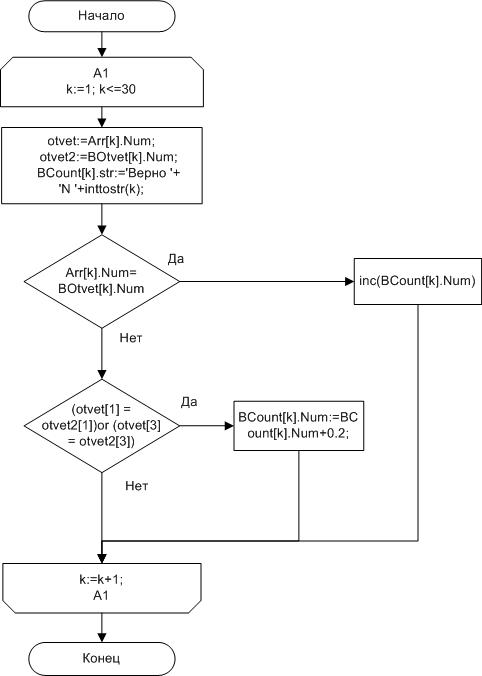
****

Рисунок 3.5 – схема алгоритма для подсчета верных ответов с двумя вариантами в части А.

В части-В подсчет количества верных ответов происходит по тому же алгоритму, что используется для заданий с одним вариантом ответа.

## **3.3. Разработка алгоритма вычисления весовых коэффициентов заданий**

Чтобы вычислить коэффициент сложности для каждого задания необходимо разделить количество правильных ответов (для данного задания) на количество абитуриентов, выполнивших это задание и результат вычесть из единицы.

При подсчете весовых коэффициентов учитывается следующее:

* Если коэффициент больше 0.9, то ему присваивается значение 0.9.
* Если коэффициент меньше 0.1, то ему присваивается значение 0.1.

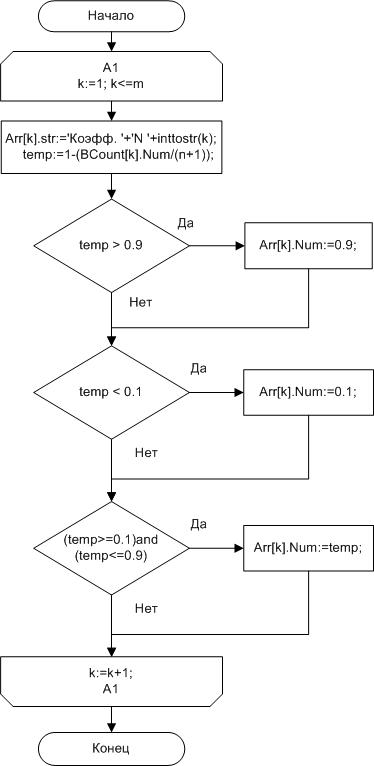


Рисунок 3.6 – схема алгоритма подсчета коэффициентов сложности для заданий с одним вариантом ответа.

## **3.4. Разработка алгоритма создания бланков абитуриентов**

Чтобы создать бланки для абитуриентов, количество которых задает пользователь (n) необходимо сделать цикл:

* Создание очередного бланка с ответами.
* Сохранение его в файл.
* Подсчет количества правильно решенных заданий.

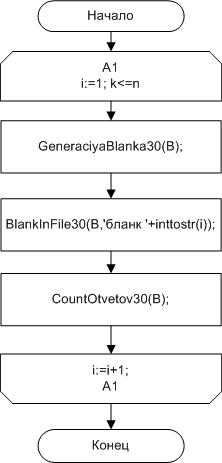


Рисунок 3.7 – схема алгоритма создания бланков абитуриентов.

## **3.5. Разработка алгоритма подсчета итогового балла**

Подсчет итогового балла тоже зависит от того, сколько возможно вариантов ответа. Поэтому выделяется два способа: подсчет итогового балла с одним вариантом ответов и с двумя вариантами.

**1) Подсчет итогового балла для тестов с одним вариантом ответов:**

Все коэффициенты сложности заданий складываются вместе (SummaCoeff).

Проверяется бланк с ответами пользователя, если задание правильно, то коэффициент данного задания складывается в сумму коэффициентов пользователя (Summa).

Итоговый балл будет равен частному от деления суммы коэффициентов пользователя на сумму всех коэффициентов, умноженному на 100.

Далее итоговый балл округляется по правилам математики.

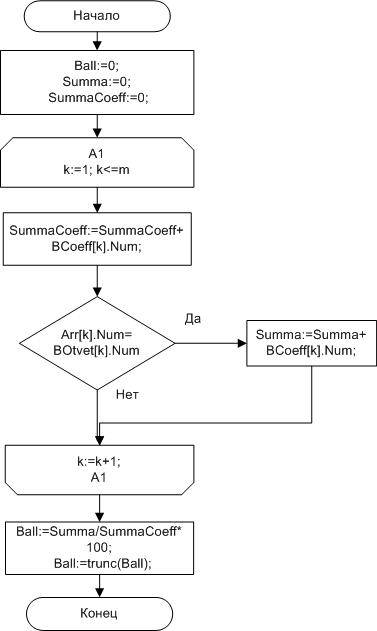


Рисунок 3.8 – схема алгоритма подсчета итогового балла для тестов с одним вариантом ответов.

**2) Подсчет итогового балла для тестов с двумя вариантами ответов:**

Этапы вычисления итогового балла для тестов с двумя вариантами ответов такие же, как и для тестов с одним вариантом ответа, кроме этапа с проверкой бланка с ответами пользователя.

Проверяется, сколько правильных ответов находится в бланке пользователя. Коэффициенты за задания уже ранее подсчитаны в программе.

Схема алгоритма изображена на рисунке 3.9.

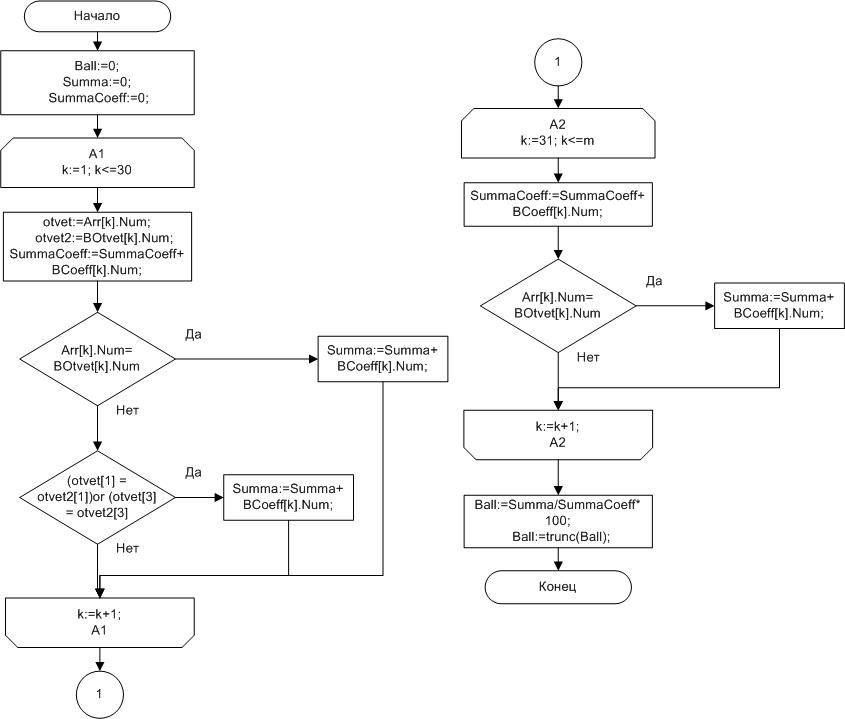


Рисунок 3.9 – схема алгоритма подсчета итогового балла для тестов с двумя вариантами ответов.

## **3.6. Динамический список**

Я использовал однонаправленный динамический список. В него сохраняется история прошлых прохождений тестирования по различным предметам.

Для реализации сохранения истории я выделил 3 процедуры:

* Создание списка
* Добавление в список
* Сохранение списка

При нажатии на кнопку «Итоговый результат» вызывается процедура добавления в динамический список. В список записывается предмет тестирования и итоговый балл.

После каждого завершения программы последняя история сохранится в папке «История тестирований» в файле, с соответствующей датой.

Процедуры находятся в модуле UnitResult.

После нажатия на кнопку «Завершить программу» и подтверждения выхода из программы вызывается процедура сохранения списка в файл. Т.е. результаты по всем предметам, по которым было проведено тестирование будут сохранены вместе. Об этом еще будет рассказано в руководстве по использованию.

# **4. СОЗДАНИЕ (КОНСТРУИРОВАНИЕ) ПРОГРАММНОГО СРЕДСТВА**

## 4.1. Формы в программе

В программном средстве используется одна форма – frmMain. Все основные действия, связанные с проведением тестирования, происходят на ней:

* Выбор предмета тестирования.
* Указание числа абитуриентов, выполнивших тест.
* Выбор способа загрузки бланка пользователя.
* Подсчет и отображение итогового балла.

Страница с проведением теста представлена на рисунке 4.1.

