МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ «ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНО-ЭКОНОМИЧЕСКИЙ КОЛЛЕДЖ»

Специальность 2-40 01 01 «Программное обеспечение

информационных технологий»

**КУРСОВОЙ ПРОЕКТ**

по дисциплине: «Основы алгоритмизации и программирования»

**на тему: «****Разработка обучающего приложения «Физика для старшей школы»»**

Пояснительная записка

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Разработал |  |  |  |  | Батюк Е. А. |
|  | (дата) |  | (подпись) |  | (ФИО) |
| Руководитель |  |  |  |  | Глушенок А. В. |
|  | (дата) |  | (подпись) |  | (ФИО) |

Гомель, 2022

СОДЕРЖАНИЕ

[Введение 3](#_Toc101636964)

[Простые типы данных **Ошибка! Закладка не определена.**](#_Toc101636965)

[ОПИСАНИЕ АЛГОРИТМОВ И ТЕХНОЛОГИИ РЕШЕНИЯ 9](#_Toc106637723)

[2.1 Описание требований и исходных данных 9](#_Toc106637724)

[2.2 Разработка интерфейса приложения 9](#_Toc106637726)

[3 Практическая часть выполнения задачи 13](#_Toc106637727)

[3.1 Интерфейс приложения 13](#_Toc106637728)

[3.2 Программная реализация 19](#_Toc106637729)

[4 результаты программы 20](#_Toc106637730)

# Введение

Цель курсового проекта – разработка мобильного приложения «».

Мобильное приложение

Приложение «Физика для старшей школы» включает в себя повторение пройденного материала, также изучение нового.

В основном, учебными пособиями по предмету пользуются как справочным материалом для подготовки домашнего задания и к проверочным работам. Разрабатываемая система даёт возможность не затрачивать дополнительного времени на поиск нужной информации и проводить самоконтроль и выявление слабых мест в изученном материале.

Задачи:

1. Создание краткого справочного материала по предмету «Физика»;
2. создание тестов самоконтроля.

Этапы создания системы:

1. Изучение учебного пособия, на основе которых создается справочный материал;
2. составление тестов самоконтроля по определённым темам;
3. изучение методов и алгоритмов языка программирования «Dart» в области создания мобильных приложений;
4. создание страницы поиска нужного материала;
5. разработка методов показа информации;
6. разработка методов прохождения тестов самоконтроля;
7. разработка интерфейса.

Возможности приложения:

* Просматривать справочный материал по темам;
* проходить тесты самоконтроля по темам;
* просматривать статистику прохождения тестов;
* просматривать статистику исходя из результатов пройденных тестов.

По итогу разработки системы будет получено приложение, которое несёт в себе краткую выдержку из тем, находящиеся в учебном пособии, тесты самоконтроля, формулы и таблицы.

# Простые типы данных

Тип данных — множество значений и операций над этими значениями.

Типизация — операция назначения типа информационным сущностям.

В языке Object Pascal существуют следующие простые типы данных:

* целочисленный;
* вещественный;
* символьный;
* строковый;
* логический.

Целочисленные типы содержат в себе значения, интерпретируемые как числа (знаковые и беззнаковые).[6]

В таблице 1.1. представлен диапазон значений целочисленных типов.

Таблица 1.1 – диапазон значений целочисленных переменных

|  |  |  |
| --- | --- | --- |
| Тип | Диапазон | Формат |
| shortint | -128 ..+127 | 8 битов со знаком |
| integer | -32768 .. 32767 | 16 битов со знаком |
| longint | -2147483648 +2147483647 | 32 бита со знаком |
| byte | 0 .. 255 | 8 битов без знака |
| word | 0 .. 65535 | 16 битов без знака |

На рисунке 1.1. представлен пример описания целочисленной переменной типа «integer».

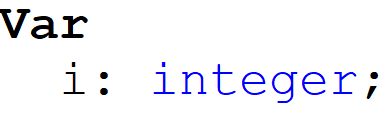


Рисунок 1.1 – Объявление переменной типа «Integer»

Над целочисленными переменными проводятся все математические операции, а также операции целочисленного деления.[1]

К вещественному типу относится подмножество вещественных чисел, которые могут быть представлены в формате с плавающей запятой с фиксированным числом цифр.

На рисунке 1.2. представлена таблица переменных вещественных типов.

Таблица 1.2 – Диапазон значений переменных вещественного типа

|  |  |  |
| --- | --- | --- |
| Тип | Диапазон | Цифры |
| Real  Single  Duble  Extende  comp | 2.9x10Е-39 до 1.7x10Е 38  1.5x10Е-45 до 3.4x10Е 38  5.0x10Е-324 до 1.7x10Е 308  3.4x10Е-493 до 1.1x10Е 403  -2Е 63 до 2Е 63 | от 11 до 12  от 7 до 8  от 15 до 16  от 19 до 20  от 19 до 20 |

На рисунке 1.2. представлен пример описания вещественной переменной типа «real».

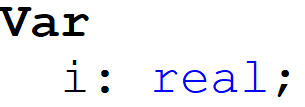


Рисунок 1.2 – Объявление переменной типа «Real»

Над вещественными переменными проводятся все математические операции, а также операции отбрасывания дробной части (TRUNC(x)) и округления (ROUND(x)).[2]

Символьный тип данных - тип данных, предназначенный для хранения одного символа (управляющего или печатного) в определённой кодировке.[3]

При описании символьной переменной используется зарезервированное слово «char», пример представлен на рисунке 1.3.

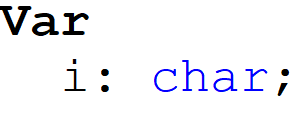


Рисунок 1.3 – Объявление переменной типа «Char»

Символы можно представить в виде числового кода.

ASCII — название таблицы, в которой некоторым распространённым печатным и непечатным символам сопоставлены числовые коды.

Числовой код символов представлен на рисунке 1.4.

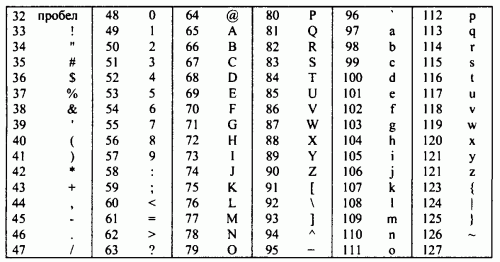


Рисунок 1.4 – Числовой код символов

Функции, которые применимы к символьным переменным:

* «ORD(X)» - определяет порядковый номер символа «Х»;
* CHR(X) - определяет символ по номеру;
* PRED(X) - выдает символ, стоящий перед символом Х;
* SUCC(X) - выдает символ, следующий после символа Х.

Строка– последовательность из определенного количества символов.[4]

Объявление типа «string» показаны на рисунке 1.5.

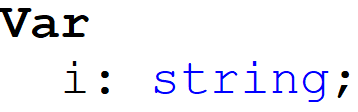


Рисунок 1.5 – Объявление переменной типа «string»

Процедуры обработки строк:

* Delete(St, Poz, N) – удаление N символов строки St, начиная с позиции Poz;
* Insert(Str1, Str2, Poz) – вставка строки Str1 в строку Str2, начиная с позиции Poz;
* Str(I, Srt) – преобразование числового значения величины I (целого или вещественного типа) и помещение результата в строку Str.
* Val(Str, I, Cod) – преобразует значение Str в величину целочисленного или вещественного типа и помещает результат в I. Функции обработки строк:
* Copy(Str, Poz, N) – выделяет из строки St подстроку длинной N символов, начиная с позиции Poz.
* Concat(Str1, Srt2, …, StrN) – выполняет сцепление строк Str1, Srt2, …, StrN в том порядке, в каком они указаны в списке параметров.
* Length(Str) – вычисляет длину в символах строки Str;
* Pos(Str1, Str2) – обнаруживает первое появление в строке Str2 подстроки Str1.

Логический тип данных – первоначальный тип данных в информатике, который принимают два значения переменной, определяемой истиной (true) и ложью (false).[5]

Объявление переменной типа «boolean» представлено на рисунке 1.6.

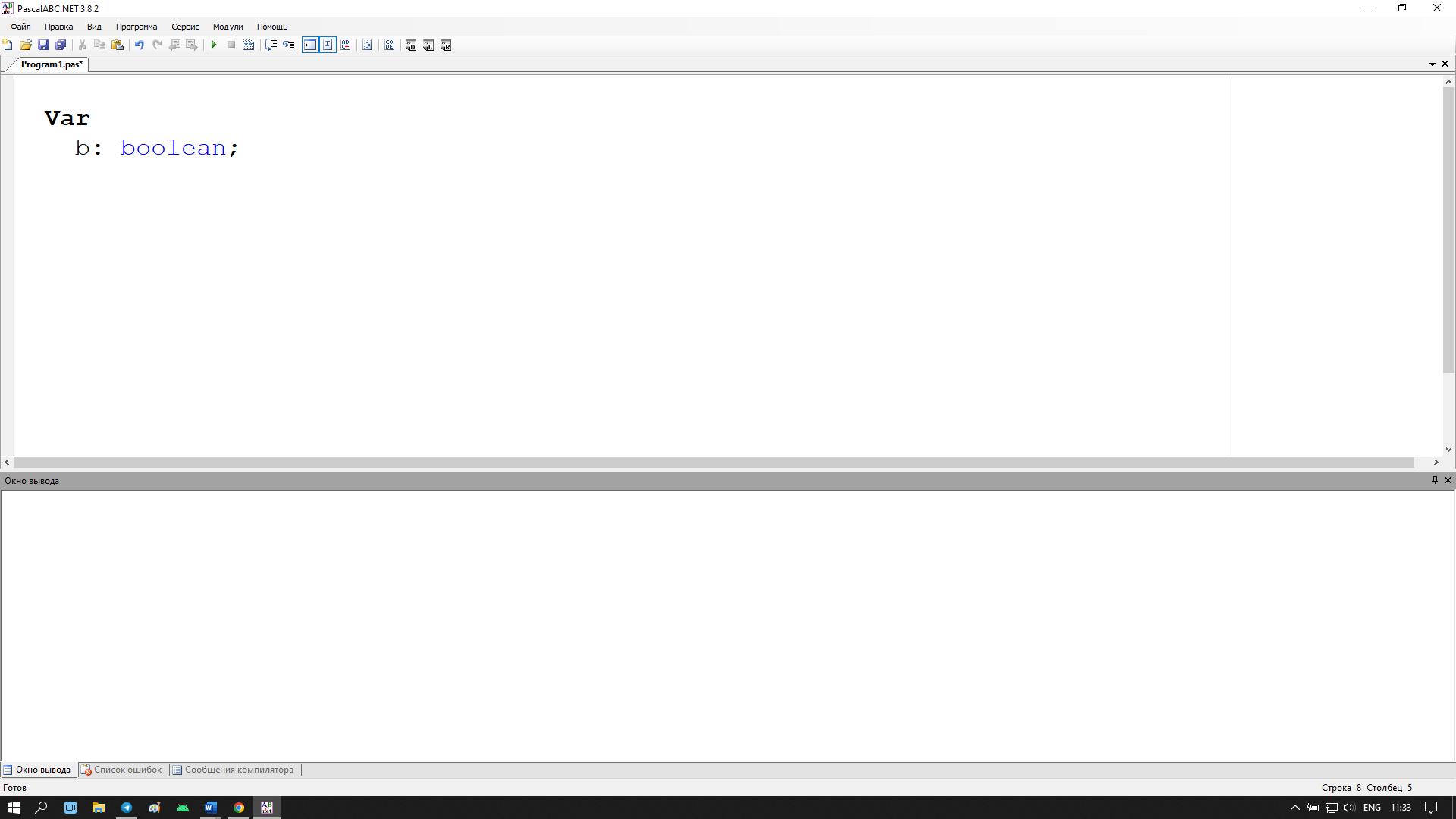


Рисунок 1.6 – Объявление переменной типа «boolean»

Значением логической переменной может быть только «true» или «false».

При работе с логическими переменными используют логические операции.

Логические операции представлены в таблице 1,3.

Таблица 1,3 – Логические операции

|  |  |  |  |
| --- | --- | --- | --- |
| Оператор | Операция | Тип операндов | Тип результата |
| not | отрицание | Boolean | Boolean |
| and | логическое И | Boolean | Boolean |
| or | логическое ИЛИ | Boolean | Boolean |
| xor | логическое исключающее ИЛИ | Boolean | Boolean |

Операции над логическими переменными составляют таблицу истинности.

Таблица истинности представлена в таблице 1,4.

Таблица 1,4 – Таблица истинности

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | NOT (A) | (A) OR (B) | (A) AND (B) |
| T | T | F | T | T |
| T | F | F | T | F |
| F | T | T | T | F |
| F | F | T | F | F |

# ОПИСАНИЕ АЛГОРИТМОВ И ТЕХНОЛОГИИ РЕШЕНИЯ

## Описание требований и исходных данных

Приложение должно иметь удобный и понятный для пользователя интерфейс, приятный стиль оформления и рабочий функционал.

В приложении должны присутствовать следующие функции:

* просмотр материала;
* фильтрация материала по классам и разделам;
* прохождение тестов самоконтроля.

Статистика сохраняется в словаре, в который заносятся пара ключ-значение. Ключом выступает название пройденного теста, а значением число правильных ответов.

Пример словаря со значением правильных ответов представлен на рисунке 2.1.

