Geekbrains

**Разработка программного обеспечения для учащихся 9 и 11 классов для проведения государственных аттестаций.**

**Дипломный проект**

студента 1 года обучения

специальности “Программист”

Челпанов Н.Ю.

Новосибирск

2025

Содержание

[Введение 3](#_Toc198625049)

[Теоретическая часть 5](#_Toc198625050)

[Основные определения 5](#_Toc198625051)

[Создание меню для генерации теста 7](#_Toc198625052)

[Библиотека request 10](#_Toc198625053)

[Описание переменных в проекте 12](#_Toc198625054)

[Практическая реализация программы 13](#_Toc198625055)

[Таблица соответствия первичных баллов и баллов Итоговых аттестаций школьников 21](#_Toc198625056)

[Типы заданий на примере ЕГЭ по математики профильный уровень 26](#_Toc198625057)

[Технические характеристики компьютера для проведения экзамена 28](#_Toc198625058)

[Безопасность и экологичность проекта 29](#_Toc198625059)

[Анализ и оценка вредных факторов при разработке САКЗ 29](#_Toc198625060)

[Планы на будущее. 42](#_Toc198625061)

[ЗАКЛЮЧЕНИЕ 43](#_Toc198625062)

[Список литературы 45](#_Toc198625063)

[Приложение А 46](#_Toc198625064)

[Приложение B 53](#_Toc198625065)

[Приложение С 54](#_Toc198625066)

## **Введение**

В наше время для оптимизации работы всё больше и больше используются программы, и они написаны на самых различных языках программирования, что позволяет решать широкий спектр задач. Существует множество языков программирования, таких как Python, Java, C# и еще многие другие языки, каждый из которых имеет свои особенности и области применения. В данной разработке используется язык Python, который был выбран как наиболее широко известный популярный среди разработчиков благодаря своей простоте и мощным библиотекам.

В качестве сферы, в которой будет использоваться разработанная программа, выбрано образование, поскольку в современном мире тесты до сих пор печатаются на бумаге, а школьников становится всё больше. А как известно, бумага – ценный ресурс, поэтому использование цифровых технологий в образовании становится всё более актуальным и необходимым.

Таким образом, целью данной работы является создание инновационной программы для организации проведения итоговых аттестаций у школьников 9 и 11 классов, что поможет оптимизировать процесс оценки знаний.

Выделим основной план работы над проектом: поиск источника информации для использования в программе (в качестве такового будет выступать сайт открытого банка задач ФИПИ), выбор языка программирования (что уже было произведено на самом начальном этапе), поиск библиотек, которые будут использоваться для реализации функциональности, создание алгоритма программы, а также непосредственное создание самой программы, создание базы данных для хранения результатов тестирования, и, наконец, взаимодействие между программой и базой данных.

Данная работа была выполнена в команде из одного человека, который исполнял роль программиста. В рамках выполнения проекта также были проведены тестирование программного продукта и управление проектом – проджект-менеджмент. Эта работа будет рассмотрена с точки зрения программиста, охватывая все аспекты разработки.

Практическая значимость проекта заключается в том, что помимо использования приложения на экзаменах, его также можно использовать и для подготовки к экзаменам. Это связано с тем, что вариант, генерируемый программой, будет уникальным благодаря большой выборке заданий из банка задач. Кроме того, существует возможность модернизировать программу, интегрировав различные сайты для подготовки и реальной сдачи экзаменов. Однако главной проблемой остаётся необходимость использования однотипных сайтов, создание которых не было включено в рамки данного проекта, что может являться предметом дальнейшего исследования.

# **Теоретическая часть**

# **Основные определения**

Программирование — это сложный и многогранный процесс создания и модификации компьютерных программ, который требует определенных навыков и знаний.

Python — это высокоуровневый язык программирования, который отличается своей эффективностью, простотой и универсальностью использования, что делает его особенно популярным среди разработчиков.

Он широко применяется в разработке веб-приложений и прикладного программного обеспечения, а также в таких сложных и интересных областях, как машинное обучение и обработка больших данных, что расширяет его возможности.