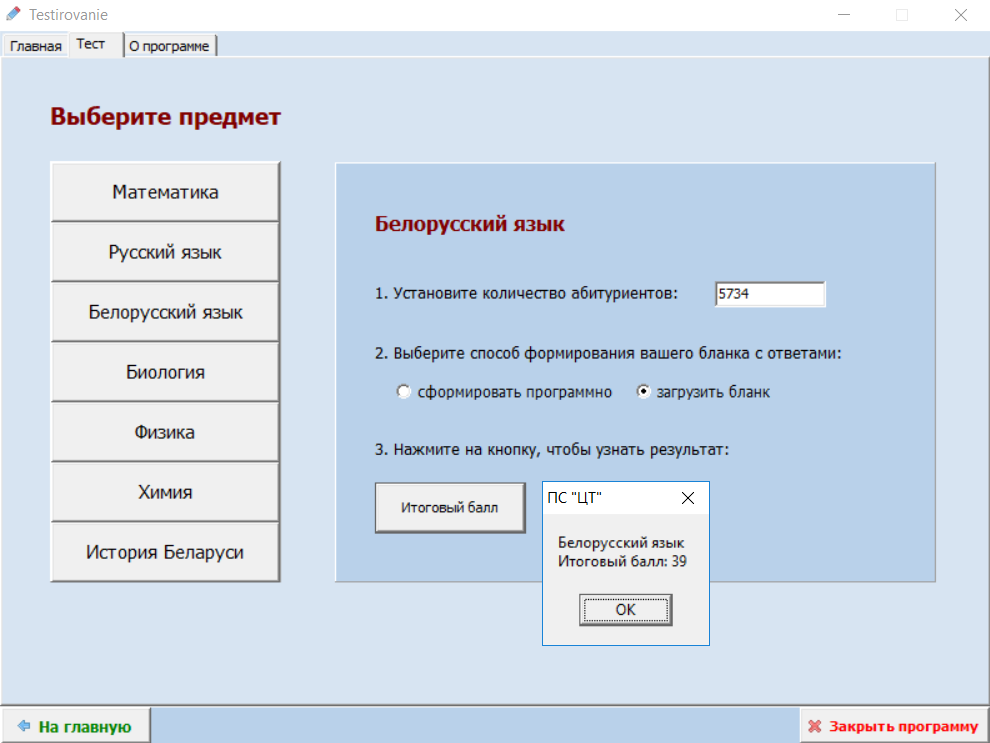


Рисунок 4.1 – скриншот страницы программы с проведением теста.

Схема алгоритма работы всей программы представлена в приложении 1. Текст программы находится в приложении 2.

## 4.2. Структура модулей программы

В процессе разработки программного средства я выделил следующие модули. Некоторые из модулей можно объединить отдельную в группу:

* frmMain – главный модуль программы.
* UnitResult – модуль с набором процедур для сохранения результатов тестирования.

Модули для предметов с разным количеством заданий:

* Unit30zad – модуль с набором процедур для предметов тестирования, состоящих из 30-ти заданий.
* Unit40zad – модуль с набором процедур для предметов тестирования, состоящих из 40-ка заданий.
* Unit50zad – модуль с набором процедур для предметов тестирования, состоящих из 50-ти заданий.

Отличие данных модулей Unit30zad, Unit40zad и Unit50zad в том, что процедуры формирования бланков ответов, подсчета коэффициентов и подсчета количества правильно выполненных заданий содержат разные алгоритмы для каждого модуля. Это связано с тем, что предметы тестирования:

* Имеют разное количество заданий: 30, 40, 50.
* Могут содержать несколько вариантов ответа в А части теста (для предметов, состоящих из 40-ка заданий).
* Имеют разные особенности для подсчета коэффициентов у заданий с несколькими вариантами ответов (для предметов, состоящих из 40-ка заданий).
* Содержат ответы, которые могут быть представлены как в виде числа, так и в виде слова (например, ответы для предметов с 30-ю заданиями представляются только в числовом виде).

## 4.3. Описание модуля MyUnit

Модуль MyUnit является юнитом главной формы. В нем находятся методы класса формы. В основном это обработчики событий.

Описание основных подпрограмм, которые содержатся в модуле MyUnit, приведено в таблице 4.1.

Таблица 4.1. – основные подпрограммы модуля Main

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Описание подпрограммы** | **Заголовок подпрограммы** | **Имя параметра** | **Назначение параметра** |
| TfrmMain.bitbtnStartClick | Обработка нажатия мыши на кнопку «Начать тестирование» | procedure TfrmMain.bitbtnStartClick(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| TfrmMain.bitbtnInstructionClick | Обработка нажатия мыши на кнопку «Прочесть инструкцию» | procedure  TfrmMain.bitbtnInstructionClick(Sender:  TObject); | Sender | Объект, который сгенерировал событие |
| TfrmMain.bitbtnHomeClick | Обработка нажатия мыши на кнопку «На главную» | procedure TfrmMain.bitbtnHomeClick(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| TfrmMain.bitbtnCloseClick | Обработка нажатия мыши на кнопку «Закрыть программу» | procedure TfrmMain.bitbtnCloseClick(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| TfrmMain.btnMatemClick | Обработка нажатия мыши на кнопку «Математика» | procedure TfrmMain.btnMatemClick(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| TfrmMain.btnRussYaClick | Обработка нажатия мыши на кнопку «Русский язык» | procedure TfrmMain.btnRussYaClick(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| TfrmMain.btnBelYaClick | Обработка нажатия мыши на кнопку «Белорусский язык» | procedure TfrmMain.btnBelYaClick(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| TfrmMain.btnBioClick | Обработка нажатия мыши на кнопку «Биология» | procedure TfrmMain.btnBioClick  (Sender: TObject); | Sender | Объект, который сгенерировал событие |
| TfrmMain.btnFizikaClick | Обработка нажатия мыши на кнопку «Физика» | procedure TfrmMain.btnFizikaClick(Sender: TObject) | Sender | Объект, который сгенерировал событие |
| TfrmMain.btnHimiyaClick | Обработка нажатия мыши на кнопку «Химия» | procedure TfrmMain.btnHimiyaClick(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| TfrmMain.btnIstorBelClick | Обработка нажатия мыши на кнопку «История Беларуси» | procedure TfrmMain.btnIstorBelClick(Sender: TObject); | Sender | Объект, который сгенерировал событие |

Таблица 4.1. – основные подпрограммы модуля Main

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TfrmMain.btnResultClick | Обработка нажатия мыши на кнопку «Итоговый балл» | procedure TfrmMain.btnResultClick(Sender: TObject); | Sender | Объект, который сгенерировал событие |

## 4.4. Описание модуля Unit30zad

Модуль Unit30zad является юнитом формы frmMain. Этот модуль предназначен для проведения тестирования предметов, состоящих из 30 заданий и имеющих по одному варианту ответа, представленного в виде числа (математика и физика). Описание основных подпрограмм, которые содержатся в модуле Unit30zad, приведено в таблице 4.2.

Таблица 4.2. – основные подпрограммы модуля Unit30zad

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Описание подпрограммы** | **Заголовок подпрограммы** | **Имя параметра** | **Назначение параметра** |
| GeneraciyaBlanka30 | Генерация бланка с ответами  (30 заданий) | procedure GeneraciyaBlanka30  (var Arr:TM); | Arr | Бланк с ответами на задания |
| BlankInFile30 | Сохранение бланка с ответами (30 заданий) в файл | procedure BlankInFile30(var Arr:TM; s:string); | Arr | Бланк с ответами на задания |
| BlankInFileReal  30 | Сохранение бланка с коэффициентами (30 заданий) в файл | procedure BlankInFileReal30(var Arr:TCoeff; s:string); | Arr | Бланк с коэффициентами |
| CountOtvetov30 | Подсчет количества верных ответов (30 зад.) | procedure CountOtvetov30(var Arr:TM); | Arr | Бланк с числом верных ответов |
| CoefficientZadaniya30 | Вычисление коэффициентов (30 заданий) | procedure CoefficientZadaniya30(var Arr:TCoeff); | Arr | Бланк с коэффициентами |
| VsegoBallov30 | Подсчет итогового балла за тест (30 зад.) | procedure VsegoBallov30(var Arr:TM); | Arr | Бланк пользователя |
| Abitur30 | Составление бланков (30 зад.) для абитуриентов | procedure Abitur30; |  |  |
| DownloadFile30 | Загрузка бланка пользователя (30 зад.) | procedure DownloadFile30(var Arr:TM); | Arr | Бланк пользователя |
| Ochistka30 | Очистка бланка с коэффиц. (30 зад.) | procedure Ochistka30(var Arr:TM); | Arr | Бланк с коэффиц. |
| Testirovanie30 | Тестирование по предмету (30 зад.) | procedure Testirovanie30; |  |  |

## 4.5. Описание модуля Unit40zad

Модуль Unit40zad является юнитом формы frmMain. Он предназначен для проведения тестирования предметов, состоящих из 40-ка заданий и имеющих по два варианта ответа в части А. Ответы в части В представлены в виде слова или последовательности букв и цифр. Подсчет модифицированного балла абитуриента происходит особым образом, о котором было рассказано ранее в спецификации функциональных требований. Описание основных подпрограмм, которые содержатся в модуле Unit40zad, приведено в таблице 4.3.