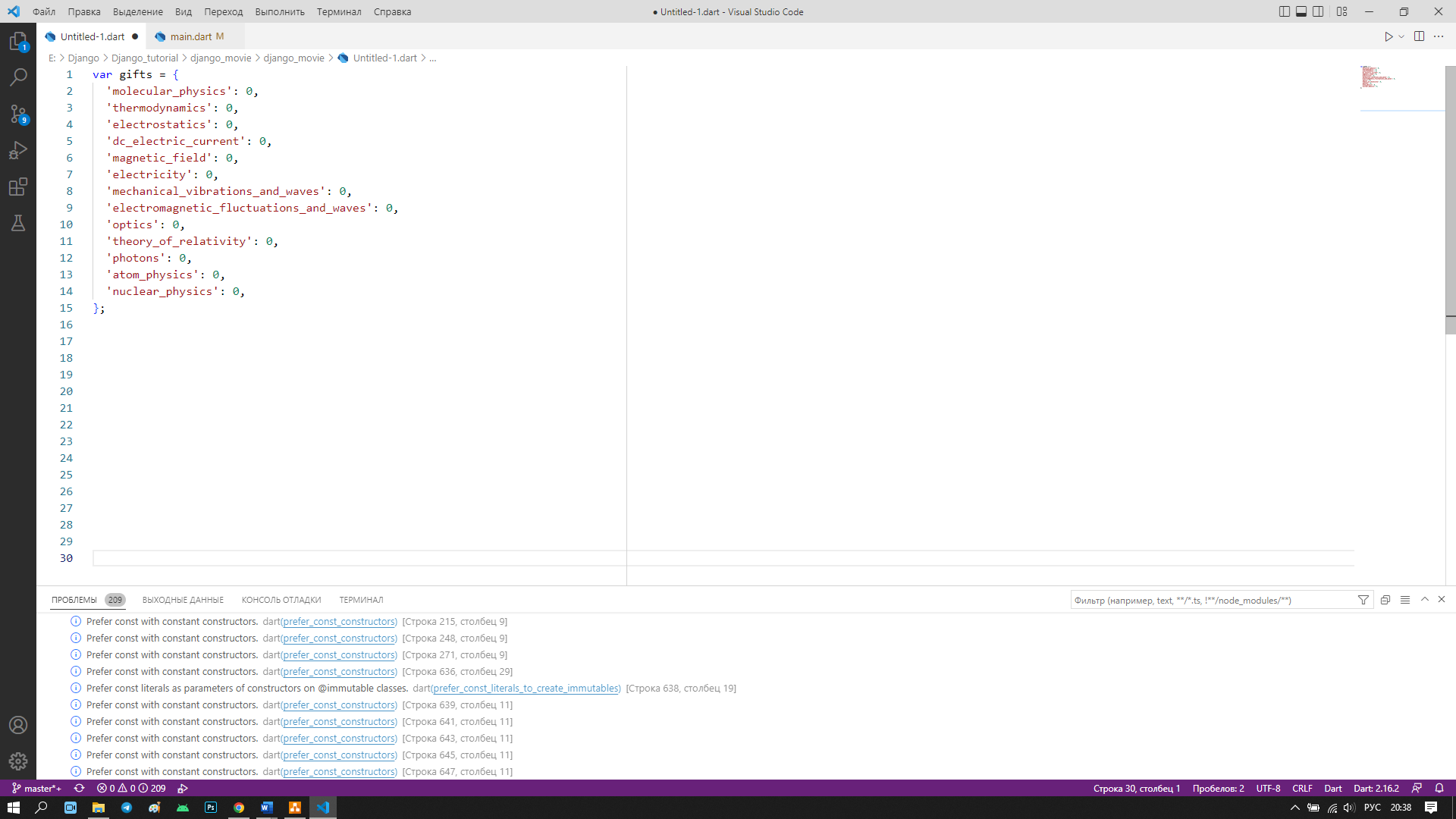


Рисунок 2.1 – Пример словаря со значением правильных ответов

## 2.3 Разработка интерфейса приложения

Приложение должно иметь удобный и понятный для пользователя интерфейс, приятный стиль оформления и рабочий функционал.

В приложении должны присутствовать следующие функции:

* просмотр материала;
* фильтрация материала по классам и разделам;
* прохождение тестов самоконтроля;
* просмотр формул, отфильтрованных по классам;
* просмотр таблиц значений.

На главной форме пользователю предоставляется информация об его изучении материала. Ниже от статистики пользователя располагается список выбора, предоставляемого материала разбитый по классам.

После выбора класса, пользователю будет предоставлен выбор теории или формул.

После выбора теории, пользователь может выбрать нужный ему раздел. Далее в текстовой и графическом формате, ему будет представлена информация материала.

После выбора формул, пользователю будет предоставлены формулы в графической и текстовой форме.

После прохождения каждого раздела пользователю будет представлена возможность пройти тест по пройденному материалу.

Форма – это место, где пользователь общается с программой. Приложение может иметь несколько форм, каждая из которых выполняет свое особое предназначение.

Главная форма – это основная форма, с которой начинается построение программы.

Макет графического интерфейса начальной формы представлен на рисунке 2.2.

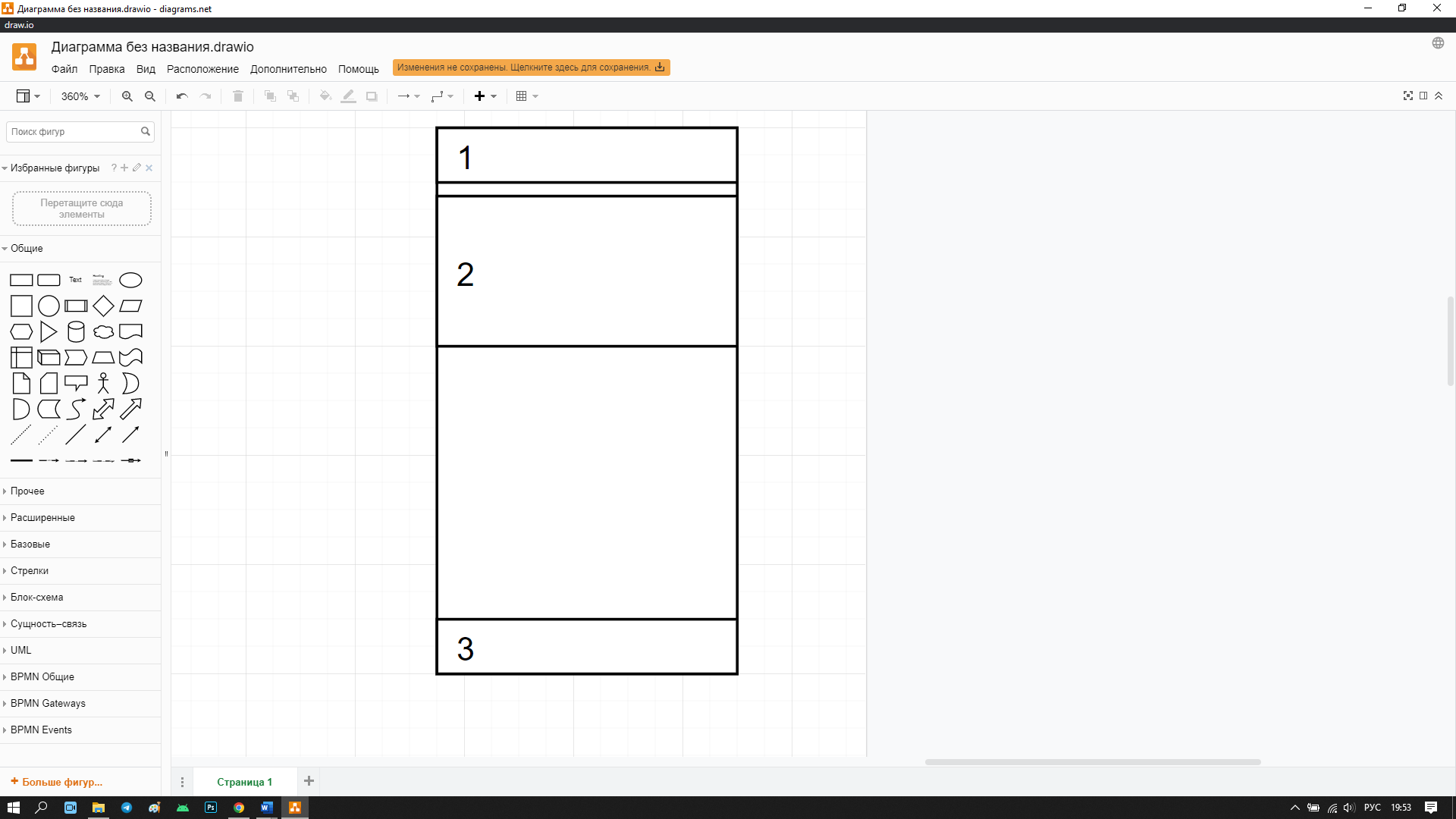


Рисунок 2.2 – Главная форма приложения. 1 – Название раздела. 2 – Список доступных разделов

Макет графического интерфейса формы показа материала представлен на рисунке 2.3.

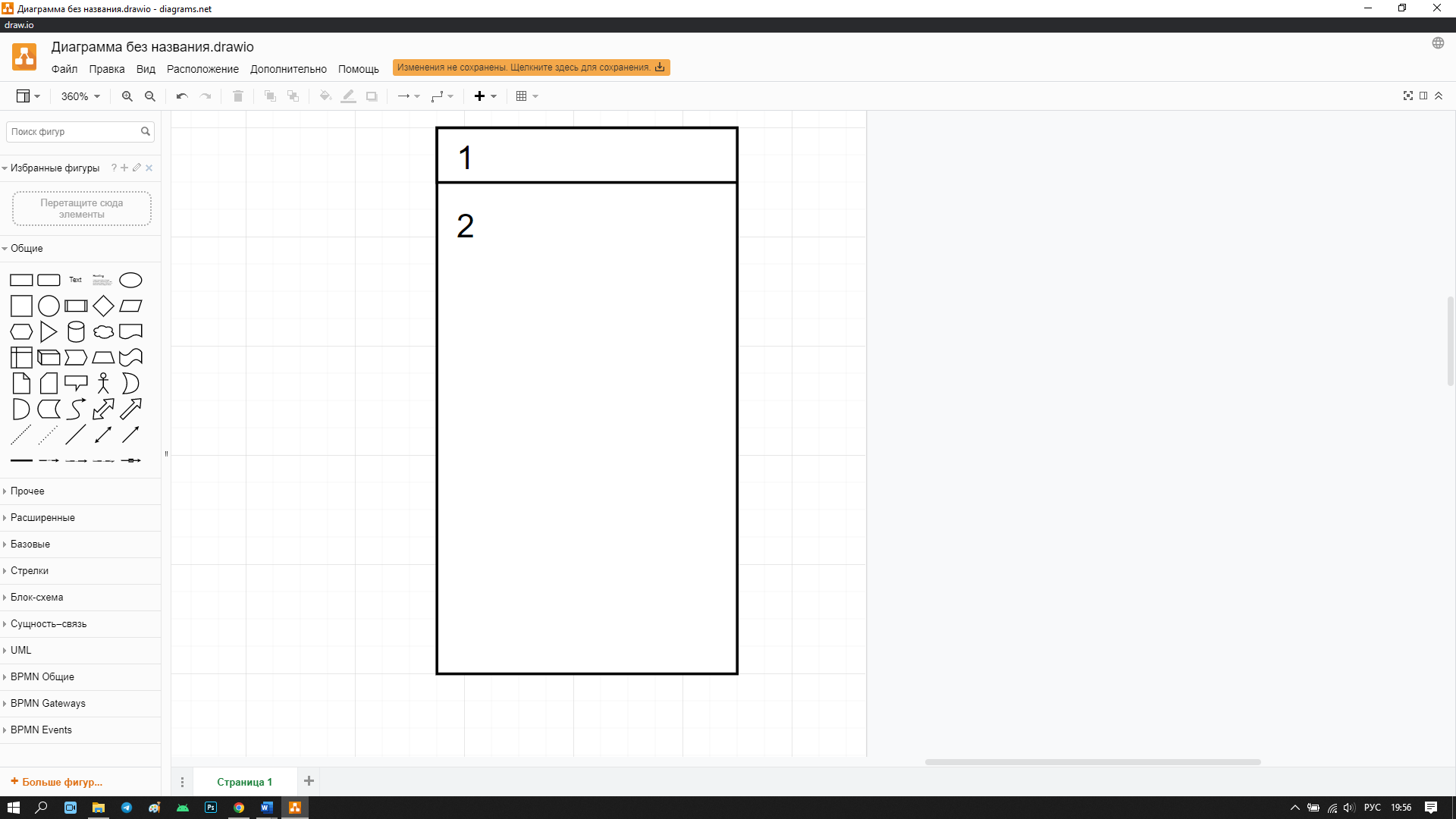


Рисунок 2.3 – Форма показа материала. 1 – название раздела; 2 – текст материала.

Макет графического интерфейса формы прохождения тестов представлен на рисунке 2.4.