HTML (HyperText Markup Language) — это язык гипертекстовой разметки, который используется для создания веб-страниц и оформления их содержания

Он позволяет описывать структуру и содержание веб-страницы, такие как заголовки, абзацы, списки, картинки и ссылки, что значительно упрощает жизнь веб-разработчикам.

Основная цель HTML — это структурировать и оформлять контент на сайте, что является ключевым моментом в веб-дизайне. HTML создаёт иерархическую структуру веб-страницы, используя заголовки, абзацы, списки и таблицы, что обеспечивает удобство восприятия информации. Такая структура, безусловно, помогает пользователю легче ориентироваться на сайте и находить нужную информацию.

База данных – это совокупность данных, хранимых в соответствии со схемой данных, манипулирование которыми выполняют в соответствии с правилами средств моделирования данных и стандартами.

Реляционная база данных — это тип базы данных, основанный на реляционной модели данных, который используется для организации и хранения информации.

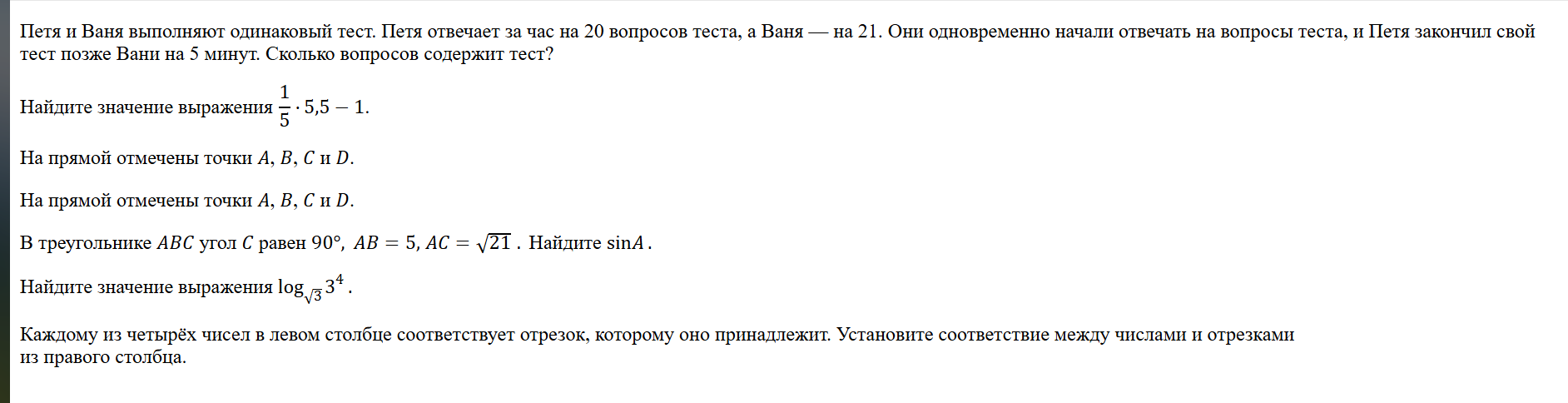
В такой базе данные структурированы в виде таблиц, которые содержат строки и столбцы. Каждая строка таблицы представляет собой набор связанных значений, относящихся к одному объекту или сущности, что делает управление данными более удобным и эффективным.

Теперь рассмотрим из каких частей должна состоять программа для генерации тестов и в самом конце какие пункты удалось сделать.

Структура программы

1. **Основная часть** – в ней происходит генерация варианта, и эта часть программы написана на языке программирования Python, который отлично подходит для решения таких задач.
2. **Вариант -** это сгенерированная html-страница, которая формируется в результате работы основной части программы и отображает весь необходимый контент.
3. **База данных**, которая написана на языке MySQL, служит для хранения всех необходимых данных и позволяет эффективно управлять информацией.