Таблица 4.3. – основные подпрограммы модуля Unit40zad

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Описание подпрограммы** | **Заголовок подпрограммы** | **Имя параметра** | **Назначение параметра** |
| GeneraciyaBlanka40 | Генерация бланка с ответами  (40 заданий) | procedure GeneraciyaBlanka40(var Arr:TM40); | Arr | Бланк с ответами на задания |
| BlankInFile40 | Сохранение бланка с ответами (40 заданий) в файл | procedure BlankInFile40(var Arr:TM40; s:string); | Arr | Бланк с ответами на задания |
| BlankInFileReal  40 | Сохранение бланка с коэффициентами (40 заданий) в файл | procedure BlankInFileReal40  (var Arr:TCoeff; s:string); | Arr | Бланк с коэффициентами |
| CountOtvetov40 | Подсчет количества верных ответов (40 зад.) | procedure CountOtvetov40(var Arr:TM40); | Arr | Бланк с числом верных ответов |
| CoefficientZadaniya40 | Вычисление коэффициентов  (40 заданий) | procedure CoefficientZadaniya40  (var Arr:TCoeff); | Arr | Бланк с коэффициентами |
| VsegoBallov40 | Подсчет итогового балла за тест  (40 зад.) | procedure VsegoBallov40(var Arr:TM40); | Arr | Бланк пользователя |
| Abitur40 | Составление бланков (40 зад.) для абитуриентов | procedure Abitur40; |  |  |
| DownloadFile40 | Загрузка бланка пользователя  (40 зад.) | procedure DownloadFile40(var Arr:TM40); | Arr | Бланк пользователя |
| Ochistka40 | Очистка бланка с коэффиц. (40 зад.) | procedure Ochistka40(var Arr:TCoeff); | Arr | Бланк с коэффиц. |
| Testirovanie40 | Тестирование по предмету (40 заданий) | procedure Testirovanie40; |  |  |

## 4.6. Описание модуля Unit50zad

Модуль Unit50zad является юнитом формы frmMain. Этот модуль предназначен для проведения тестирования предметов, состоящих из 50 заданий и имеющих по одному варианту ответа. Ответы в части В представлены в виде слова или последовательности букв и цифр. Описание основных подпрограмм, которые содержатся в модуле Unit50zad, приведено в таблице 4.4.

Таблица 4.4. – основные подпрограммы модуля Unit50zad

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Описание подпрограммы** | **Заголовок подпрограммы** | **Имя параметра** | **Назначение параметра** |
| GeneraciyaBlanka50 | Генерация бланка с ответами  (50 заданий) | procedure GeneraciyaBlanka50  (var Arr:TM50); | Arr | Бланк с ответами на задания |
| BlankInFile50 | Сохранение бланка с ответами (50 заданий) в файл | procedure BlankInFile50 (var Arr:TM50; s:string); | Arr | Бланк с ответами на задания |
| BlankInFileReal  50 | Сохранение бланка с коэффициентами (50 заданий) в файл | procedure BlankInFileReal50 (var Arr:TCoeff; s:string); | Arr | Бланк с коэффициентами |
| CountOtvetov50 | Подсчет количества верных ответов (50 зад.) | procedure CountOtvetov50 (var Arr:TM50); | Arr | Бланк с числом верных ответов |
| CoefficientZadaniya50 | Вычисление коэффициентов (50 заданий) | procedure CoefficientZadaniya50 (var Arr:TCoeff); | Arr | Бланк с коэффициентами |
| VsegoBallov50 | Подсчет итогового балла за тест (50 зад.) | procedure VsegoBallov50 (var Arr:TM50); | Arr | Бланк пользователя |
| Abitur50 | Составление бланков (50 зад.) для абитуриентов | procedure Abitur50; |  |  |
| DownloadFile50 | Загрузка бланка пользователя (50 зад.) | procedure DownloadFile50 (var Arr:TM50); | Arr | Бланк пользователя |
| Ochistka50 | Очистка бланка с коэффиц. (50 зад.) | procedure Ochistka50 (var Arr:TM50); | Arr | Бланк с коэффиц. |
| Testirovanie50 | Тестирование по предмету (50 зад.) | procedure Testirovanie50; |  |  |

## 4.7. Описание модуля UnitResult

Модуль UnitResult является юнитом формы frmMain. Этот модуль предназначен для того, чтобы результаты за тестирование по разным предметам сохранялись в историю после закрытия программы. Описание основных подпрограмм, которые содержатся в модуле UnitResult, приведено в таблице 4.5.

Таблица 4.5. – основные подпрограммы модуля UnitResult

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Описание подпрограммы** | **Заголовок подпрограммы** | **Имя параметра** | **Назначение параметра** |
| CreateResult | Создает список | procedure CreateResult; |  |  |
| AddResult | Добавляет элемент в список | procedure AddResult; |  |  |
| SaveResult | Сохраняет список в файл. | procedure SaveResult; |  |  |

# **5. ТЕСТИРОВАНИЕ, ПРОВЕРКА РАБОТОСПОСОБНОСТИ И АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ**

## 5.1. Тестирование подсчета баллов

Таблица 5.1 –тестирование подсчета баллов.

|  |  |  |  |
| --- | --- | --- | --- |
| **Номер теста** | **Последовательность действий** | **Ожидаемый результат** | **Полученный результат** |
| 1 | 1) в качестве предмета выбрана физика.  2)в поле с количеством абитуриентов указано: 1.  3)в папке Download в файле Download30fiz.txt все числа заменить на 0.  4)способ формирования бланка пользователя: загрузить бланк.  5)нажатие на кнопку «Итоговый балл» | Итоговый балл равен 0.  Т.к. ответы на задания не должны быть равны нулю.  Это значит, что пользователь не выполнил ни одного правильного ответа. | Безымянный.png  Тест пройден! |
| 2 | 1)в качестве предмета выбрать математику.  2)в коде заменить программный способ формирования бланка с ответами, на загрузку из файла. Загрузить файл Download30mat.txt  3) в поле с количеством абитуриентов указать: 1  4)способ формирования бланка пользователя: загрузить бланк.  5)нажатие на кнопку «Итоговый балл» | Итоговый балл равен 0.  Потому что в бланк с правильными ответами и в бланк с ответами пользователя загружаются одинаковые значения. Это значит, что пользователь выполнил правильно все задания. | Безымянный.png  Тест пройден! |

Я протестировал подсчет баллов в критических случаях, когда: пользователь не дал ни одного правильного ответа и ответил на все задания правильно. Подсчет баллов в этих двух случаях был произведен правильно. Во всех остальных случаях итоговый балл будет зависеть от коэффициентов заданий, которые вычисляются автоматически и зависят от бланков абитуриентов, ответы в которых формирует программа (подробнее будет рассказано в тестировании вычисления коэффициентов).

## 5.2. Тестирование подсчета количества абитуриентов, правильно выполнивших задание

Таблица 5.2 –тестирование подсчета правильных ответов.

|  |  |  |
| --- | --- | --- |
| **Последовательность действий** | **Ожидаемый результат** | **Полученный результат** |
| 1) Пройти тестирование по предмету Химия, с количеством абитуриентов: 3.  2) В папке «Результаты(50зад)Химия» проанализировать бланки 3-х абитуриентов, бланк пользователя и бланк с количеством верных ответов.  3) Самостоятельно сравнить бланки по 1-му заданию и посчитать сколько правильных ответов на это задание. | После сравнения ответов первое задание получилось, что число верных ответов на первое задание равно 1, т.к. правильный ответ был только в бланке 1.  Следовательно, в бланке с количеством верных ответов для первого задания должна быть написана 1. | Безымянный.png  Безымянный.png  Безымянный.png  Безымянный.png  Безымянный.png  Безымянный.png  Тест пройден! |

Также было проведено тестирование и анализ бланков целиком для заданий с двумя вариантами ответов. Было создано два бланка абитуриентов и бланк пользователя. Все данные содержащиеся в бланках представлены на рисунке 5.1.

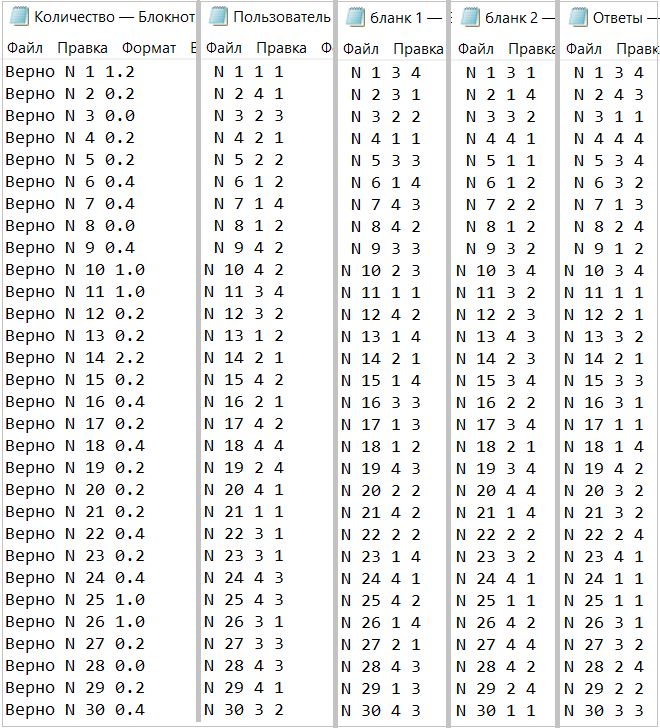


Рисунок 5.1 – скриншоты бланков после тестирования.

После анализа этих данных и учета того, как ставятся баллы, когда только один ответ из двух правильный, можно сказать, что количество правильно решенных заданий подсчитано правильно. Тест пройден.