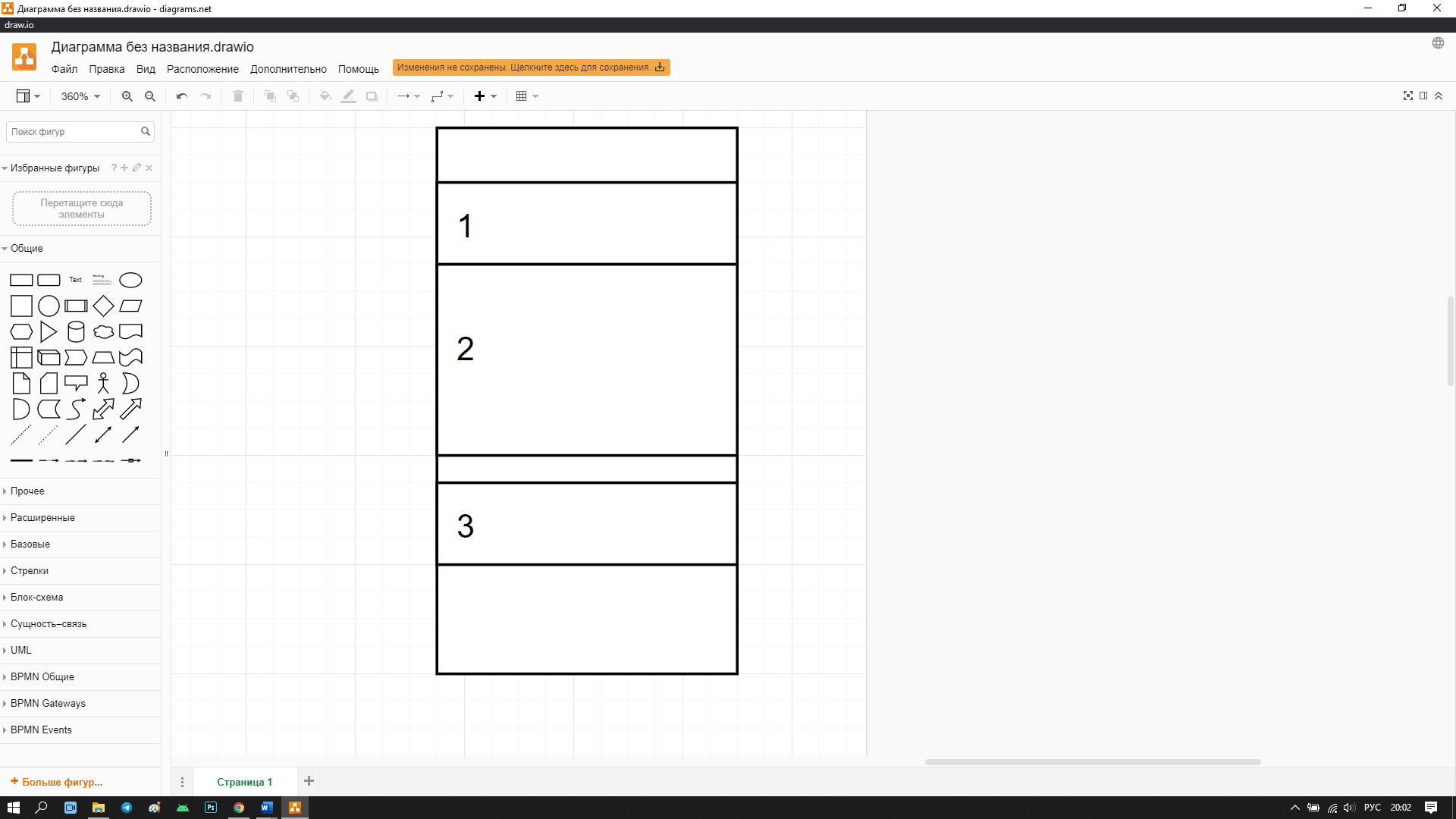


Рисунок 2.4 – Форма прохождения тестов. 1 – Текст вопроса; 2 – Список выбираемого ответа; 3– Проверка результата после прохождения теста.

Макет графического интерфейса формы просмотра результата пройденного теста представлен на рисунке 2.5.

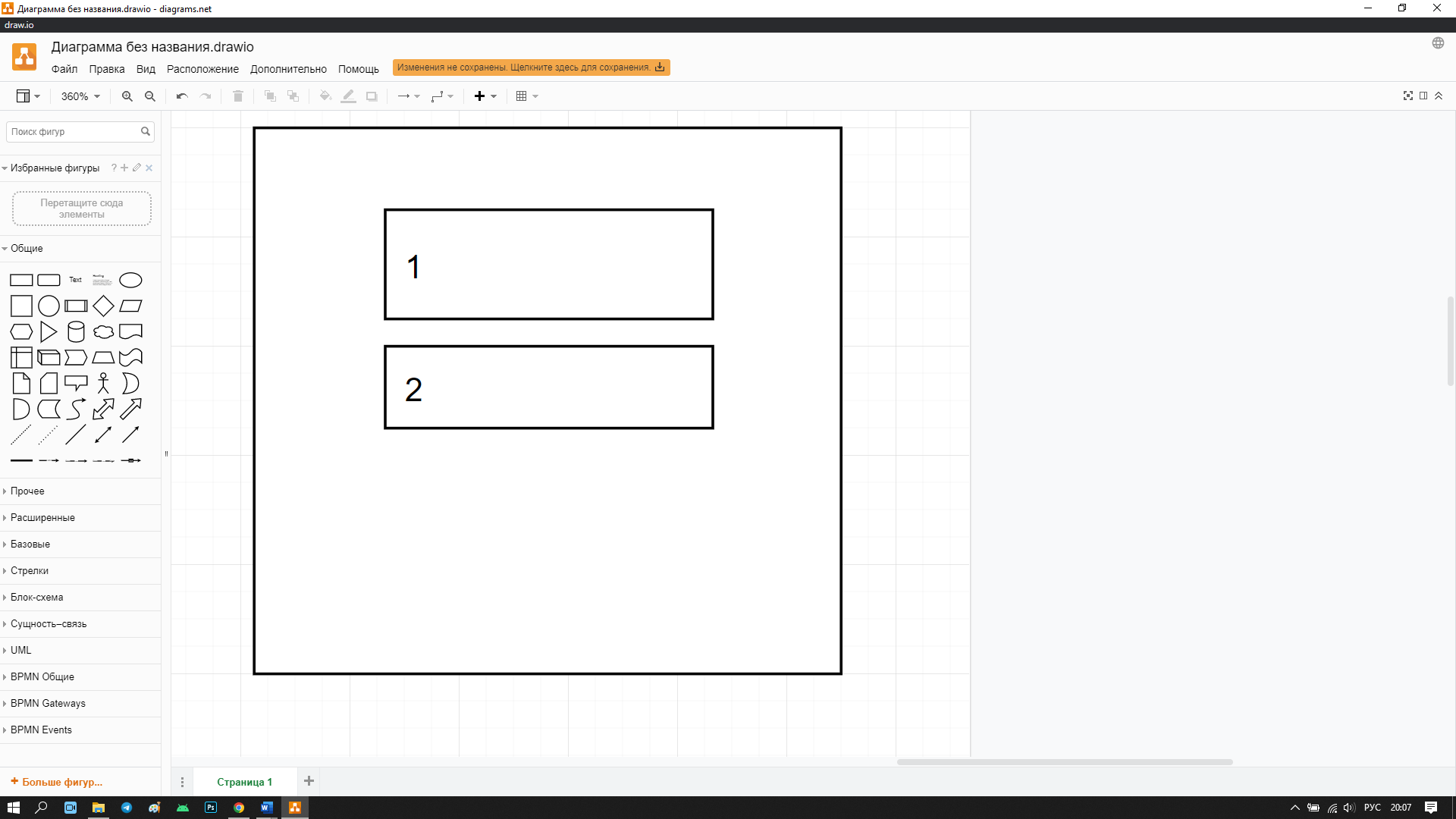


Рисунок 2.5 – Форма просмотра результата пройденного теса. 1 – Число правильных ответов; 2 – Возращение на главную форму

# 3 ИСПОЛНЕНИЕ ПРАКТИЧЕСКОЙ ЧАСТИ

## 3.1 Описание интерфейса

Интерфейс — это «мост», посредник между человеком, программами и машинами, иными системами.

При открытии приложения, показывается начальное окно. Начальным окном является форма, где располагается список доступного материала, разбитый по классам и разделам.

Начальная форма представлена на рисунке 3.1.



Рисунок 3.1 – Главная форма приложения

При нажатии на атрибут списка выбора, появляется список доступного материала, разбитый на разделы.

Список материалов представлен на рисунке 3.2.

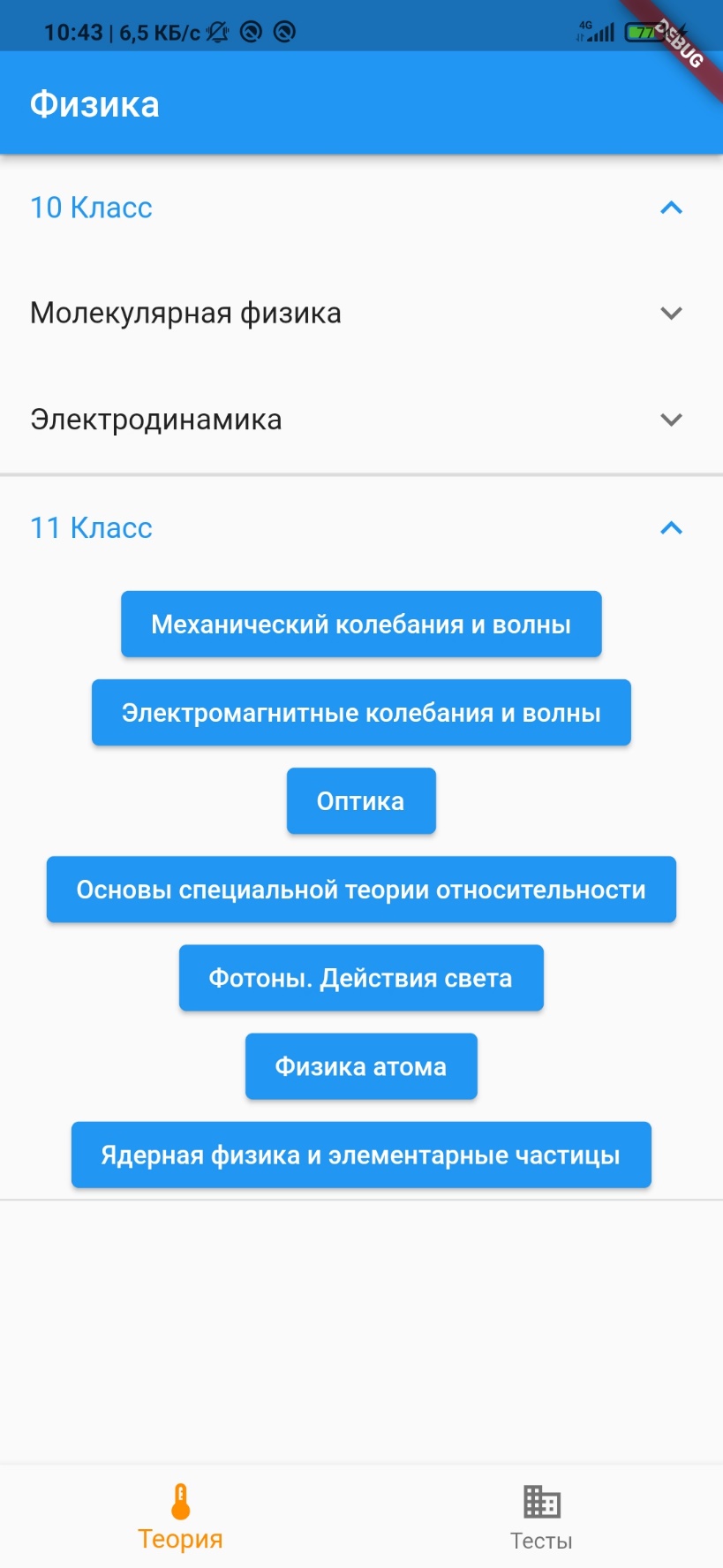


Рисунок 3.2 – Форма выбора материала

По нажатию на кнопку с названием раздела, открывается страница с выбранным материалом.

На рисунке 3.3 представлена форма показа материала.