Генерация программы происходит через консоль, что обеспечивает удобный интерфейс для пользователя. Основной процесс генерации состоит из парсинга html-страницы из банка вариантов, что является ключевым этапом. В банке вариантов есть варианты, которые имеют уникальный шифр, и этот шифр мы используем в программе для каждого вопроса рандомным образом. Тем самым, благодаря этому подходу, получается уникальный вариант, не похожий на другие, что значительно повышает качество создания тестовых заданий и делает их более разнообразными и интересными.



Теперь рассмотрим, какие библиотеки были использованы в проекте, а так же рассмотрим необходимость их использования.

# **Создание меню для генерации теста**

В данном, проекте было реализовано полноценное меню с помощью разработанной библиотеки Tkinter. В современном программировании многие программы на сегодняшний день активно используют графический интерфейс, который, несомненно, является более интуитивным и удобным для пользователя, чем привычная консоль. И с помощью языка программирования Python также можно с легкостью создавать различные графические программы. Для этой цели в Python по умолчанию применяется специальный тулкит — набор компонентов, который называется tkinter. tkinter доступен в виде отдельного встроенного модуля, который включает в себя все необходимые графические компоненты — кнопки, текстовые поля и множество других элементов, которые делают интерфейс более дружелюбным.

По сути, Tkinter представляет собой интерфейс в Python для графической библиотеки Tk (собственно само название "Tkinter" является сокращением "Tk interface"). Первоначально данная библиотека разрабатывалась для языка Tcl — её создал в 1988 году талантливый Джон Остерхаут (John Ousterhout), профессор computer science из Беркли, для создания графических приложений для своего языка Tcl. Однако впоследствии Tk была адаптирована для широкого ряда динамических языков, в частности, для Ruby, Perl и, естественно, для языка Python, что произошло в 1994 году. И на сегодняшний день и библиотека Tk, и сам тулкит tkinter доступны для большинства популярных операционных систем, включая Mac OS, Linux и Windows, что делает их невероятно универсальными.

\*\*Преимущества Tkinter:\*\*

Данный тулкит по умолчанию включен в стандартную библиотеку языка Python в виде отдельного модуля, поэтому пользователю не потребуется что-то дополнительно устанавливать, что, безусловно, является удобным.

Tkinter также кроссплатформенный — один и тот же код будет работать одинаково на различных платформах (Mac OS, Linux и Windows), что значительно упрощает процесс разработки.

Тулкит Tkinter легко изучать: сам тулкит, хотя и содержит некоторый готовый код, виджеты и графические элементы, однако при этом он довольно лаконичен и прост, что делает его доступным для начинающих программистов.

Кроме того, Tk распространяется по BSD-лицензии, что означает, что библиотека может быть использована как в опенсорсных проектах, так и в коммерческих наработках, что расширяет её применение.

Чтобы создать меню, нужно выполнить следующие шаги:

- Создать экземпляр меню от класса Menu.

- Привязать его к виджету, на котором оно будет удобно расположено. Обычно таковым выступает главное окно приложения.

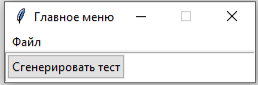
- Добавить пункты меню с помощью метода add\_command. К ним можно добавить опцию command, связав тем самым с какой-либо функцией-обработчиком клика, чтобы обеспечить интерактивность.

- Создать новые экземпляры Menu и подвязать их к главному меню с помощью метода add\_cascade, что позволит создать многоуровневое меню.

Некоторые методы, которые можно использовать в Tkinter, включают:

- `add\_separator` — этот метод добавляет линию-разделитель в меню, которая используется для визуального разделения групп команд, что делает интерфейс более понятным.

- `post` — позволяет создать всплывающее меню, также известное как контекстное меню (если настроить его появление по клику правой кнопкой мыши), что добавляет дополнительный уровень взаимодействия для пользователя.



# **Библиотека request**

Requests — это многофункциональная и весьма известная библиотека для работы с HTTP-запросами в языке программирования Python. Она предоставляет разработчикам мощный и удобный интерфейс для выполнения разнообразных операций с REST API, загрузки различных данных с веб-страниц, а также несомненно, для отправки файлов и оформления процесса авторизации.