## 5.3. Тестирование вычисления коэффициентов

Использую полученные данные с предыдущего тестирования (тестирование подсчета правильных ответов). В итоге у меня есть информация о количестве абитуриентов прошедших тестирование – 4 бланка (бланк пользователя + 3 бланка абитуриентов). Есть информация об ответе на первое задание для каждого бланка, о правильном ответе и количестве абитуриентов, которые правильно ответили на задание – 1 (первый бланк). Чтобы посчитать коэффициент первого задания необходимо разделить число абитуриентов, правильно выполнивших задание, на число абитуриентов, которые прошли тестирование и результат вычесть из единицы: 1 - ( ) = 0.75. Коэффициент для первого задания получился 0.75. Проверяю, как программа вычислила коэффициент (рисунок 5.2).

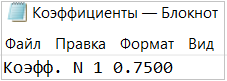


Рисунок 5.2 – скриншот коэффициента для первого задания.

Коэффициент, подсчитанный автоматически, равен коэффициенту, который я получил самостоятельно. Это означает, что данный тест пройден!

## 5.4. Тестирование производительности программы

Я провел тестирование производительности программного средства. Оно основано на тестировании быстроты выполнения программы. Скорость выполнения программы напрямую связана с количеством абитуриентов, выполнивших тест (рисунок 5.3).

Безымянный.png

Рисунок 5.3 – скриншот с панели тестирования.

После проведения тестирования было установлено, что программное средство работает быстро, если в поле количества абитуриентов введено четырехзначное число. При введении пятизначного числа, для выполнения тестирования программе понадобится несколько секунд. Для шестизначного числа времени понадобится еще больше. Время, отводимое на вычисление итогового балла за тест, прямо пропорционально количеству абитуриентов, заданному пользователем. Стоит отметить, что при введении девятизначного числа, программа не ломается, но время выполнения установить не удалось (медленно). Для быстрого выполнения программы рекомендуется вводить четырехзначное число.

## 5.5. Тестирование сохранения результатов в историю

Таблица 5.3 –тестирование сохранения истории.

|  |  |  |  |
| --- | --- | --- | --- |
| **Номер теста** | **Последовательность действий** | **Ожидаемый результат** | **Полученный результат** |
| 1 | 1) Пройти тестирование по предмету Химия.  2) Узнать итоговый балл.  3) Нажать на кнопку «Закрыть программу» и подтвердить действие. | Т.к. на экране отобразился итоговый балл равный 28, то  в папке «История тестирований» в файле 19.05.2018.txt (а тестирование я проводил именно этого числа) будет содержаться информация:  Химия 28 | Безымянный.png  Тест пройден! |
| 2 | 1) Пройти тестирование по предмету Физика и узнать итоговый балл за тест.  2) Пройти тестирование по предмету Русский язык и узнать итоговый балл за тест.  3) Пройти тестирование по предмету Математика и узнать итоговый балл за тест.  4) Пройти тестирование по предмету Белорусский язык и узнать итоговый балл за тест.  5) Нажать на кнопку «Закрыть программу» и подтвердить действие. | Т.к. итоговый балл по физике – 24, по русскому языку – 40,  по математике – 16, по белорусскому языку – 41, то  в папке «История тестирований» в файле 19.05.2018.txt будет содержаться информация:  Физика 24  Русский язык 40  Математика 16  Белорусский язык 41 | Безымянный.png  Тест пройден! |

## 5.6. Тестирование функционала окон и кнопок приложения

Таблица 5.4 –тестирование сохранения истории.

|  |  |  |  |
| --- | --- | --- | --- |
| **Номер теста** | **Последовательность действий** | **Ожидаемый результат** | **Полученный результат** |
| 1 | Нажатие на кнопку «Прочесть инструкцию» | Переход на страницу «О программе» | Тест пройден! |
| 2 | Нажатие на кнопку «Начать тестирование» | Переход на страницу «Тест» | Тест пройден! |
| 3 | Нажатие на кнопку «На главную» | Переход на страницу «Главная» | Тест пройден! |
| 4 | Нажатие на кнопку «Закрыть программу» | Закрытие программы и сохранение истории тестирования | Тест пройден! |
| 5 | Нажатие на одну из кнопок предметов | Открытие панели для прохождения теста выбранного предмета | Тест пройден! |
| 6 | Нажатие на кнопку «Итоговый балл» | Отображение итогового балла за тест | Тест пройден! |
| 7 | Поставить галочку на «сформировать программно» | Формирование бланка пользователя программным способом | Тест пройден! |
| 8 | Поставить галочку на «загрузить бланк» | Загрузка готового бланка пользователя | Тест пройден! |

## 5.7. Итоги тестирования

Программное средство прошло все этапы тестирования. Рекомендацией к использованию программы является введение в поле с количеством абитуриентов, выполнивших тест, четырехзначного числа. Программа корректно работает и с большими числами, но для вычисления итогового бала понадобится время.

# **6. РУКОВОДСТВО ПО УСТАНОВКЕ И ИСПОЛЬЗОВАНИЮ**

Чтобы начать использование программы, необходимо запустить файл MyProgramm.exe. После открытия программы появится окно, показанное на рисунке 6.1.

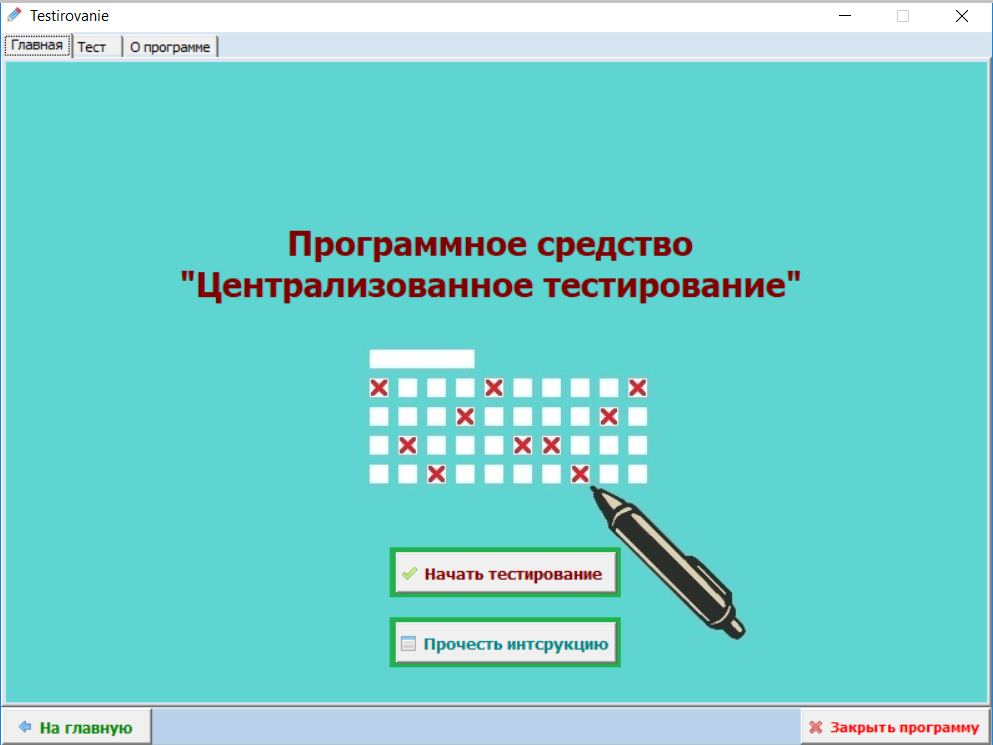


Рисунок 6.1 – главное окно ПС централизованного тестирования.

Программа состоит из 3-х страниц: «Главная», «Тест», «О программе». Работа с приложением начинается с главной страницы. Перед началом работы пользователь может ознакомиться с инструкцией к данному приложению, для этого необходимо нажать на кнопку «Прочесть инструкцию» (рисунок 6.2).

Безымянный.png

Рисунок 6.2 – кнопка для перехода к инструкции.

После нажатия на кнопку откроется страница «О программе», на которой находится сама инструкция, состоящая из 7 пунктов, где подробно рассказано о том, как пройти тестирование и узнать итоговый балл (рисунок 6.3).

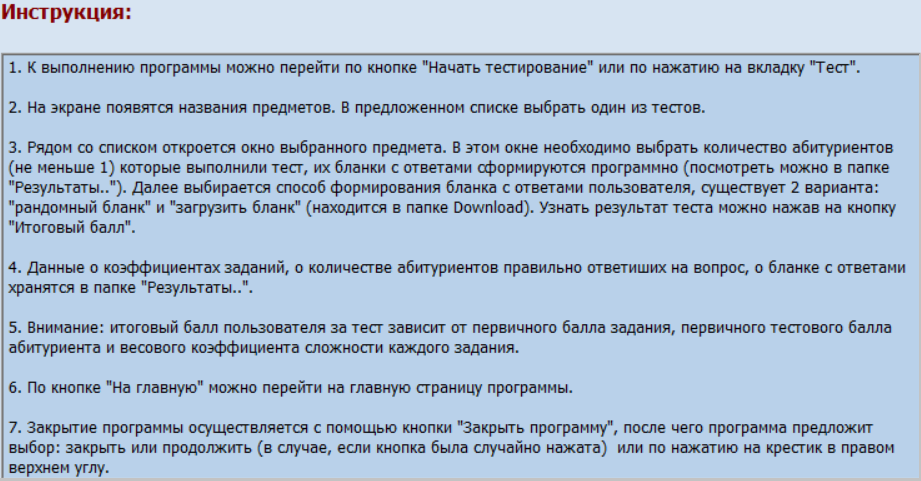


Рисунок 6.3 – инструкция в приложении.