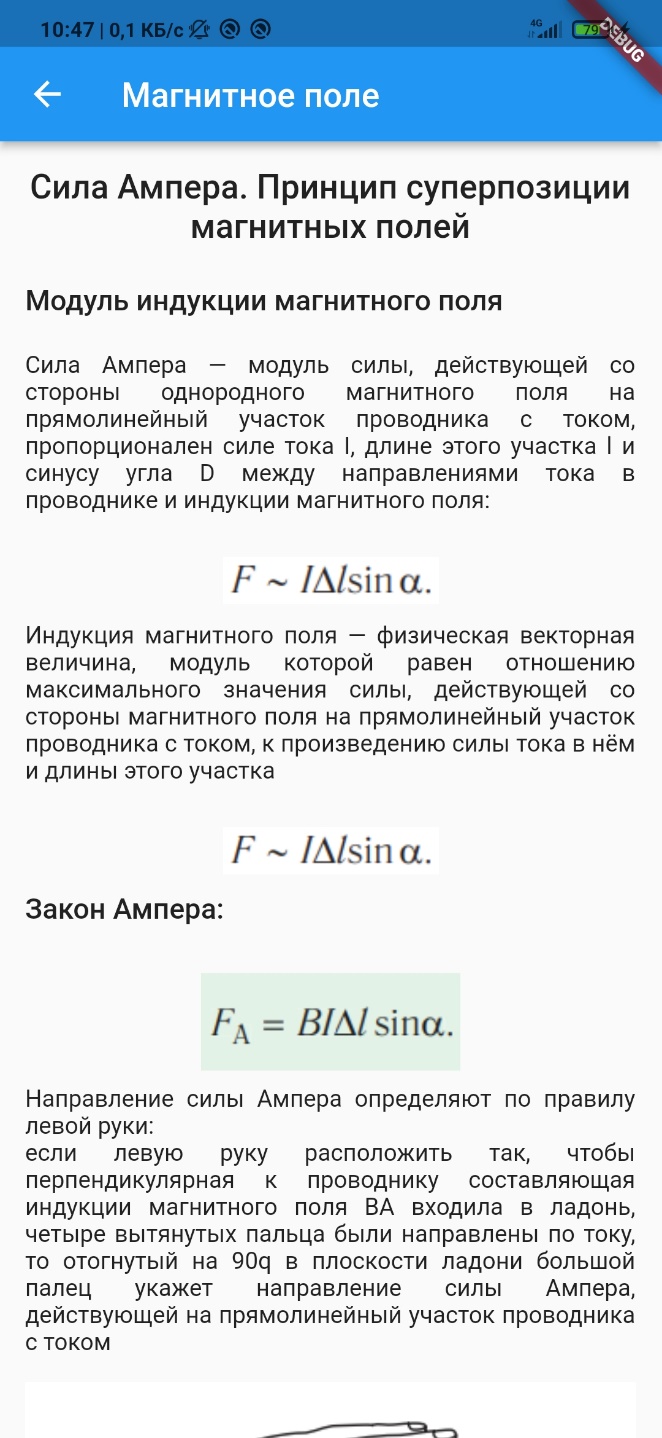


Рисунок 3.3 – Форма показа материала

Просмотр определенного параграфа происходит путем пролистывания формы вниз. Переход по параграфам происходит путем свайпа влево или вправо.

По нажатию на кнопку тесты, снизу экрана приложения, открывается страница со списком тестов.

Форма просмотра списка тестов представлен на рисунке 3.4.

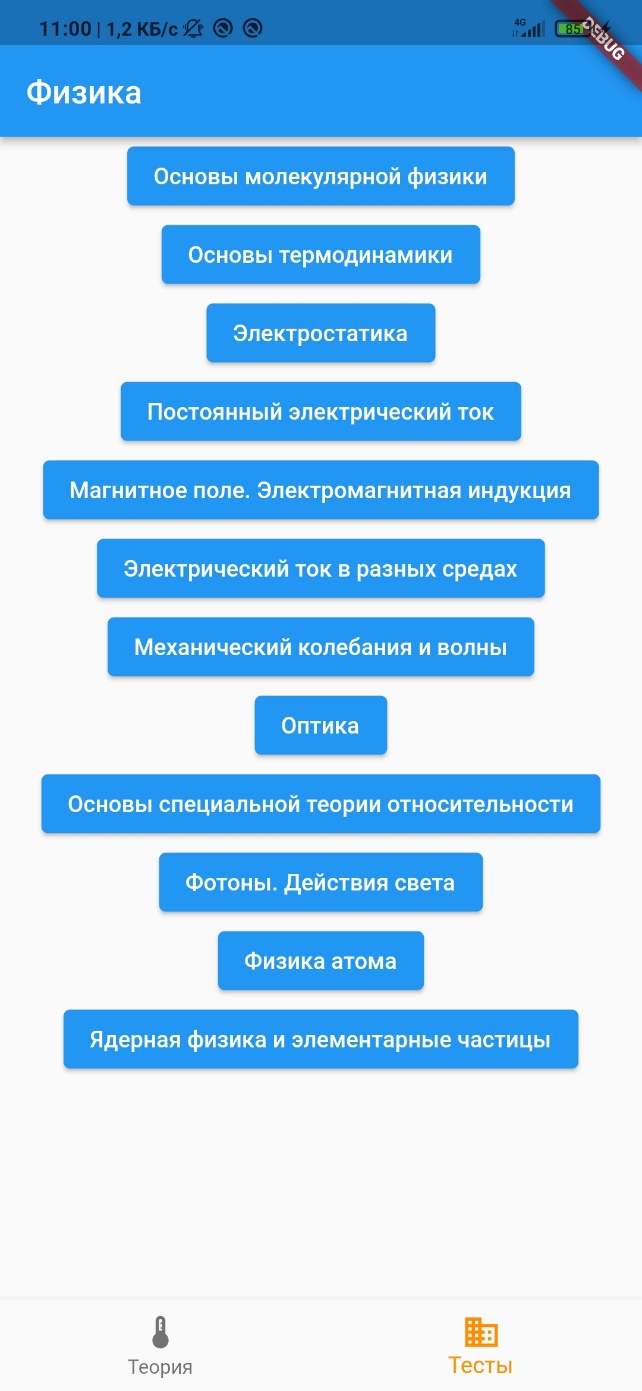


Рисунок 3.4 – Форма показа списка тестов

При нажатии кнопки с желаемой темой прохождения теста, открывается окно с прохождением теста. Для начала прохождения теста, необходимо сверху нажать на кнопку «Начать тест». Правильный ответ помечается круглой галочкой напротив ответа. Для проверки правильности ответа следует нажать на кнопку «Проверить»

Форма прохождения теста представлена на рисунке 3.5.

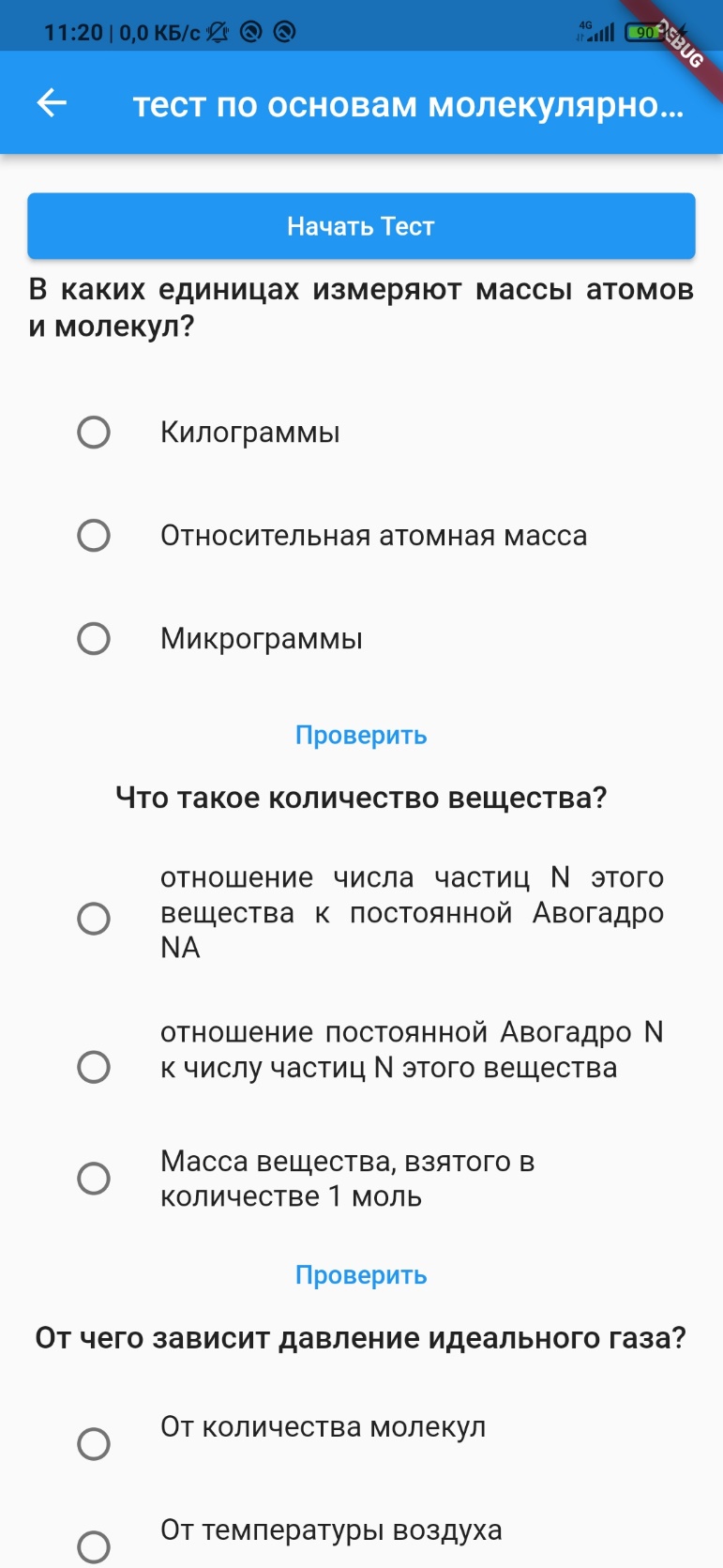


Рисунок 3.5 – Форма прохождения теста

Для просмотра числа правильных ответов, необходимо нажать на кнопку «Результат». После нажатия кнопки, откроется форма с числом правильных ответов. При нажатии на кнопку «На главную», пользователь перенаправляется на главную страницу.

Форма показа числа правильных ответов представлена на рисунке 3.6.



Рисунок 3.6 – Форма показа числа правильных ответов

## 3.2 Программная реализация

Пользователь, при взаимодействии с программой, переходит по формам. Каждая форма имеет свою ссылку, по которой программа переадресовывает пользователя по контенту.

Адресация форм представлена на рисунке 3.7.

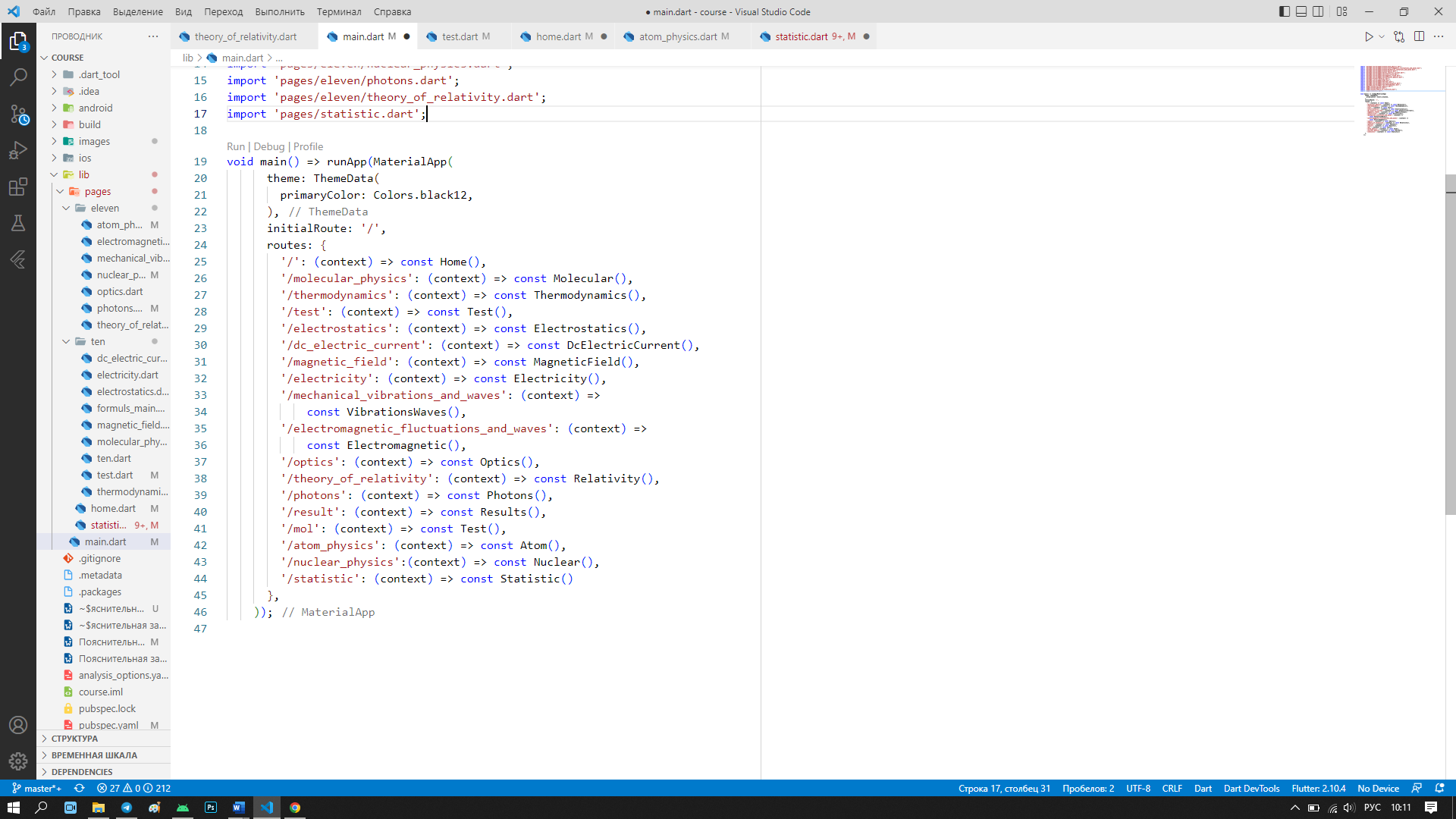


Рисунок 3.7 – Адресация форм приложения

Кнопки служат для перенаправления пользователя по программе и подтверждения действия.