Некоторые, можно сказать, ключевые возможности и особенности библиотеки Requests:

- поддержка постоянного HTTP-соединения и его повторное, если это необходимо, использование;

- применение разнообразных международных и даже национальных доменов;

- использование Cookie: передача и получение значений в формате «ключ: значение», что делает работу с сессиями более комфортной;

- автоматическое декодирование контента, что экономит время и усилия разработчиков;

- SSL-верификация для обеспечения безопасности передаваемой информации;

- аутентификация пользователей на большинстве популярных ресурсов с сохранением, что значительно упрощает процесс взаимодействия с ними;

- поддержка прокси-серверов при необходимости, что дает дополнительные возможности для управления сетевым трафиком;

- загрузка и выгрузка файлов для удобного обмена данными;

- стриминговые загрузки и фрагментированные запросы для оптимизации передачи больших объемов информации;

- задержки соединений, которые могут быть полезны для управления нагрузкой;

- передача требуемых заголовков на web-ресурсы и множество других функций, которые делают библиотеку очень гибкой и полезной.

Стоит отметить, что библиотека Requests не поставляется вместе с самим Python, поэтому, перед её использованием, обязательно нужно установить её с помощью менеджера пакетов, например, pip, что достаточно просто и интуитивно.

Некоторые из методов, предоставляемых библиотекой Requests, включают:

- `requests.get()` — этот метод отправляет GET-запрос для получения данных с сервера, что является одним из самых распространенных способов обращения к API;

- `requests.post()` — с помощью этого метода отправляется POST-запрос, предназначенный для отправки данных на сервер, что позволяет создавать новые записи или изменять существующие;

- `requests.put()` — этот метод отправляет PUT-запрос для обновления данных на сервере, что делает его удобным для редактирования информации;

- `requests.delete()` — с помощью этого метода осуществляется отправка DELETE-запроса для удаления данных с сервера, что завершает работу с ненужной информацией.