В самом низу окна программы находится панель управления с кнопками: «На главную» и «Закрыть программу» (рисунок 6.4).

Безымянный.png

Рисунок 6.4 – панель управления.

По нажатию на кнопку «На главную» в любой момент времени можно переключиться на первую страницу приложения. Чтобы выйти из программы необходимо нажать на кнопку «Закрыть программу». Далее программа выведет предупреждение и предложит выбрать: закрыть или продолжить (рисунок 6.5). Данная опция была введена для того, чтобы, если кнопка была случайно, можно было отменить закрытие приложения.

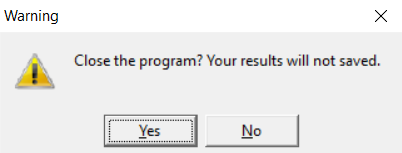


Рисунок 6.5 – подтверждение закрытия.

Находясь на главной странице программы можно перейти к самому тестированию. Необходимо нажать на кнопку «Начать тестирование» (рисунок 6.6).

Безымянный.png

Рисунок 6.6 – переход к тестированию.

После нажатия на кнопку откроется страница программы «Тест», на которой и осуществляется весь процесс тестирования. На экране появится список предметов для тестирования (рисунок 6.7).

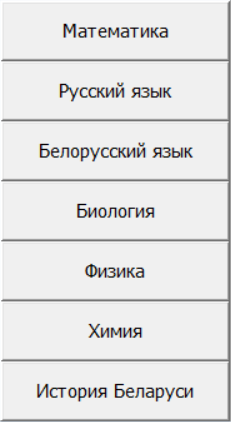


Рисунок 6.6 – список предметов для тестирования.

Когда будет выбран один из предложенных предметов, появится панель для дальнейшей работы (рисунок 6.7).

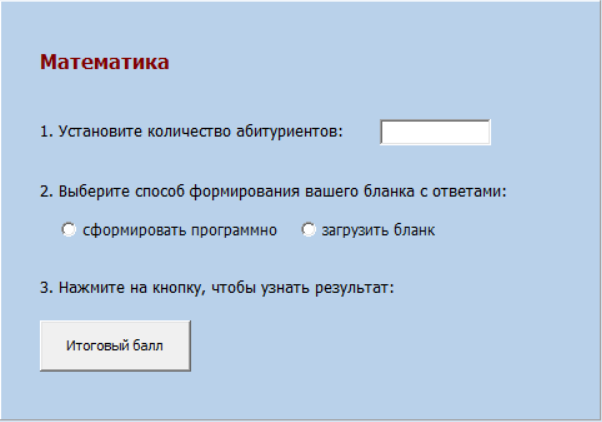


Рисунок 6.7 – панель выбранного предмета.

На этой панели отобразится название предмета (на рисунке: математика) и три обязательных пункта с информацией о том, что нужно сделать, чтобы получить результат.

Первый пункт: установить количество абитуриентов, выполнивших тестирование по данному предмету. Необходимо нажать на белую прямоугольную область рядом с первым пунктом и ввести с клавиатуры целое положительное число не меньшее, чем единица. Ограничения на максимальное число нет, но тестирование показало, что, чем больше число абитуриентов, тем медленнее программа будет анализировать данные и подсчитывать результат, поэтому для оптимальной скорости выполнения программы рекомендуется вводить четырехзначное число.

Второй пункт: выбрать способ формирования бланка пользователя. Бланк можно сформировать программно или загрузить готовый. В папке Download находятся бланки для загрузки, для каждого предмета свой бланк, в названии бланка числом указано количество заданий, после написано название предмета в сокращении (рисунок 6.8).

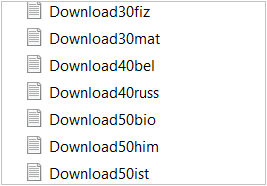


Рисунок 6.8 – бланки пользователя для загрузки.

Данные бланки перед началом загрузки пользователь может редактировать.

Третий пункт: нажать на кнопку «Итоговый балл», чтобы узнать результат. После программа автоматически подсчитает результат и выведет его (рисунок 6.9).

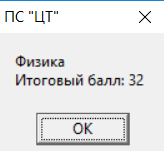


Рисунок 6.9 – итоговый балл по физике.

Также пользователь может перейти на любую страницу программы и с помощью соответствующих кнопок на главной странице приложения (рисунок 6.10).

Безымянный.png

Рисунок 6.10 – переключение между страницами.

После выполнения тестирования для каждого предмета можно посмотреть все данные о бланках и коэффициентах. Находятся они в соответствующих папках, которые показаны на рисунке 6.11.

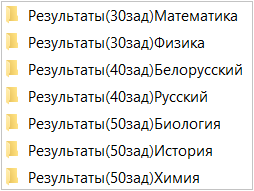


Рисунок 6.11 – папки с данными для каждого предмета.

После закрытия приложения программа предоставляет возможность пользователю просмотреть историю результатов тестирования. Все результаты хранятся в папке «История тестирования» в соответствующих текстовых документах, в названии которых указана дата прохождения тестирования (рисунок 6.12).



Рисунок 6.12 – текстовые документы с результатами.

В этих документах содержится вся история прохождения тестирования (рисунок 6.13).

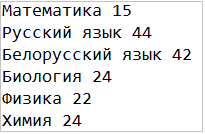


Рисунок 6.13 – скриншот из файла с историей результатов.

# **ЗАКЛЮЧЕНИЕ**

Результатом выполнения курсового проекта является программное средство централизованного тестирования. Программа имитирует процесс создания тестовых бланков и осуществляет автоматическую проверку результатов. Пользователю предоставляется в распоряжение 7 предметов тестирования.

Приложение выигрывает у аналогов тем, что использует не усредненные коэффициенты сложности для каждого задания, а рассчитывает их автоматически, опираясь на количество абитуриентов, выполнивших задание. Следствием является более точный итоговый балл. Такую же методику использует и Республиканский Институт Контроля Знаний.

Простой и удобный пользовательский интерфейс программы позволяет пользователю быстро разобраться с функционалом программы и приступить к тестированию. А наличие инструкции в приложении позволяет сделать этот процесс еще быстрее. Программа предоставляет возможность просмотра и анализа всех данных, полученных в ходе тестирования.

После прохождения тестирования и отладки программного средства не было обнаружено случаев некорректной работы программы, но были рекомендованы некоторые условия использования.

В будущем программное средство может быть улучшено. Например, добавлением в приложение условий для каждого вопроса и возможностью дать пользователю на них ответ прямо в программе, а также в данное приложение возможно добавление еще нескольких предметов для тестирования. И программное средство можно будет применять в образовательных целях. Например, для подготовки учащихся старших классов к централизованному тестированию.

В ходе выполнения курсового проекта я закрепил знания по работе с динамическими структурами данных, с текстовыми файлами. Получил знания и опыт в организации пользовательского интерфейса.

# **СПИСОК ЛИТЕРАТУРЫ**

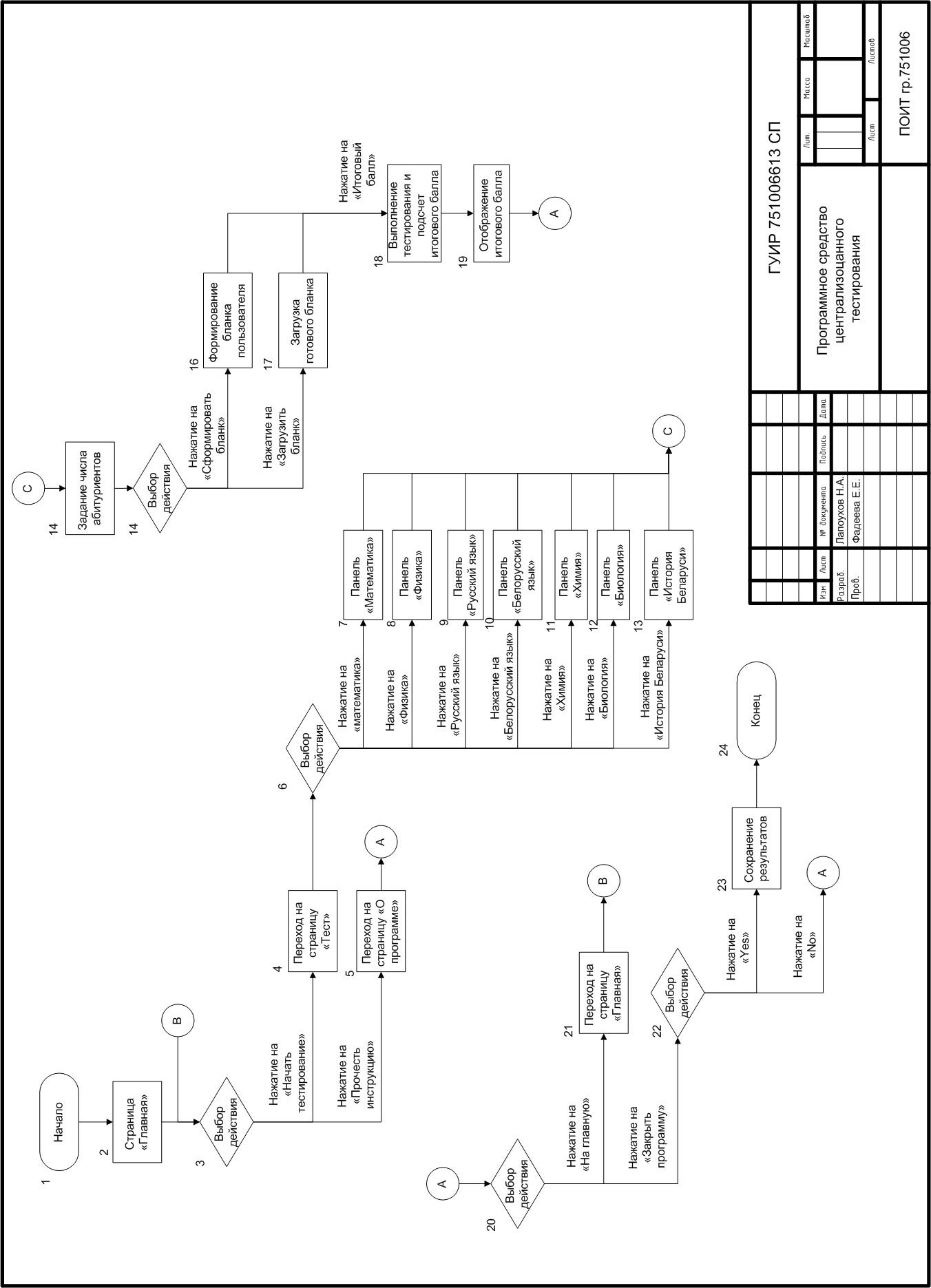
[1] Положение о порядке организации и проведения централизованного тестирования [Электронный ресурс]. Режим доступа: http://pravo.by/document/?guid=3871&p0=C20600714