Код кнопки переадресации представлен на рисунке 3.8.

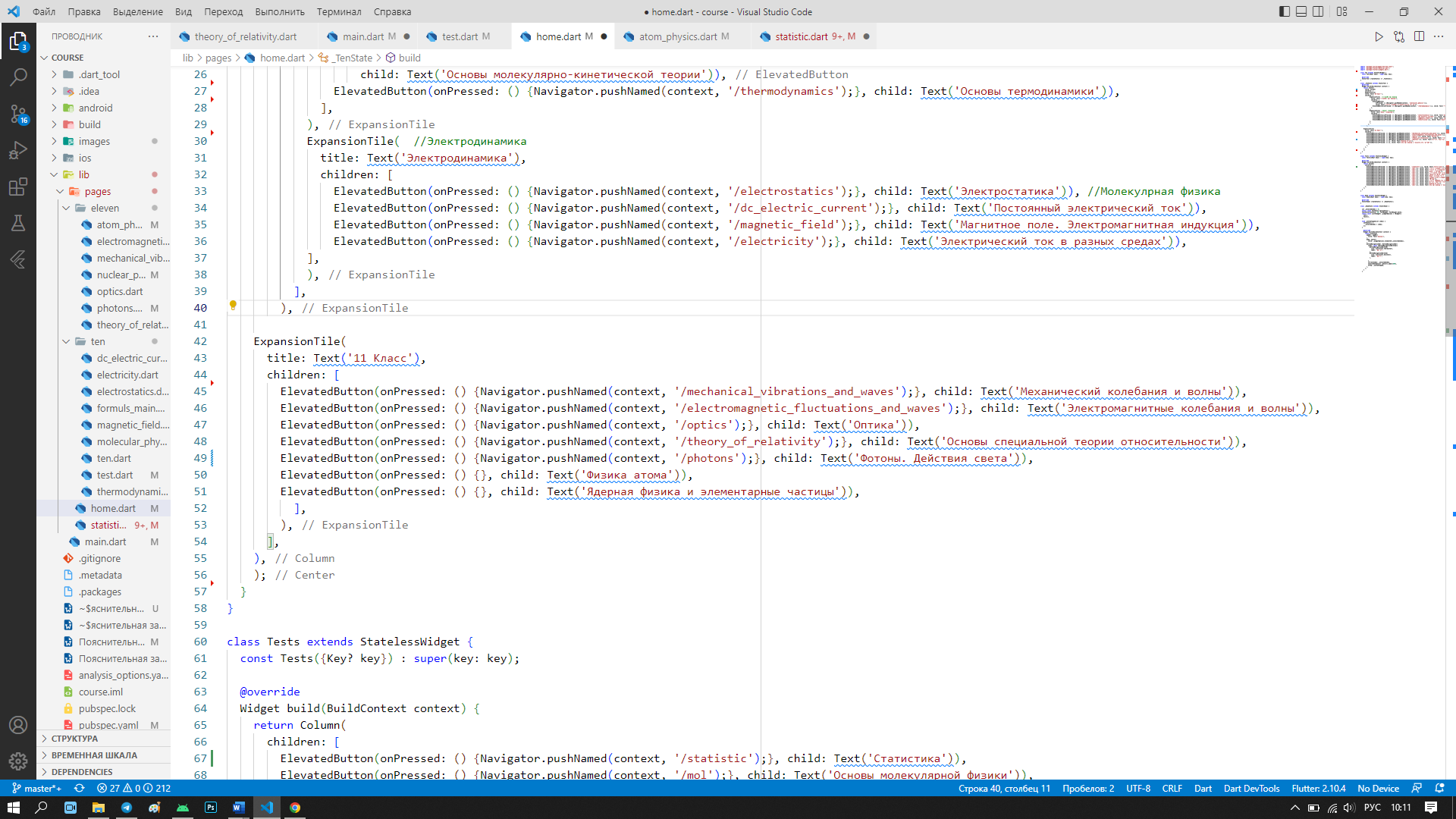


Рисунок 3.8 – Кнопка переадресации

Для перемещения между материалом и тестом использовалась нижняя панель навигации.

Код нижней панели навигации представлен на рисунке 3.9.

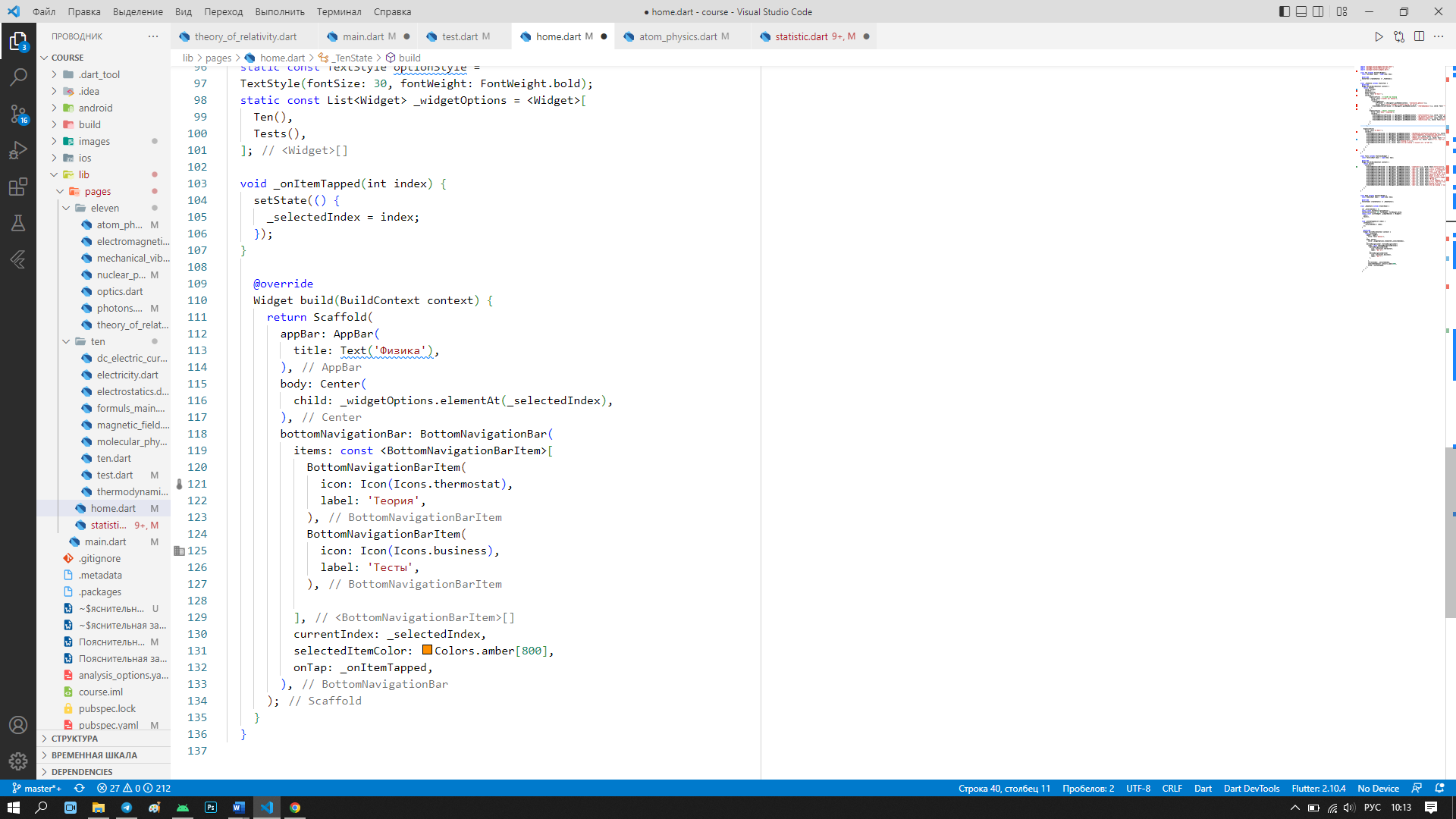


Рисунок 3.9 – Нижняя панель навигации

Для проверки теста каждому варианту ответа присваивался ключ, по которому вариант ответа сверялся с правильным ответом, после чего, если ответ пользователя верный, к счетчику правильных ответов пользователя прибавляется единица.

Код функции проверки правильности ответа пользователя и увеличения счетчика правильных ответов представлен на рисунке 3.10.

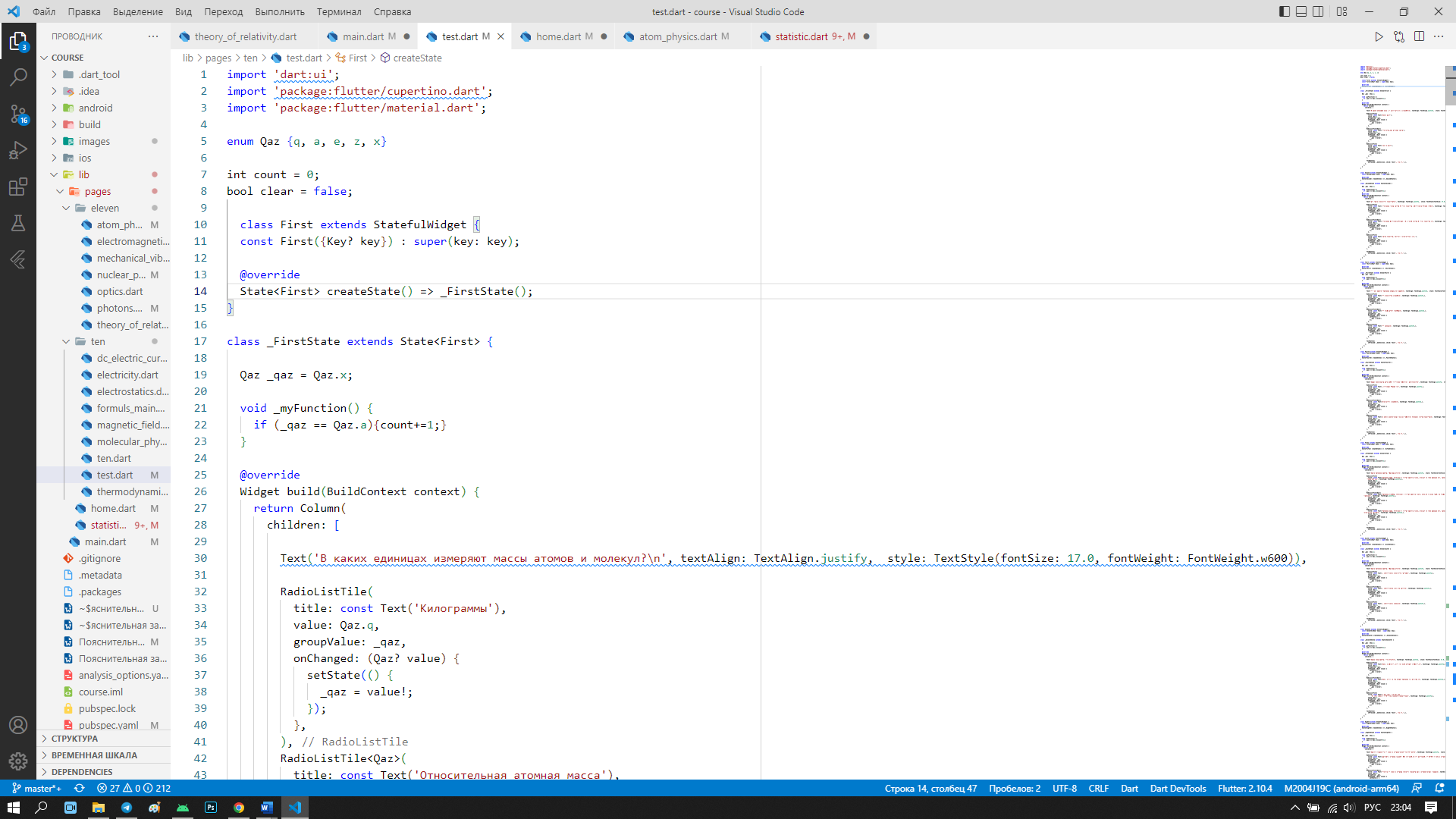


Рисунок 3.10 – Проверка правильности ответа пользователя и увеличения счетчика правильных ответов пользователя

Полный код программы представлен в приложении А.

# 4 результаты программы

При запуске программы открывается главное окно приложения, на котором пользователь может выбрать теоретический материал.

Интерфейс начального окна приложения представлен на рисунке 4.1.



Рисунок 4.1 – Интерфейс главного окна

При нажатии на кнопку «10 Класс» или «11 Класс», появится список доступного материала.

Список материалов представлен на рисунке 4.2.