# **Описание переменных в проекте**

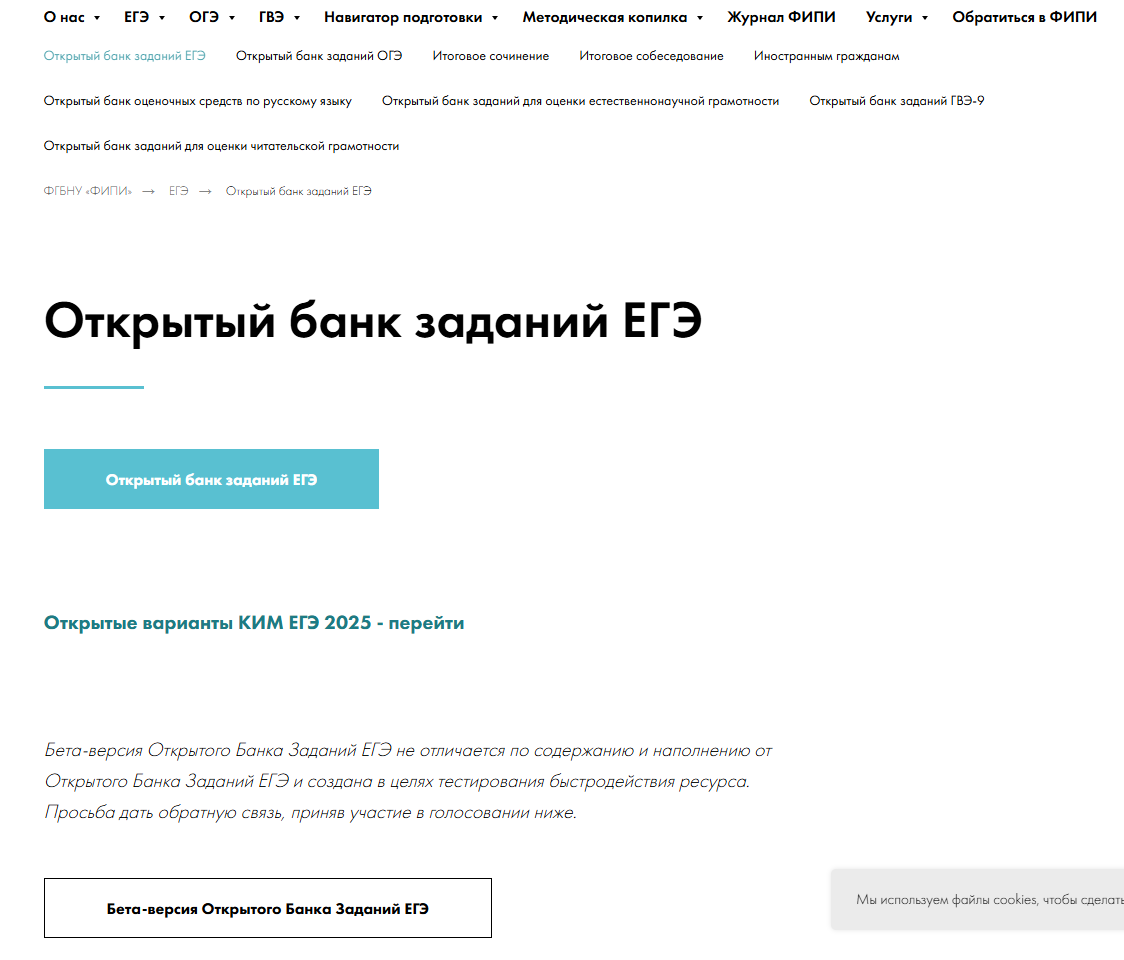
В рамках данного проекта было использовано множество переменных, каждая из которых играет свою уникальную роль в функционировании программы. Перечислим все переменные и их описание, чтобы обеспечить полное понимание кода:

* **list** – количество страниц с данными на сайте ФИПИ, что позволяет программе эффективно управлять количеством запрашиваемой информации.
* **sections\_array** – массив разделов задач, который содержит структурированные данные о различных категориях вопросов, что облегчает навигацию и выбор заданий.
* **number** – уникальный строковый id варианта, который используется для идентификации конкретного теста или набора вопросов, что важно для отслеживания результатов и статистики.
* **finder** – строка, которая может использоваться для поиска определённых данных или выполнения обработок внутри программы в зависимости от логики приложения.
* **link\_one** – динамически изменяемая ссылка, которая формируется в зависимости от значений переменных **list** и **sections\_array**, обеспечивая гибкость в запросах к удалённым ресурсам.
* **req** – переменная, хранящая результат request-запроса, который отправляется на сервер для получения данных о задачах, играя ключевую роль в взаимодействии с API.
* **src** – текст ответа, полученного от сервера, который содержит информацию о задачах и результатах запроса, так как необходимо извлекать и обрабатывать эти данные для дальнейшего использования.