[2] Правила подсчета тестового балла [Электронный ресурс]. Режим доступа: https://adukar.by/images/photo/Pravila\_podschyota\_testovogo\_balla.pdf

[3] Адукар [Электронный ресурс]. Режим доступа: https://adukar.by/news/bally-na-ct

[4] Глухова, Л.А. Основы алгоритмизации и программирования: учебное пособие. В 2 Ч. / Л.А. Глухова. – БГУИР, 2006 – Ч. 1. – 195 с.

# **ПРИЛОЖЕНИЕ 1**

**Схема алгоритма работы программы**

# **ПРИЛОЖЕНИЕ 2**

**Исходный код программы**

**Модуль главной формы:**

unit MyUnit;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, ComCtrls, Buttons, ExtCtrls, jpeg, Unit30zad, Unit40zad,

Unit50zad, UnitResult;

type

TfrmMain = class(TForm)

pagecontrolMain: TPageControl;

tabsheetHome: TTabSheet;

tabsheetTest: TTabSheet;

labelName: TLabel;

bitbtnClose: TBitBtn;

panelBottom: TPanel;

imgFonHome: TImage;

tabsheetDiscription: TTabSheet;

bitbtnStart: TBitBtn;

bitbtnHome: TBitBtn;

labelPredmet: TLabel;

panelPredmet: TPanel;

btnMatem: TButton;

btnRussYa: TButton;

btnBio: TButton;

btnBelYa: TButton;

btnFizika: TButton;

btnHimiya: TButton;

btnIstorBel: TButton;

panelTestirovanie: TPanel;

labelAbitur: TLabel;

editAbitur: TEdit;

labelUserBlank: TLabel;

radbtnRandom: TRadioButton;

radbtnDownload: TRadioButton;

labelPredmetName: TLabel;

btnResult: TButton;

labelResult: TLabel;

Image1: TImage;

bitbtnInstruction: TBitBtn;

memoInstruction: TMemo;

labelInstruction: TLabel;

procedure bitbtnInstructionClick(Sender: TObject);

procedure btnIstorBelClick(Sender: TObject);

procedure btnHimiyaClick(Sender: TObject);

procedure btnFizikaClick(Sender: TObject);

procedure btnBioClick(Sender: TObject);

procedure btnBelYaClick(Sender: TObject);

procedure btnRussYaClick(Sender: TObject);

procedure btnResultClick(Sender: TObject);

procedure btnMatemClick(Sender: TObject);

procedure bitbtnHomeClick(Sender: TObject);

procedure bitbtnStartClick(Sender: TObject);

procedure bitbtnCloseClick(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

frmMain: TfrmMain;

implementation

{$R \*.dfm}

procedure TfrmMain.bitbtnHomeClick(Sender: TObject);

begin

pagecontrolMain.ActivePage:=tabsheetHome;

end;

procedure TfrmMain.bitbtnInstructionClick(Sender: TObject);

begin

pagecontrolMain.ActivePage:=tabsheetDiscription;

end;

procedure TfrmMain.bitbtnStartClick(Sender: TObject);

begin

pagecontrolMain.ActivePage:=tabsheetTest;

CreateResult;

end;

procedure TfrmMain.bitbtnCloseClick(Sender: TObject);

Var Res: Integer;

begin

Res := MessageDlg('Close the program? Your results will not saved.', mtWarning, [mbYes, mbNo], 0);

if Res = mrYes then

begin

if (FloatToStr(Ball)) <> '0' then

SaveResult;

frmMain.Close;

end;

end;

procedure TfrmMain.btnMatemClick(Sender: TObject);

begin

panelTestirovanie.Show;

labelPredmetName.Caption:='Математика';

end;

procedure TfrmMain.btnRussYaClick(Sender: TObject);

begin

panelTestirovanie.Show;

labelPredmetName.Caption:='Русский язык';

end;

procedure TfrmMain.btnBelYaClick(Sender: TObject);

begin

panelTestirovanie.Show;

labelPredmetName.Caption:='Белорусский язык';

end;

procedure TfrmMain.btnBioClick(Sender: TObject);

begin

panelTestirovanie.Show;

labelPredmetName.Caption:='Биология';

end;

procedure TfrmMain.btnFizikaClick(Sender: TObject);

begin

panelTestirovanie.Show;

labelPredmetName.Caption:='Физика';

end;

procedure TfrmMain.btnHimiyaClick(Sender: TObject);

begin

panelTestirovanie.Show;

labelPredmetName.Caption:='Химия';

end;

procedure TfrmMain.btnIstorBelClick(Sender: TObject);

begin

panelTestirovanie.Show;

labelPredmetName.Caption:='История Беларуси';

end;

procedure TfrmMain.btnResultClick(Sender: TObject);

begin

n:=strtoint(editAbitur.Text);

if radbtnRandom.Checked = true then

begin

if labelPredmetName.Caption='Математика' then

begin

predm:='Математика';

GeneraciyaBlanka30(BUser);

BlankInFile30(BUser,'Пользователь');

Testirovanie30;

end;

if labelPredmetName.Caption='Физика' then

begin

predm:='Физика';

GeneraciyaBlanka30(BUser);

BlankInFile30(BUser,'Пользователь');

Testirovanie30;

end;

if labelPredmetName.Caption='Русский язык' then

begin

predm:='Русский язык';

GeneraciyaBlanka40(BUser40);

BlankInFile40(BUser40,'Пользователь');

Testirovanie40;

end;

if labelPredmetName.Caption='Белорусский язык' then

begin

predm:='Белорусский язык';

GeneraciyaBlanka40(BUser40);

BlankInFile40(BUser40,'Пользователь');

Testirovanie40;

end;

if labelPredmetName.Caption='Химия' then

begin

predm:='Химия';

GeneraciyaBlanka50(BUser50);

BlankInFile50(BUser50,'Пользователь');

Testirovanie50;

end;

if labelPredmetName.Caption='Биология' then

begin

predm:='Биология';

GeneraciyaBlanka50(BUser50);

BlankInFile50(BUser50,'Пользователь');

Testirovanie50;

end;

if labelPredmetName.Caption='История Беларуси' then

begin

predm:='История Беларуси';

GeneraciyaBlanka50(BUser50);

BlankInFile50(BUser50,'Пользователь');

Testirovanie50;

end;

end;

if radbtnDownload.Checked = true then

begin

if labelPredmetName.Caption='Математика' then

begin

predm:='Математика';

DownloadFile30(BUser);

Testirovanie30;

end;

if labelPredmetName.Caption='Физика' then

begin

predm:='Физика';

DownloadFile30(BUser);

Testirovanie30;

end;

if labelPredmetName.Caption='Русский язык' then

begin

predm:='Русский язык';

DownloadFile40(BUser40);

Testirovanie40;

end;

if labelPredmetName.Caption='Белорусский язык' then

begin

predm:='Белорусский язык';

DownloadFile40(BUser40);

Testirovanie40;

end;

if labelPredmetName.Caption='Химия' then

begin

predm:='Химия';

DownloadFile50(BUser50);

Testirovanie50;

end;

if labelPredmetName.Caption='Биология' then

begin

predm:='Биология';

DownloadFile50(BUser50);

Testirovanie50;

end;

if labelPredmetName.Caption='История Беларуси' then

begin

predm:='История Беларуси';

DownloadFile50(BUser50);

Testirovanie50;

end;

end;

ShowMessage(labelPredmetName.Caption+#13#10+'Итоговый балл: '+FloatToStr(Ball));

AddResult;

radbtnRandom.Checked:=False; radbtnDownload.Checked:=False;

editAbitur.Text:='';

panelTestirovanie.Hide;

end;

end.