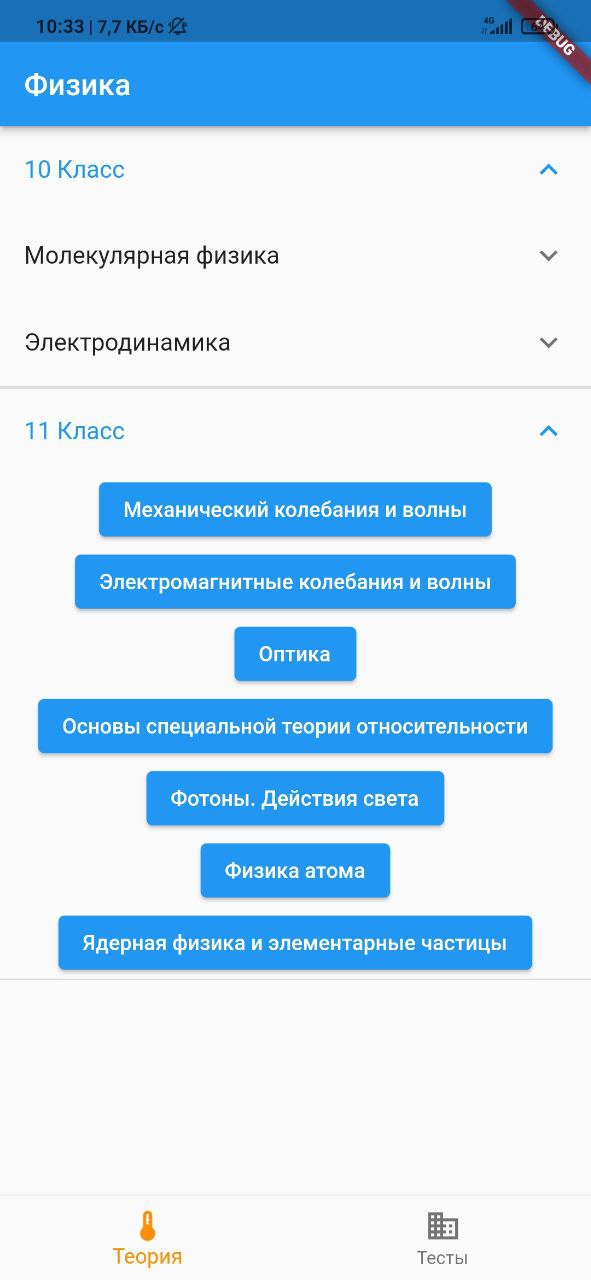


Рисунок 4.2 – Список доступных материалов

При нажатии на кнопку с названием раздела, пользователя перенаправляет на страницу с выбранным материалом.

Страница материала представлена на рисунке 4.3.



Рисунок 4.3 – Страница показа выбранного материала

На странице теоретического материала, при нажатии на формулу, появляется окно внизу экрана, которое поясняет буквы формулы.

Поясняющее окно представлено на рисунке 4.4.

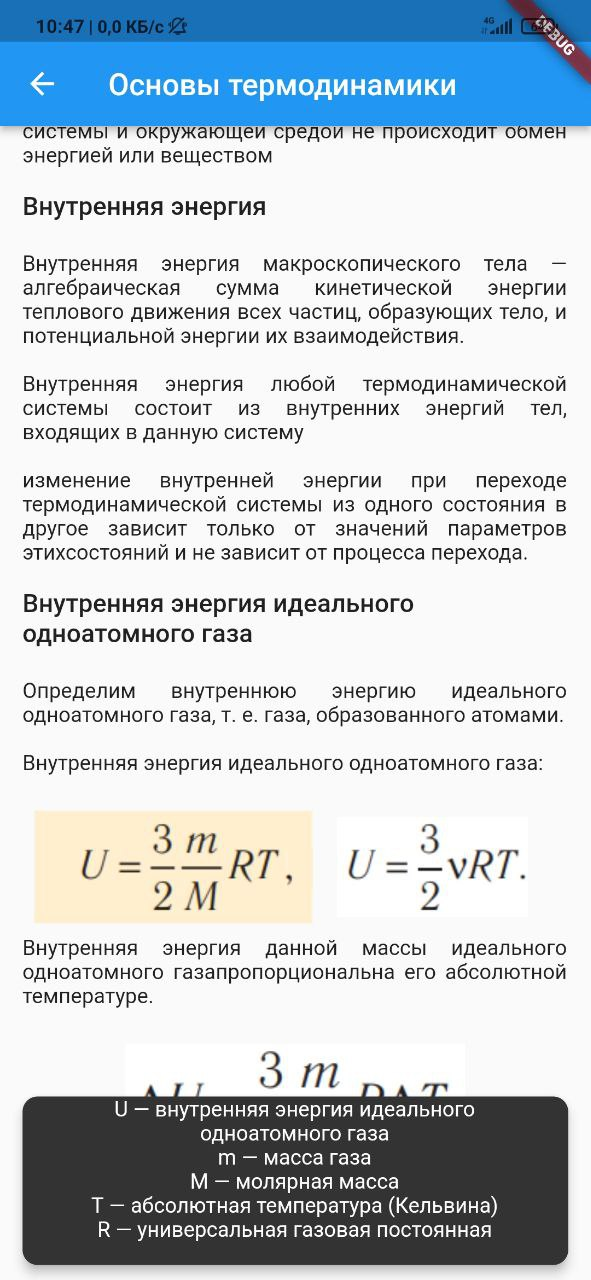


Рисунок 4.4 – Поясняющее окно

При нажатии на кнопку «Тесты» на нижней панели навигации открывается страница с кнопкой для просмотра статистики и со списком доступным тестов.

Окно доступных тестов представлен и кнопкой для просмотра статистики на рисунке 4.5.

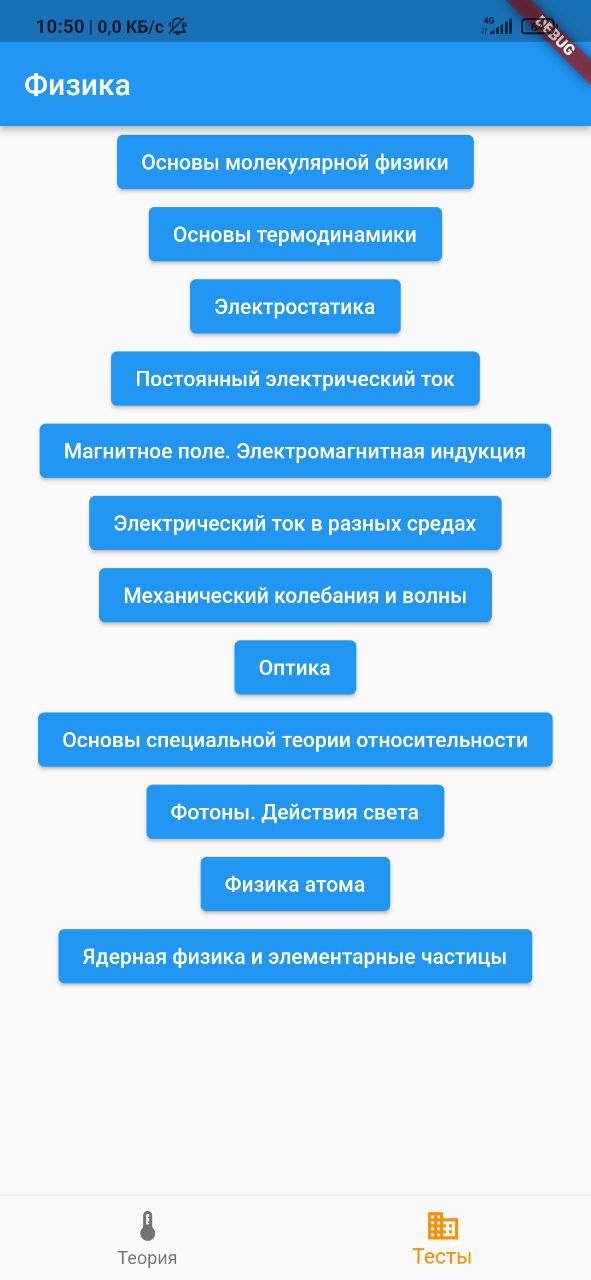


Рисунок 4.5 – Список доступных тестов

При нажатии на кнопку с надписью темы теста, открывается окно для прохождения теста. Тест начинается при нажатии на кнопку «Начать тест». После выбора ответа, пользователю необходимо нажать на кнопку «Проверить» для отправки результата ответа. Результат показывается в новом окне при нажатии на кнопку «Результат».

Окно прохождения теста представлен на рисунке 4.6.

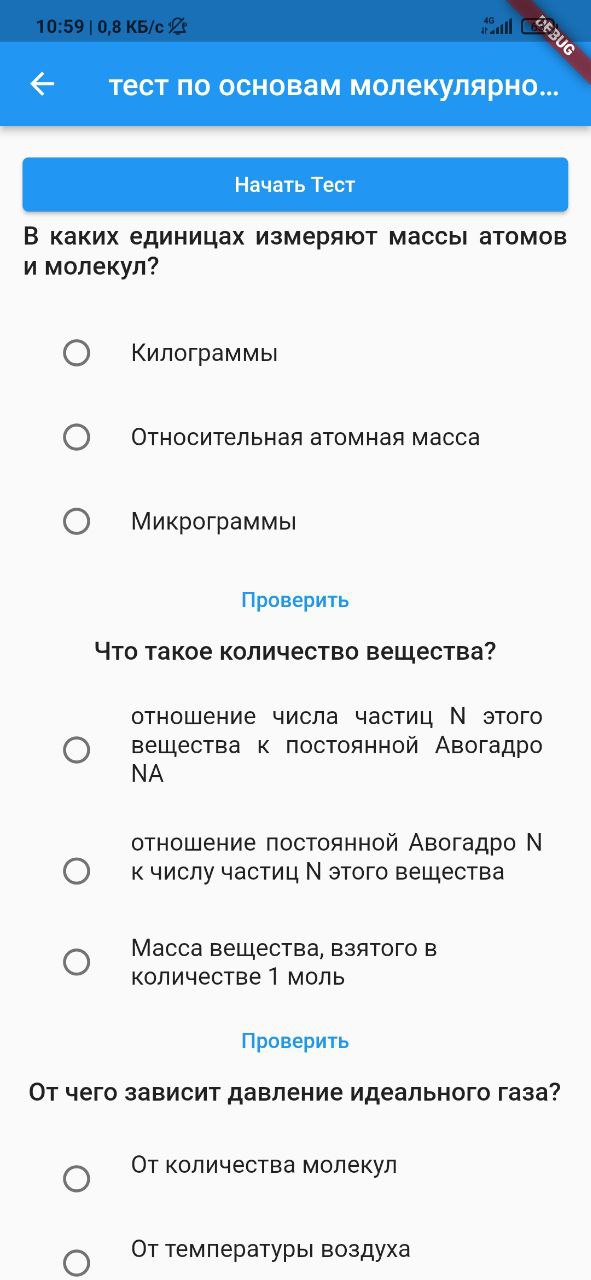


Рисунок 4.6 – Окно прохождения теста

Окно показа результата прохождения теста представлен на рисунке 4.7.



Рисунок 4.7 – Окно показа результата прохождения теста

При нажатии на кнопку «На главную», пользователя перенаправляет на главную страницу.

Главная страница представлена на рисунке 4.1.

При нажатии на кнопку «Статистика», открывается окно, где находятся темы и число правильных ответов теста.

Окно статистики представлено на рисунке 4.8.

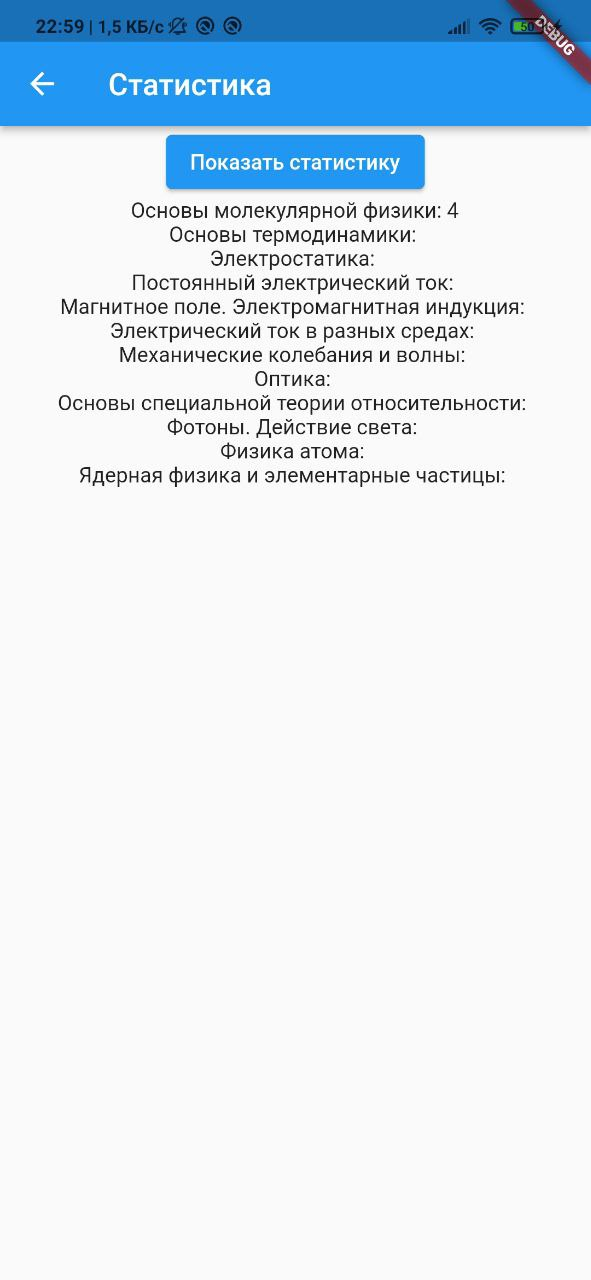


Рисунок 4.8 – Окно показа статистики

### **Заключение**

Исходя из цели курсового проекта, изучен теоретический материал на тему «Простые типы данных» и разработана обучающее приложение «Физика для старшей школы».