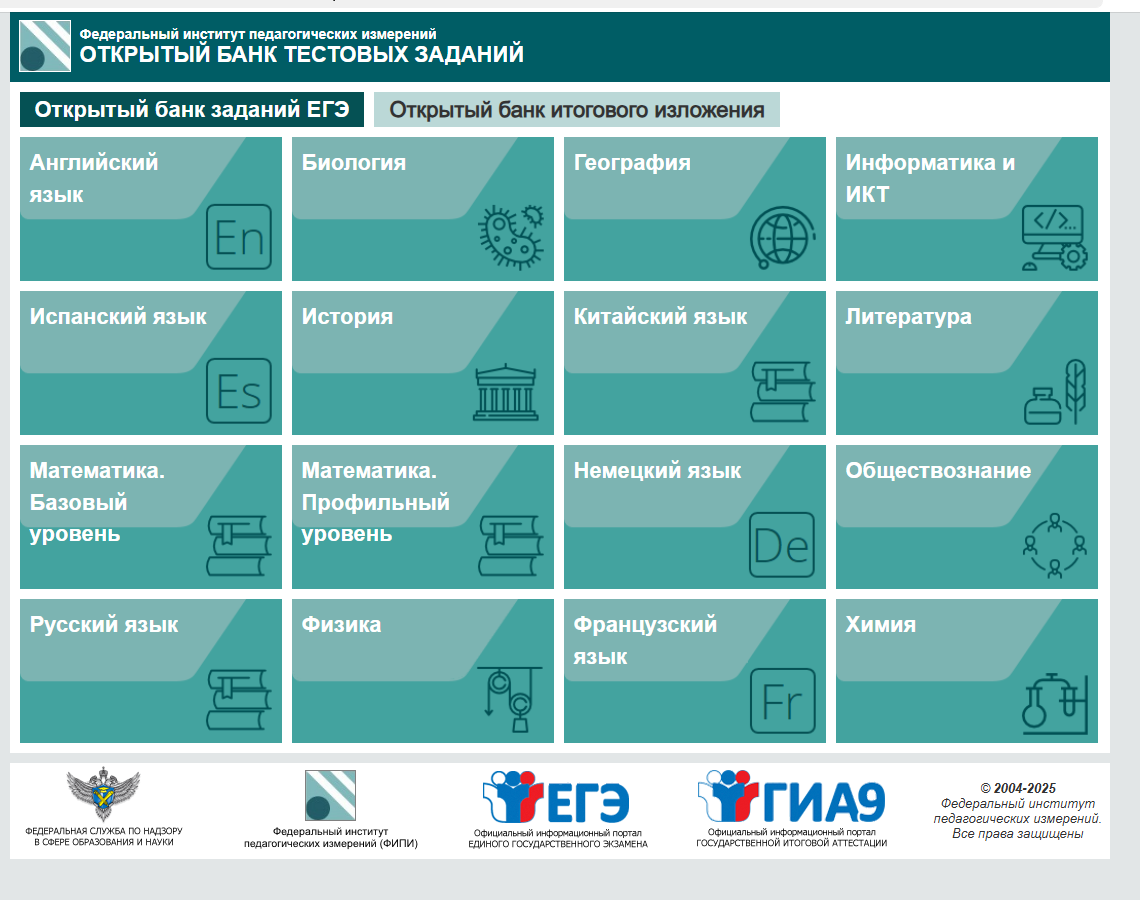
Каждая из этих переменных была тщательно продумана и внедрена в проект, что позволило добиться необходимой функциональности и эффективности приложения.

# **Практическая реализация программы**

Для реализации данной программы необходимо зайти на сайт Открытого банка заданий ФИПИ, в нашем примере, на сайт по материалам ЕГЭ: <https://fipi.ru/ege/otkrytyy-bank-zadaniy-ege>