**Модуль Unit30zad:**

unit Unit30zad;

interface

Uses

SysUtils, Windows;

const m=30;

p=18;

Type TRec=record

str:string[10];

Num:integer;

end;

TM=array [1..m] of TRec;

Var BUser:TM;

n,j:integer;

Ball,Summa,SummaCoeff,temp:real;

f:textfile;

predm:string;

procedure Testirovanie30;

procedure GeneraciyaBlanka30(var Arr:TM);

procedure BlankInFile30(var Arr:TM; s:string);

procedure DownloadFile30(var Arr:TM);

implementation

const Mnozh1:array[1..4]of integer=(35,27,13,23);

Mnozh2:array[1..4]of integer=(87,75,61,93);

Mnozh3:array[1..4]of integer=(144,109,221,195);

Type TCoeffRec=record

str:string[12];

Num:real;

end;

TCoeff=array [1..m] of TCoeffRec;

Var BOtvet,B,BCount:TM;

BCoeff:TCoeff;

i,k:integer;

procedure GeneraciyaBlanka30(var Arr:TM);

begin

for k:=1 to m do

Arr[k].str:='N '+inttostr(k);

for k:=1 to 18 do

Arr[k].Num:=Random(4)+1;

for k:=18 to 21 do

begin

j:=Random(4)+1;

Arr[k].Num:=Mnozh1[j];

end;

for k:=22 to 25 do

begin

j:=Random(4)+1;

Arr[k].Num:=Mnozh2[j];

end;

for k:=26 to m do

begin

j:=Random(4)+1;

Arr[k].Num:=Mnozh3[j];

end;

end;

procedure BlankInFile30(var Arr:TM; s:string);

begin

if predm='Математика' then

assignfile(f,'Результаты(30зад)Математика/'+s+'.txt');

if predm='Физика' then

assignfile(f,'Результаты(30зад)Физика/'+s+'.txt');

rewrite(f);

for k:=1 to m do

writeln(f,Arr[k].str:4,' ',Arr[k].Num:3);

closefile(f);

end;

procedure BlankInFileReal30(var Arr:TCoeff; s:string);

begin

if predm='Математика' then

assignfile(f,'Результаты(30зад)Математика/'+s+'.txt');

if predm='Физика' then

assignfile(f,'Результаты(30зад)Физика/'+s+'.txt');

rewrite(f);

for k:=1 to m do

writeln(f,Arr[k].str:4,' ',Arr[k].Num:3:4);

closefile(f);

end;

procedure CountOtvetov30(var Arr:TM);

begin

for k:=1 to m do

begin

BCount[k].str:='Верно '+'N '+inttostr(k);

if Arr[k].Num=BOtvet[k].Num then

inc(BCount[k].Num);

end;

end;

procedure CoefficientZadaniya30(var Arr:TCoeff);

begin

for k:=1 to m do

begin

Arr[k].str:='Коэфф. '+'N '+inttostr(k);

temp:=1-(BCount[k].Num/(n+1));

if temp > 0.9 then

Arr[k].Num:=0.9;

if temp < 0.1 then

Arr[k].Num:=0.1;

if (temp>=0.1)and(temp<=0.9) then

Arr[k].Num:=temp;

end;

end;

procedure VsegoBallov30(var Arr:TM);

begin

Ball:=0; Summa:=0; SummaCoeff:=0;

for k :=1 to m do

begin

SummaCoeff:=SummaCoeff+BCoeff[k].Num;

if Arr[k].Num=BOtvet[k].Num then

Summa:=Summa+BCoeff[k].Num;

end;

Ball:=Summa/SummaCoeff\*100;

Ball:=trunc(Ball);

end;

procedure Abitur30;

begin

for i:=1 to n do

begin

GeneraciyaBlanka30(B);

BlankInFile30(B,'бланк '+inttostr(i));

CountOtvetov30(B);

end;

CountOtvetov30(BUser);

end;

procedure DownloadFile30(var Arr:TM);

begin

if predm='Математика' then

assignfile(f,'Download/Download30mat.txt');

if predm='Физика' then

assignfile(f,'Download/Download30fiz.txt');

reset(f);

for k:=1 to m do

begin

readln(f,Arr[k].Num);

end;

closefile(f);

end;

procedure Ochistka30(var Arr:TM);

begin

for k:=1 to m do

Arr[k].Num:=0;

end;

procedure Testirovanie30;

begin

GeneraciyaBlanka30(BOtvet);

BlankInFile30(BOtvet,'Ответы');

Abitur30;

BlankInFile30(BCount,'Количество');

CoefficientZadaniya30(BCoeff);

BlankInFileReal30(BCoeff,'Коэффициенты');

VsegoBallov30(BUser);

Ochistka30(BCount);

end;

end.

**Модуль Unit40zad:**

unit Unit40zad;

interface

Uses

SysUtils, Windows, Unit30zad;

const m=40;

MnozhOtvetov:array[1..5]of string[10]=('А1Б2В3Г4','А4Б3В2Г1','А2Б3В1Г4','А3Б1В2Г4','А4Б1В2Г3');

Type TRec=record

str:string[10];

Num:string;

end;

TM40=array [1..m] of TRec;

Var BUser40:TM40;

otvet,otvet2:string;

procedure Testirovanie40;

procedure GeneraciyaBlanka40(var Arr:TM40);

procedure BlankInFile40(var Arr:TM40; s:string);

procedure DownloadFile40(var Arr:TM40);

implementation

const MnozhRuss1:array[1..4]of string[10]=('наречие','частица','предлог','союз');

MnozhRuss2:array[1..5]of string[10]=('сделать','книга','земля','радуга','красный');

MnozhBel1:array[1..4]of string[10]=('злучнік','прыслоўе','дзеяслоў','лічэбнік');

MnozhBel2:array[1..5]of string[10]=('ролі','надаецца','якія','дзеянне','прадмет');

Type TCoeffRec=record

str:string[12];

Num:real;

end;

TCoeff=array [1..m] of TCoeffRec;

Var BOtvet,B:TM40;

BCoeff,BCount:TCoeff;

i,k:integer;

procedure GeneraciyaBlanka40(var Arr:TM40);

begin

for k:=1 to m do

Arr[k].str:='N '+inttostr(k);

for k:=1 to 30 do

begin

otvet:=IntToStr(Random(4)+1);

otvet2:=IntToStr(Random(4)+1);

Arr[k].Num:=otvet+' '+otvet2;

end;

for k:=31 to 33 do

begin

j:=Random(4)+1;

if predm='Русский язык' then

Arr[k].Num:=MnozhRuss1[j];

if predm='Белорусский язык' then

Arr[k].Num:=MnozhBel1[j];

end;

for k:=34 to 36 do

begin

j:=Random(5)+1;

if predm='Русский язык' then

Arr[k].Num:=MnozhRuss2[j];

if predm='Белорусский язык' then

Arr[k].Num:=MnozhBel2[j];

end;

for k:=37 to m do

begin

j:=Random(5)+1;

Arr[k].Num:=MnozhOtvetov[j];

end;

end;

procedure BlankInFile40(var Arr:TM40; s:string);

begin

if predm='Русский язык' then

assignfile(f,'Результаты(40зад)Русский/'+s+'.txt');

if predm='Белорусский язык' then

assignfile(f,'Результаты(40зад)Белорусский/'+s+'.txt');

rewrite(f);

for k:=1 to m do

writeln(f,Arr[k].str:4,' ',Arr[k].Num:3);

closefile(f);

end;

procedure BlankInFileReal40(var Arr:TCoeff; s:string);

begin

if predm='Русский язык' then

assignfile(f,'Результаты(40зад)Русский/'+s+'.txt');

if predm='Белорусский язык' then

assignfile(f,'Результаты(40зад)Белорусский/'+s+'.txt');

rewrite(f);

for k:=1 to m do

if s='Количество' then

writeln(f,Arr[k].str:4,' ',Arr[k].Num:3:1)

else

writeln(f,Arr[k].str:4,' ',Arr[k].Num:3:4);

closefile(f);

end;

procedure CountOtvetov40(var Arr:TM40);

begin

for k:=1 to 30 do

begin

otvet:=Arr[k].Num;

otvet2:=BOtvet[k].Num;

BCount[k].str:='Верно '+'N '+inttostr(k);

if Arr[k].Num = BOtvet[k].Num then

BCount[k].Num:=BCount[k].Num+1

else

if (otvet[1] = otvet2[1])or(otvet[3] = otvet2[3])then

BCount[k].Num:=BCount[k].Num+0.2;

end;

for k:=31 to m do

begin

BCount[k].str:='Верно '+'N '+inttostr(k);

if Arr[k].Num = BOtvet[k].Num then

BCount[k].Num:=BCount[k].Num+1;

end;

end;

procedure CoefficientZadaniya40(var Arr:TCoeff);

begin

for k:=1 to m do

begin

Arr[k].str:='Коэфф. '+'N '+inttostr(k);

temp:=1-(BCount[k].Num/(n+1));

if temp > 0.9 then

Arr[k].Num:=0.9;

if temp < 0.1 then

Arr[k].Num:=0.1;

if (temp>=0.1)and(temp<=0.9) then

Arr[k].Num:=temp;

end;

end;

procedure VsegoBallov40(var Arr:TM40);

begin

Ball:=0; Summa:=0; SummaCoeff:=0;

for k :=1 to 30 do

begin

otvet:=Arr[k].Num;

otvet2:=BOtvet[k].Num;