Для реализации всей разработки был выбран язык программирования «Dart» и фреймворк «Flutter», разрабатывая на котором, можно просто и быстро создавать мобильное приложение, путем создания виджетов.

Актуальность курсового проекта заключается в том, что при помощи программы, учащиеся старшей школы могут легко и быстро повторять материал и проходить тесты самоконтроля для того, чтобы проверить свои знания на практике.

В ходе создания курсового проекта были решены такие задачи как:

* подготовка пояснительной записки;
* разработка алгоритмов;
* разработка интерфейсов;
* разработка стандартного функционала программы.

При выполнении курсового проекта была изучена теория и технология работы со средой разработки «Flutter» и проанализирована соответствующая литература.

Курсовой проект содержит следующие функции:

* Просмотр теоретических материалов;
* прохождение тестов самоконтроля;
* просмотр статистики пройденных тестов.

### **Список используемых источников**

1. Осипов А В. PascalABC.NET: Введение в современное программирование: учебник / Осипов А В.– Мн. : – Ростов-на-Дону, 2019. – 38 с.
2. Абрамян М. Э. Структуры данных в PascalABC.NET. Выпуск 1. Массивы и последовательности. Запросы. / Абрамян М. Э.- Мн.: — Ростов н/Д, 2016. – 7 c.
3. Павловская Т. А. Паскаль. Программирование на языке высокого уровня: Учебник для вузов. 2-е изд./ Павловская Т. А. – Мн. : — СПб.: Питер, 2010. – 49с.
4. Культин Н. Б. Программирование в Turbo Pascal 7.0 и Delphi: 3-е изд., перераб. и доп./ Культин Н. Б.— Мн.: — СПб.: БХВ-Петербург, 2007. — 27 с.
5. Зеленяк О.П. Практикум программирования на Turbo Pascal. Задачи, алгоритмы и решения.— 3-е изд., испр. и доп./ Зеленяк О.П.— Мн.: — СПб.: ДиаСофтЮП, М.: ДМК — 17 с.
6. Д. Ушаков, Т. Юркова Паскаль для школьников, 2-е издание — СПб.: Питер, 2011/ Д. Ушаков, Т. Юркова — Мн.: — СПб.: Питер, 2011. — 38 c.

**Приложение А**

(обязательное)

**Текст** **приложения «Физика для старшей школы»**

Текст модуля theory\_of\_relativity.dart

import 'package:flutter/material.dart';

Text text(string) => Text(string+'\n', textAlign: TextAlign.justify,);

Text title(string) => Text(string+'\n', textAlign: TextAlign.center,

  style: const TextStyle(fontSize: 20.0, fontWeight: FontWeight.w600),);

const PADING = EdgeInsets.only(top: 15, left: 15, right: 15,);

class First extends StatelessWidget {

  const First({Key? key}) : super(key: key);

  @override

  Widget build(BuildContext context) {

    return ListView(

      padding: PADING,

      children: [

        text('Специальная теория относительности — раздел физики, в котором изучается '

            'свойства пространства и времени, а также законы движения тел при скоростях, '

            'сравнимых со скоростью света.'),

        title('Постулаты специальной теории относительности'),

        text('Первый постулат СТО (постулат относительности): во всех инерциальных '

            'система отсчета все физические явления при одинаковых начальных условиях '

            'происходят одинаковым образом.'),

        text('Второй постулат СТО (постулат постоянства скорости света): во всех '

            'инерциальных системах отсчета скорость света в вакууме одинакова и '

            'не зависит ни от скорости источника, ни от скорости приемника излучения.'),],);}}

class Second extends StatefulWidget {

  const Second({Key? key}) : super(key: key);

  @override

  State<Second> createState() => \_SecondState();

}

class \_SecondState extends State<Second> {

  @override

  Widget build(BuildContext context) {

    return ListView(

      padding: PADING,

      children: [

        title('Элементы релятвистской динамики. Взаимосвязь массыи энергии'),

        text('Утверждение о том, что энергия покоя пропорциональна массе, а изменение '

            'энергии покоя связано с изменением массы, называют законом взаимосвязи массы и энергии:'),

        TextButton(

          child: const Image(image: AssetImage('images/theory\_of\_relativity/d\_m.png')),

          onPressed: () {

            ScaffoldMessenger.of(context).showSnackBar(

              SnackBar(

                content: const Text(

                  'Δm — изменение массы тела\n'

                      'ΔE0 — изменение энергии покоя тела\n'

                      'с — скорость света в вакууме\n', textAlign: TextAlign.center,),

                padding: const EdgeInsets.symmetric(horizontal: 15.0,),

                behavior: SnackBarBehavior.floating,

                shape: RoundedRectangleBorder(

                  borderRadius: BorderRadius.circular(10.0),

                ),),);},),],);}}

class Relativity extends StatelessWidget {

  const  Relativity({Key? key}) : super(key: key);

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(title: const Text('Основы специальной теории относительности'),),

      body: PageView(

        children: [

          First(),

          Second(),],),);}}

Текст модуля main.dart

import 'package:course/pages/eleven/atom\_physics.dart';

import 'package:course/pages/eleven/electromagnetic\_fluctuations\_and\_waves.dart';

import 'package:course/pages/eleven/mechanical\_vibrations\_and\_waves.dart';

import 'package:course/pages/eleven/optics.dart';

import 'package:course/pages/ten/dc\_electric\_current.dart';

import 'package:course/pages/ten/electricity.dart';

import 'package:course/pages/ten/magnetic\_field.dart';

import 'package:course/pages/ten/molecular\_physics.dart';

import 'package:flutter/material.dart';

import 'package:course/pages/home.dart';

import 'package:course/pages/ten/test.dart';

import 'package:course/pages/ten/electrostatics.dart';

import 'package:course/pages/ten/thermodynamic.dart';

import 'pages/eleven/nuclear\_physics.dart';

import 'pages/eleven/photons.dart';

import 'pages/eleven/theory\_of\_relativity.dart';

import 'pages/statistic.dart';

void main() => runApp(MaterialApp(

      theme: ThemeData(

        primaryColor: Colors.black12,

      ),

      initialRoute: '/',

      routes: {

        '/': (context) => const Home(),

        '/molecular\_physics': (context) => const Molecular(),

        '/thermodynamics': (context) => const Thermodynamics(),

        '/test': (context) => const Test(),

        '/electrostatics': (context) => const Electrostatics(),

        '/dc\_electric\_current': (context) => const DcElectricCurrent(),

        '/magnetic\_field': (context) => const MagneticField(),

        '/electricity': (context) => const Electricity(),

        '/mechanical\_vibrations\_and\_waves': (context) =>

            const VibrationsWaves(),

        '/electromagnetic\_fluctuations\_and\_waves': (context) =>

            const Electromagnetic(),

        '/optics': (context) => const Optics(),

        '/theory\_of\_relativity': (context) => const Relativity(),

        '/photons': (context) => const Photons(),

        '/result': (context) => const Results(),

        '/mol': (context) => const Test(),

        '/atom\_physics': (context) => const Atom(),

        '/nuclear\_physics':(context) => const Nuclear(),

        '/statistic': (context) => const Statistic()},));

Текст модуля test.dart

import 'dart:ui';

import 'package:flutter/cupertino.dart';

import 'package:flutter/material.dart';

enum Qaz {q, a, e, z, x}

int count = 0;

bool clear = false;

  class First extends StatefulWidget {

  const First({Key? key}) : super(key: key);

  @override

  State<First> createState() => \_FirstState();

}

class \_FirstState extends State<First> {

  Qaz \_qaz = Qaz.x;

  void \_myFunction() {

    if (\_qaz == Qaz.a){count+=1;}

  }

  @override

  Widget build(BuildContext context) {

    return Column(

      children: [

        Text('В каких единицах измеряют массы атомов и молекул?\n', textAlign: TextAlign.justify,  style: TextStyle(fontSize: 17.0, fontWeight: FontWeight.w600)),

        RadioListTile(

          title: const Text('Килограммы'),

          value: Qaz.q,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile<Qaz>(

          title: const Text('Относительная атомная масса'),

          value: Qaz.a,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile(

          title: const Text('Микрограммы'),

          value: Qaz.e,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        TextButton(

          onPressed: \_myFunction, child: Text('Проверить'),),],);}}

class Second extends StatefulWidget {

  const Second({Key? key}) : super(key: key);

  @override

  State<Second> createState() => \_SecondState();

class \_SecondState extends State<Second> {

  Qaz \_qaz = Qaz.x;

  void \_myFunction() {

    if (\_qaz == Qaz.q){count+=1;}

  }

  @override

  Widget build(BuildContext context) {

    return Column(

      children: [

        Text('Что такое количество вещества?\n', textAlign: TextAlign.justify,  style: TextStyle(fontSize: 17.0, fontWeight: FontWeight.w600)),

        RadioListTile(

          title: const Text('отношение числа частиц N этого вещества к постоянной Авогадро NA\n', textAlign: TextAlign.justify,),

          value: Qaz.q,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile<Qaz>(

          title: const Text('отношение постоянной Авогадро N к числу частиц N этого вещества \n', textAlign: TextAlign.justify,),

          value: Qaz.a,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile(

          title: const Text('Масса вещества, взятого в количестве 1 моль'),

          value: Qaz.e,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        TextButton(

          onPressed: \_myFunction, child: Text('Проверить'),),],);}}

class Thirt extends StatefulWidget {

  const Thirt({Key? key}) : super(key: key);

  @override

  State<Thirt> createState() => \_ThirtState();

}

class \_ThirtState extends State<Thirt> {

  Qaz \_qaz = Qaz.x;

  void \_myFunction() {

    if (\_qaz == Qaz.q){count+=1;}

  }

  @override

  Widget build(BuildContext context) {

    return Column(

      children: [

        Text('От чего зависит давление идеального газа?\n', textAlign: TextAlign.justify,  style: TextStyle(fontSize: 17.0, fontWeight: FontWeight.w600)),

        RadioListTile(

          title: const Text('От количества молекул\n', textAlign: TextAlign.justify,),

          value: Qaz.q,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile<Qaz>(

          title: const Text('От температуры воздуха\n', textAlign: TextAlign.justify,),

          value: Qaz.a,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;

            });},),

        RadioListTile(

          title: const Text('От объёма\n', textAlign: TextAlign.justify,),

          value: Qaz.e,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        TextButton(

          onPressed: \_myFunction, child: Text('Проверить'),),

],);}}

class Fourth extends StatefulWidget {

  const Fourth({Key? key}) : super(key: key);

  @override

  State<Fourth> createState() => \_FourthState();

}

class \_FourthState extends State<Fourth> {

  Qaz \_qaz = Qaz.x;

  void \_myFunction() {

    if (\_qaz == Qaz.e){count+=1;}

  }

  @override

  Widget build(BuildContext context) {

    return Column(

      children: [

        Text('Какая величина характеризует состояние теплового равновесия?\n', textAlign: TextAlign.justify,  style: TextStyle(fontSize: 17.0, fontWeight: FontWeight.w600)),

        RadioListTile(

          title: const Text('Постоянная Авагадро\n', textAlign: TextAlign.justify,),

          value: Qaz.q,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile<Qaz>(

          title: const Text('Количество молекул\n', textAlign: TextAlign.justify,),

          value: Qaz.a,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile(

          title: const Text('Средняя кинетическая энергия теплового движения частиц вещества\n', textAlign: TextAlign.justify,),

          value: Qaz.e,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        TextButton(

          onPressed: \_myFunction, child: Text('Проверить'),),],);}}

class Fifen extends StatefulWidget {

  const Fifen({Key? key}) : super(key: key);

  @override

  State<Fifen> createState() => \_FifenState();

}

class \_FifenState extends State<Fifen> {

  Qaz \_qaz = Qaz.x;

  void \_myFunction() {

    if (\_qaz == Qaz.q){count+=1;}

  }

  @override

  Widget build(BuildContext context) {

    return Column(

      children: [

        Text('Какое давление называют парциальным?\n', textAlign: TextAlign.justify,  style: TextStyle(fontSize: 17.0, fontWeight: FontWeight.w600)),

        RadioListTile(

          title: const Text('Давление газа, входящего в состав газовой смеси, если бы он один занимал весь объём, предоставленный смеси, при той же '

        'температуре\n', textAlign: TextAlign.justify,),

          value: Qaz.q,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile<Qaz>(

          title: const Text('Давление воздуха, входящего в состав газовой смеси, если бы он имел такую же температуру, предоставленный смеси, при том же '

    'объёме\n', textAlign: TextAlign.justify,),

          value: Qaz.a,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile(

          title: const Text('Давление газа, входящего в состав газовой смеси, если бы он один занимал весь объём, предоставленный смеси, при том же '

    'количестве частиц\n', textAlign: TextAlign.justify,),

          value: Qaz.e,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        TextButton(

          onPressed: \_myFunction, child: Text('Проверить'),),],);}}

class Sixth extends StatefulWidget {

  const Sixth({Key? key}) : super(key: key);

  @override

  State<Sixth> createState() => \_SixthState();

}

class \_SixthState extends State<Sixth> {

  Qaz \_qaz = Qaz.x;

  void \_myFunction() {

    if (\_qaz == Qaz.a){count+=1;}

  }

  @override

  Widget build(BuildContext context) {

    return Column(

      children: [

        Text('Какое давление называют парциальным?\n', textAlign: TextAlign.justify,  style: TextStyle(fontSize: 17.0, fontWeight: FontWeight.w600)),