SummaCoeff:=SummaCoeff+BCoeff[k].Num;

if Arr[k].Num=BOtvet[k].Num then

Summa:=Summa+BCoeff[k].Num

else

if(otvet[1] = otvet2[1])or(otvet[3] = otvet2[3])then

Summa:=Summa+BCoeff[k].Num;

end;

for k :=31 to m do

begin

SummaCoeff:=SummaCoeff+BCoeff[k].Num;

if Arr[k].Num=BOtvet[k].Num then

Summa:=Summa+BCoeff[k].Num;

end;

Ball:=Summa/SummaCoeff\*100;

Ball:=trunc(Ball);

end;

procedure Abitur40;

begin

for i:=1 to n do

begin

GeneraciyaBlanka40(B);

BlankInFile40(B,'бланк '+inttostr(i));

CountOtvetov40(B);

end;

CountOtvetov40(BUser40);

end;

procedure DownloadFile40(var Arr:TM40);

begin

if predm='Русский язык' then

assignfile(f,'Download/Download40russ.txt');

if predm='Белорусский язык' then

assignfile(f,'Download/Download40bel.txt');

reset(f);

for k:=1 to m do

begin

readln(f,Arr[k].Num);

end;

closefile(f);

end;

procedure Ochistka40(var Arr:TCoeff);

begin

for k:=1 to m do

Arr[k].Num:=0;

end;

procedure Testirovanie40;

begin

GeneraciyaBlanka40(BOtvet);

BlankInFile40(BOtvet,'Ответы');

Abitur40;

BlankInFileReal40(BCount,'Количество');

CoefficientZadaniya40(BCoeff);

BlankInFileReal40(BCoeff,'Коэффициенты');

VsegoBallov40(BUser40);

Ochistka40(BCount);

end;

end.

**Модуль Unit50zad:**

unit Unit50zad;

interface

Uses

SysUtils, Windows, Unit30zad, Unit40zad;

const m=50;

Type TRec=record

str:string[10];

Num:string;

end;

TM50=array [1..m] of TRec;

Var BUser50:TM50;

procedure Testirovanie50;

procedure GeneraciyaBlanka50(var Arr:TM50);

procedure BlankInFile50(var Arr:TM50; s:string);

procedure DownloadFile50(var Arr:TM50);

implementation

const MnozhHim1:array[1..4]of string[2]=('15','44','23','11');

MnozhHim2:array[1..4]of string[2]=('62','71','49','27');

MnozhHim3:array[1..4]of string[3]=('151','113','120','132');

MnozhBio1:array[1..4]of string[2]=('47','3','78','56');

MnozhBio2:array[1..4]of string[3]=('123','108','135','150');

MnozhBio3:array[1..4]of string[10]=('ядро','оболочка','мембрана','цитоскелет');

MnozhIst1:array[1..4]of string[10]=('Полацкі','Маскоўскі','Кіеўскі','Сафійскі');

MnozhIst2:array[1..4]of string[10]=('каралём','літаратар','педагог','друкар');

MnozhIst3:array[1..4]of string[10]=('Крывічы','Дрыгавічы','Радзімічы','Паляне');

Type TCoeffRec=record

str:string[12];

Num:real;

end;

TCoeff=array [1..m] of TCoeffRec;

Var BOtvet,B:TM50;

BCoeff,BCount:TCoeff;

i,k:integer;

procedure GeneraciyaBlanka50(var Arr:TM50);

begin

for k:=1 to m do

Arr[k].str:='N '+inttostr(k);

for k:=1 to 38 do

begin

otvet:=IntToStr(Random(4)+1);

Arr[k].Num:=otvet;

end;

for k:=39 to 41 do

begin

j:=Random(4)+1;

if predm='Химия' then

Arr[k].Num:=MnozhHim1[j];

if predm='Биология' then

Arr[k].Num:=MnozhBio1[j];

if predm='История Беларуси' then

Arr[k].Num:=MnozhIst1[j];

end;

for k:=42 to 44 do

begin

j:=Random(4)+1;

if predm='Химия' then

Arr[k].Num:=MnozhHim2[j];

if predm='Биология' then

Arr[k].Num:=MnozhBio2[j];

if predm='История Беларуси' then

Arr[k].Num:=MnozhIst2[j];

end;

for k:=45 to 47 do

begin

j:=Random(4)+1;

if predm='Химия' then

Arr[k].Num:=MnozhHim3[j];

if predm='Биология' then

Arr[k].Num:=MnozhBio3[j];

if predm='История Беларуси' then

Arr[k].Num:=MnozhIst3[j];

end;

for k:=48 to m do

begin

j:=Random(5)+1;

Arr[k].Num:=MnozhOtvetov[j];

end;

end;

procedure BlankInFile50(var Arr:TM50; s:string);

begin

if predm='Химия' then

assignfile(f,'Результаты(50зад)Химия/'+s+'.txt');

if predm='Биология' then

assignfile(f,'Результаты(50зад)Биология/'+s+'.txt');

if predm='История Беларуси' then

assignfile(f,'Результаты(50зад)История/'+s+'.txt');

rewrite(f);

for k:=1 to m do

writeln(f,Arr[k].str:4,' ',Arr[k].Num:3);

closefile(f);

end;

procedure BlankInFileReal50(var Arr:TCoeff; s:string);

begin

if predm='Химия' then

assignfile(f,'Результаты(50зад)Химия/'+s+'.txt');

if predm='Биология' then

assignfile(f,'Результаты(50зад)Биология/'+s+'.txt');

if predm='История Беларуси' then

assignfile(f,'Результаты(50зад)История/'+s+'.txt');

rewrite(f);

for k:=1 to m do

if s='Количество' then

writeln(f,Arr[k].str:4,' ',Arr[k].Num:3:0)

else

writeln(f,Arr[k].str:4,' ',Arr[k].Num:3:4);

closefile(f);

end;

procedure CountOtvetov50(var Arr:TM50);

begin

for k:=1 to m do

begin

BCount[k].str:='Верно '+'N '+inttostr(k);

if Arr[k].Num = BOtvet[k].Num then

BCount[k].Num:=BCount[k].Num+1;

end;

end;

procedure CoefficientZadaniya50(var Arr:TCoeff);

begin

for k:=1 to m do

begin

Arr[k].str:='Коэфф. '+'N '+inttostr(k);

temp:=1-(BCount[k].Num/(n+1));

if temp > 0.9 then

Arr[k].Num:=0.9;

if temp < 0.1 then

Arr[k].Num:=0.1;

if (temp>=0.1)and(temp<=0.9) then

Arr[k].Num:=temp;

end;

end;

procedure VsegoBallov50(var Arr:TM50);

begin

Ball:=0; Summa:=0; SummaCoeff:=0;

for k :=1 to m do

begin

SummaCoeff:=SummaCoeff+BCoeff[k].Num;

if Arr[k].Num=BOtvet[k].Num then

Summa:=Summa+BCoeff[k].Num;

end;

Ball:=Summa/SummaCoeff\*100;

Ball:=trunc(Ball);

end;

procedure Abitur50;

begin

for i:=1 to n do

begin

GeneraciyaBlanka50(B);

BlankInFile50(B,'бланк '+inttostr(i));

CountOtvetov50(B);

end;

CountOtvetov50(BUser50);

end;

procedure DownloadFile50(var Arr:TM50);

begin

if predm='Химия' then

assignfile(f,'Download/Download50him.txt');

if predm='Биология' then

assignfile(f,'Download/Download50bio.txt');

if predm='История Беларуси' then

assignfile(f,'Download/Download50ist.txt');

reset(f);

for k:=1 to m do

begin

readln(f,Arr[k].Num);

end;

closefile(f);

end;

procedure Ochistka50(var Arr:TCoeff);

begin

for k:=1 to m do

Arr[k].Num:=0;

end;

procedure Testirovanie50;

begin

GeneraciyaBlanka50(BOtvet);

BlankInFile50(BOtvet,'Ответы');

Abitur50;

BlankInFileReal50(BCount,'Количество');

CoefficientZadaniya50(BCoeff);

BlankInFileReal50(BCoeff,'Коэффициенты');

VsegoBallov50(BUser50);

Ochistka50(BCount);

end;

end.

**Модуль UnitResult:**

unit UnitResult;

interface

Uses

SysUtils, Windows, Unit30zad;

procedure CreateResult;

procedure AddResult;

procedure SaveResult;

implementation

Type

TZap = record

Name:string;

ResultBall:string;

end;

TSP = ^ TSS;

TSS = record

INF:TZAP;

ADR:TSP;

end;

Var head,temp:TSP;

fileResult:textfile;

str:string;

procedure CreateResult;

begin

New(head);

head^.ADR:=nil;

temp:=head;

end;

procedure AddResult;

begin

New(temp^.adr);

temp:=temp^.adr;

temp^.adr:=nil;

with temp^.INF do

begin

Name:=predm;

ResultBall:=FloatToStr(Ball);

end;

end;

procedure SaveResult;

begin

str:=DateToStr(Date);

assignfile(fileResult,'История тестирований/'+str+'.txt');

rewrite(fileResult);

temp:=head^.adr;

while temp<>nil do

begin

with temp^.INF do

begin

writeln(fileResult,Name,' ',ResultBall);

end;

temp:=temp^.adr;

end;

closefile(fileResult);

end;

end.