        RadioListTile(

          title: const Text('При постоянном количестве частиц\n', textAlign: TextAlign.justify,),

          value: Qaz.q,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile<Qaz>(

          title: const Text('При постоянной молярной массы\n', textAlign: TextAlign.justify,),

          value: Qaz.a,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile(

          title: const Text('При постоянном объёме\n', textAlign: TextAlign.justify,),

          value: Qaz.e,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        TextButton(

          onPressed: \_myFunction, child: Text('Проверить'),),],);}}

class Seventh extends StatefulWidget {

  const Seventh({Key? key}) : super(key: key);

  @override

  State<Seventh> createState() => \_SeventhState();

}

class \_SeventhState extends State<Seventh> {

  Qaz \_qaz = Qaz.x;

  void \_myFunction() {

    if (\_qaz == Qaz.e){count+=1;}

  }

  @override

  Widget build(BuildContext context) {

    return Column(

      children: [

        Text('Какие тела называют твёрдыми?\n', textAlign: TextAlign.justify,  style: TextStyle(fontSize: 17.0, fontWeight: FontWeight.w600)),

        RadioListTile(

          title: const Text('Тело, крепокость которого не превышает шрупкость\n', textAlign: TextAlign.justify,),

          value: Qaz.q,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile<Qaz>(

          title: const Text('Тело, котоорое сдерживает давление в 1 атмосфер\n', textAlign: TextAlign.justify,),

          value: Qaz.a,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile(

          title: const Text('Любое тело, сохраняющее '

              'форму и объём в отсутствие внешних воздействий\n', textAlign: TextAlign.justify,),

          value: Qaz.e,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        TextButton(

          onPressed: \_myFunction, child: Text('Проверить'),),],);}}

class Eighth extends StatefulWidget {

  const Eighth({Key? key}) : super(key: key);

  @override

  State<Eighth> createState() => \_EighthState();

}

class \_EighthState extends State<Eighth> {

  Qaz \_qaz = Qaz.x;

  void \_myFunction() {

    if (\_qaz == Qaz.e){count+=1;}

  }

  @override

  Widget build(BuildContext context) {

    return Column(

      children: [

        Text('Каковы особенности строения кристаллических твёрдых тел?\n', textAlign: TextAlign.justify,  style: TextStyle(fontSize: 17.0, fontWeight: FontWeight.w600)),

        RadioListTile(

          title: const Text('Частицы кристалла образуют упорядоченную пространственную структуру в виде кристаллической решётки\n', textAlign: TextAlign.justify,),

          value: Qaz.q,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile<Qaz>(

          title: const Text('Основой строения кристалла является элементарная кристаллическая ячейка\n', textAlign: TextAlign.justify,),

          value: Qaz.a,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile(

          title: const Text('Всё сразу\n', textAlign: TextAlign.justify,),

          value: Qaz.e,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        TextButton(

          onPressed: \_myFunction, child: Text('Проверить'),),],);}}

class Ninth extends StatefulWidget {

  const Ninth({Key? key}) : super(key: key);

  @override

  State<Ninth> createState() => \_NinthState();

}

class \_NinthState extends State<Ninth> {

  Qaz \_qaz = Qaz.x;

  void \_myFunction() {

    if (\_qaz == Qaz.e){count+=1;}

  }

  @override

  Widget build(BuildContext context) {

    return Column(

      children: [

        Text('Что в строении жидкостей определяет их свойства?\n', textAlign: TextAlign.justify, style: TextStyle(fontSize: 17.0, fontWeight: FontWeight.w600)),

        RadioListTile(

          title: const Text('Расстояния между молекулами, соизмеримые с их собственными размерами\n', textAlign: TextAlign.justify,),

          value: Qaz.q,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile(

          title: const Text('Возможность относительно свободного перемещения молекул\n', textAlign: TextAlign.justify,),

          value: Qaz.e,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile(

          title: const Text('Всё сразу\n', textAlign: TextAlign.justify,),

          value: Qaz.a,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile(

          title: const Text('Ничего\n', textAlign: TextAlign.justify,),

          value: Qaz.z,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        TextButton(

          onPressed: \_myFunction, child: Text('Проверить'),),],);}}

class Tenth extends StatefulWidget {

  const Tenth({Key? key}) : super(key: key);

  @override

  State<Tenth> createState() => \_TenthState();

}

class \_TenthState extends State<Tenth> {

  Qaz \_qaz = Qaz.x;

  void \_myFunction() {

    if (\_qaz == Qaz.a){count+=1;}

    if (count == 0){count = -1;}

    add('molecular\_physics', count);

  }

  @override

  Widget build(BuildContext context) {

    return Column(

      children: [

        Text('Какой пар называют насыщенным?\n', textAlign: TextAlign.justify , style: TextStyle(fontSize: 17.0, fontWeight: FontWeight.w600)),

        RadioListTile(

          title: const Text('Пар, давление (плотность) которого меньше давления (плотности) насыщенного пара при той же '

        'температуре\n', textAlign: TextAlign.justify,),

          value: Qaz.q,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile<Qaz>(

          title: const Text('Пар, находящийся в состоянии динамического равновесия с жидкостью\n', textAlign: TextAlign.justify,),

          value: Qaz.a,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        RadioListTile(

          title: const Text('Правильного ответа нет\n', textAlign: TextAlign.justify,),

          value: Qaz.e,

          groupValue: \_qaz,

          onChanged: (Qaz? value) {

            setState(() {

              \_qaz = value!;});},),

        TextButton(

          onPressed: \_myFunction, child: Text('Проверить'),),],);}}

class Button extends StatelessWidget {

  const Button({Key? key}) : super(key: key);

  void qaz(context){

    Navigator.pushNamed(context, '/result');

  }

  @override

  Widget build(BuildContext context) {

    return Column(

      children: [

        ElevatedButton(onPressed: () {qaz;}, child: Text('Результат')),]);}}

class Results extends StatelessWidget {

  const Results({Key? key}) : super(key: key);

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(title: Text('Результат')),

      body: Center(

        child: Column(

          children: [

            Text('$count', style: TextStyle(fontSize: 30.0, fontWeight: FontWeight.w600)),

            ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/');}, child: Text('На главную')),],)));}}

class Clear extends StatefulWidget {

  const Clear({Key? key}) : super(key: key);

  @override

  State<Clear> createState() => \_ClearState();

}

class \_ClearState extends State<Clear> {

  @override

  Widget build(BuildContext context) {

    return Container(

      child: ElevatedButton(onPressed: () {count = 0;}, child: Text('Начать Тест')),);}}

class Test extends StatefulWidget {

  const Test({Key? key}) : super(key: key);

  @override

  State<Test> createState() => \_TestState();

}

class \_TestState extends State<Test> {

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(title: const Text('Тест'),),

      body: ListView(

        padding: const EdgeInsets.only(top: 15, left: 15, right: 15,),

        children: const [

          Clear(),

          First(),

          Second(),

          Thirt(),

          Fourth(),

          Fifen(),

          Sixth(),

          Seventh(),

          Eighth(),

          Ninth(),

          Tenth(),

          Button(),],),);}}

Текст модуля home.dart

import 'package:course/pages/ten/test.dart';

import 'package:flutter/material.dart';

import 'package:flutter/widgets.dart';

class Ten extends StatefulWidget {

  const Ten({Key? key}) : super(key: key);

  @override

  State<Ten> createState() => \_TenState();

}

class \_TenState extends State<Ten> {

  @override

  Widget build(BuildContext context) {

    return Center(

      child: Column(

      children: [

      ExpansionTile(

      title: Text('10 Класс'),

      children: [

            ExpansionTile(  //Молекулрная физика

              title: Text('Молекулярная физика'),

              children: [

                ElevatedButton(

                    onPressed: () {Navigator.pushNamed(context, '/molecular\_physics');},

                    child: Text('Основы молекулярно-кинетической теории')),

                ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/thermodynamics');}, child: Text('Основы термодинамики')),],),

            ExpansionTile(  //Электродинамика

              title: Text('Электродинамика'),

              children: [

                ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/electrostatics');}, child: Text('Электростатика')), //Молекулрная физика

                ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/dc\_electric\_current');}, child: Text('Постоянный электрический ток')),

                ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/magnetic\_field');}, child: Text('Магнитное поле. Электромагнитная индукция')),

                ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/electricity');}, child: Text('Электрический ток в разных средах')),],),],),

    ExpansionTile(

      title: Text('11 Класс'),

      children: [

        ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/mechanical\_vibrations\_and\_waves');}, child: Text('Механический колебания и волны')),

        ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/electromagnetic\_fluctuations\_and\_waves');}, child: Text('Электромагнитные колебания и волны')),

        ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/optics');}, child: Text('Оптика')),

        ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/theory\_of\_relativity');}, child: Text('Основы специальной теории относительности')),

        ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/photons');}, child: Text('Фотоны. Действия света')),

        ElevatedButton(onPressed: () {}, child: Text('Физика атома')),

        ElevatedButton(onPressed: () {}, child: Text('Ядерная физика и элементарные частицы')),],),],),);}}

class Tests extends StatelessWidget {

  const Tests({Key? key}) : super(key: key);

  @override

  Widget build(BuildContext context) {

    return Column(

      children: [

        ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/statistic');}, child: Text('Статистика')),

        ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/mol');}, child: Text('Основы молекулярной физики')),

        ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/mol');}, child: Text('Основы термодинамики')),

        ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/mol');}, child: Text('Электростатика')),

        ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/mol');}, child: Text('Постоянный электрический ток')),

        ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/mol');}, child: Text('Магнитное поле. Электромагнитная индукция')),

        ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/mol');}, child: Text('Электрический ток в разных средах')),

        ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/mol');}, child: Text('Механический колебания и волны')),

        ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/mol');}, child: Text('Оптика')),

        ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/mol');}, child: Text('Основы специальной теории относительности')),

        ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/mol');}, child: Text('Фотоны. Действия света')),

        ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/mol');}, child: Text('Физика атома')),

        ElevatedButton(onPressed: () {Navigator.pushNamed(context, '/mol');}, child: Text('Ядерная физика и элементарные частицы')),],);}}

class Home extends StatefulWidget {

  const Home({Key? key}) : super(key: key);

  @override

  State<Home> createState() => \_HomeState();

}

class \_HomeState extends State<Home> {

  int \_selectedIndex = 0;

  static const TextStyle optionStyle =

  TextStyle(fontSize: 30, fontWeight: FontWeight.bold);

  static const List<Widget> \_widgetOptions = <Widget>[

    Ten(),

    Tests(),];

  void \_onItemTapped(int index) {

    setState(() {

      \_selectedIndex = index;});}

    @override

    Widget build(BuildContext context) {

      return Scaffold(

        appBar: AppBar(

          title: Text('Физика'),

        ),

        body: Center(

          child: \_widgetOptions.elementAt(\_selectedIndex),

        ),

        bottomNavigationBar: BottomNavigationBar(

          items: const <BottomNavigationBarItem>[

            BottomNavigationBarItem(

              icon: Icon(Icons.thermostat),

              label: 'Теория',),

            BottomNavigationBarItem(

              icon: Icon(Icons.business),

              label: 'Тесты',),],

          currentIndex: \_selectedIndex,

          selectedItemColor: Colors.amber[800],

          onTap: \_onItemTapped,),);}}

Текст модуля statistic.dart

import 'package:flutter/material.dart';

class Statistic extends StatefulWidget {

  const Statistic({Key? key}) : super(key: key);

  @override

  State<Statistic> createState() => \_StatisticState();

}

class \_StatisticState extends State<Statistic> {

  var statistic = {

    'molecular\_physics': 0,

    'thermodynamics': 0,

    'electrostatics': 0,

    'dc\_electric\_current': 0,

    'magnetic\_field': 0,

    'electricity': 0,

    'mechanical\_vibrations\_and\_waves': 0,

    'electromagnetic\_fluctuations\_and\_waves': 0,

    'optics': 0,

    'theory\_of\_relativity': 0,

    'photons': 0,

    'atom\_physics': 0,

    'nuclear\_physics': 0,};

  void add(String, count){

    statistic[String] = count;}

  void \_show(){

    for (int i = 1; i<14; i++){

      if (key == -1){

        statistic[i] = 'Тест не пройден';}}}

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(title: Text('Статистика'),),

      body: Center(

        child:

          Column(

          children: [

            ElevatedButton(onPressed: () {\_show;}, child: Text('Показать статистику'),),

            Text('Основы молекулярной физики: ' + statistic[thermodynamics]),

            Text('Основы термодинамики: ' + statistic[electrostatics]),

            Text('Электростатика: ' + statistic[dc\_electric\_current]),

            Text('Постоянный электрический ток: ' + statistic[dc\_electric\_current]),

            Text('Магнитное поле. Электромагнитная индукция: ' + statistic[magnetic\_field]),

            Text('Электрический ток в разных средах: ' + statistic[electricity]),

            Text('Механические колебания и волны: ' + statistic[mechanical\_vibrations\_and\_waves]),

            Text('Оптика: ' + statistic[optics]),

            Text('Основы специальной теории относительности: ' + statistic[theory\_of\_relativity]),

            Text('Фотоны. Действие света: ' + statistic[photons]),

            Text('Физика атома: ' + statistic[atom\_physics]),

            Text('Ядерная физика и элементарные частицы: ' + statistic[nuclear\_physics]),

          ]),),);}}

ч