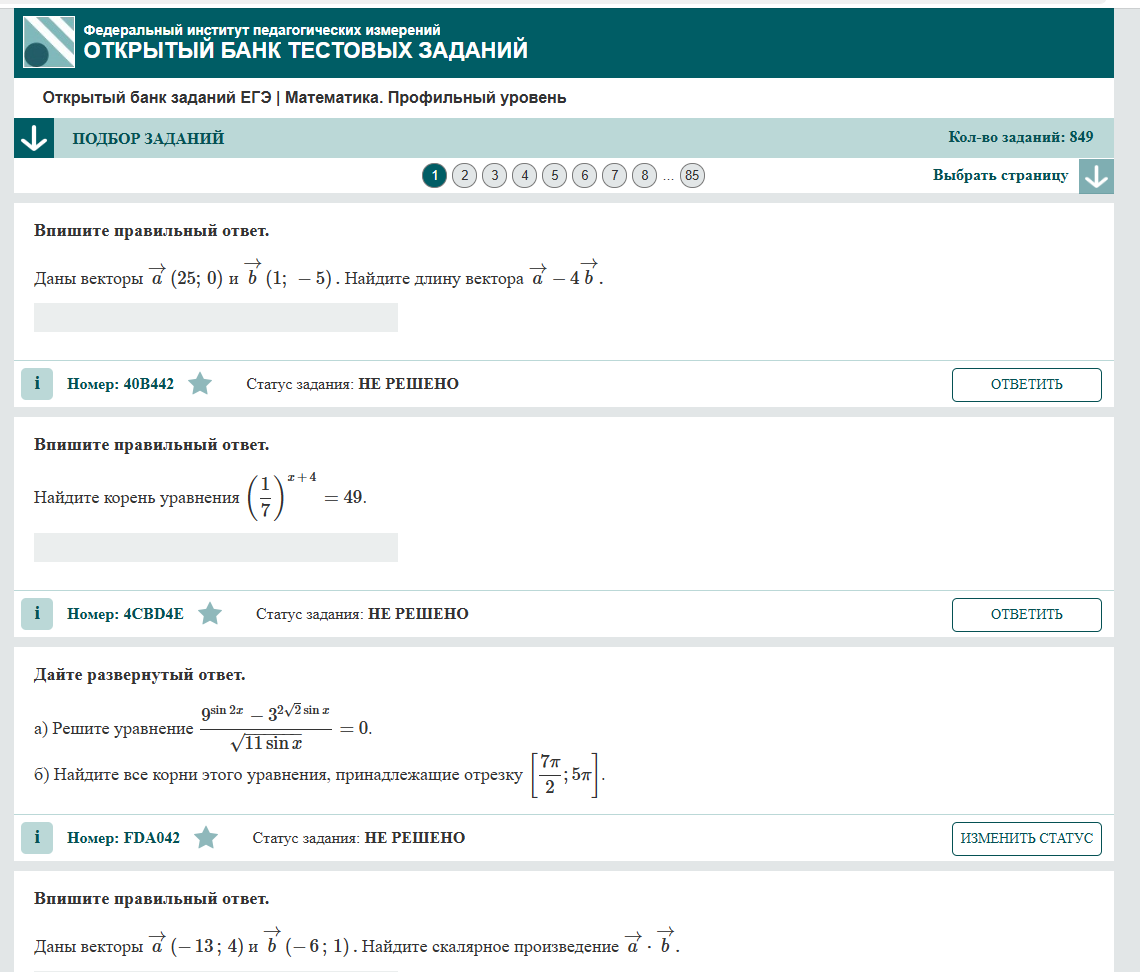


Затем нажимаем на кнопку Открытый банк заданий ЕГЭ, открывается страница со всеми предметами в рамках ЕГЭ:



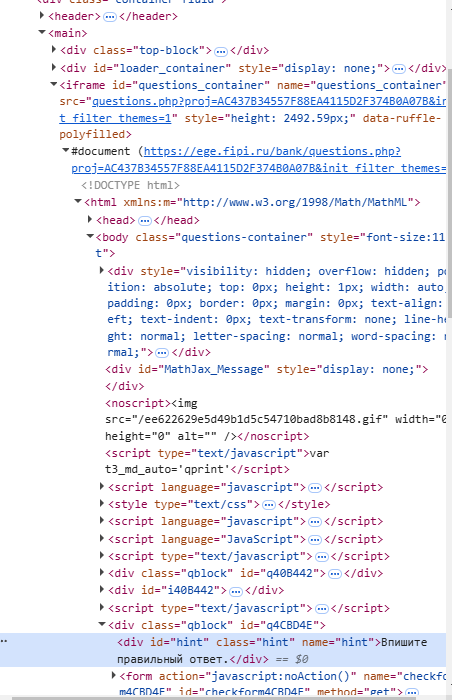
Затем, как в нашем примере, открываем раздел «Математика. Профильный уровень»

Как мы видим, у нас открылась вкладка с заданиями:

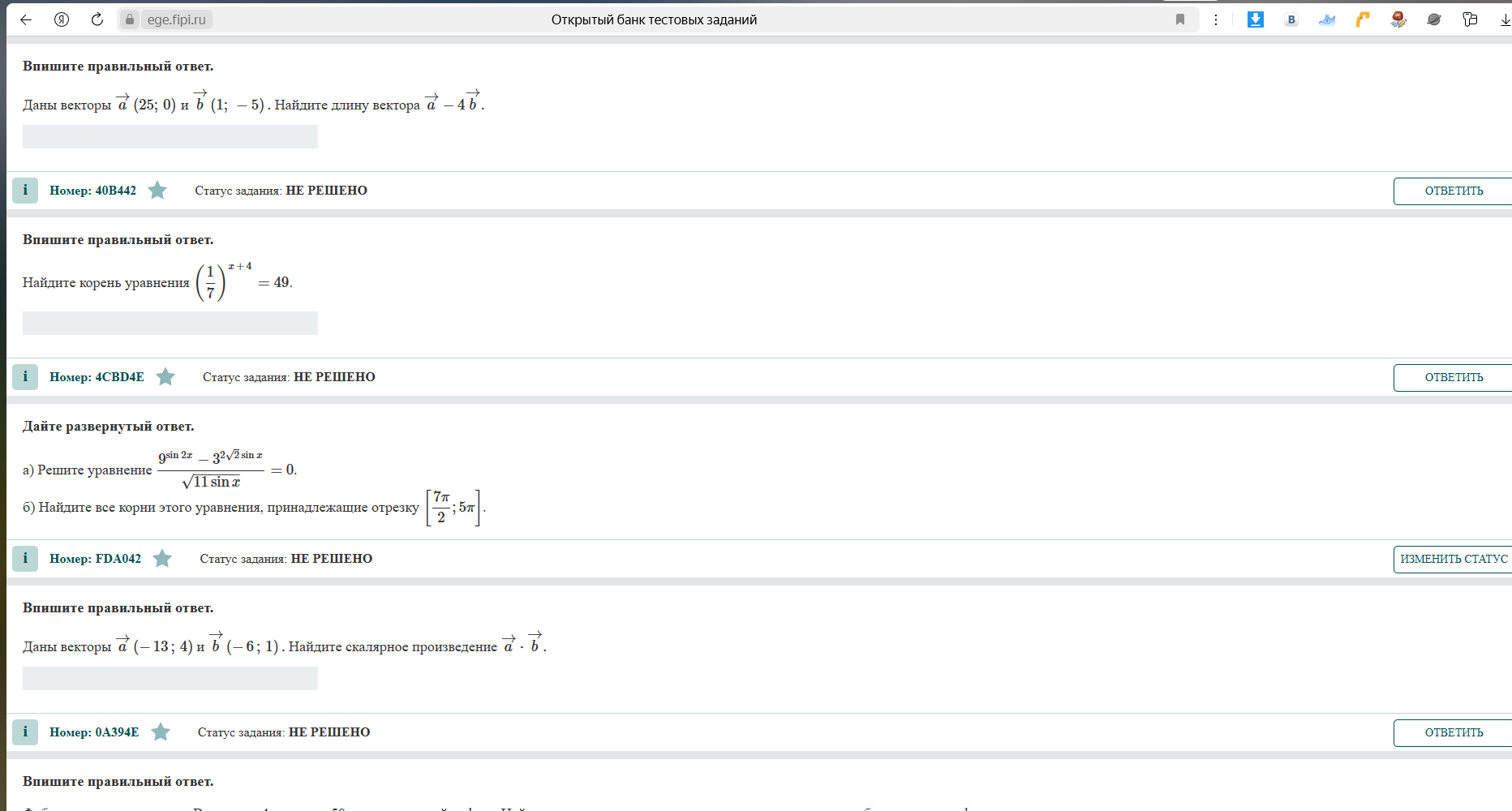


Как видим, у каждого задания есть номер, он потребуется нам в дальнейшем.

Щелкаем правой кнопкой мыши по браузеру (в данном примере используется Яндекс браузер), затем, исследовать элемент. Открывается панель разработчика:

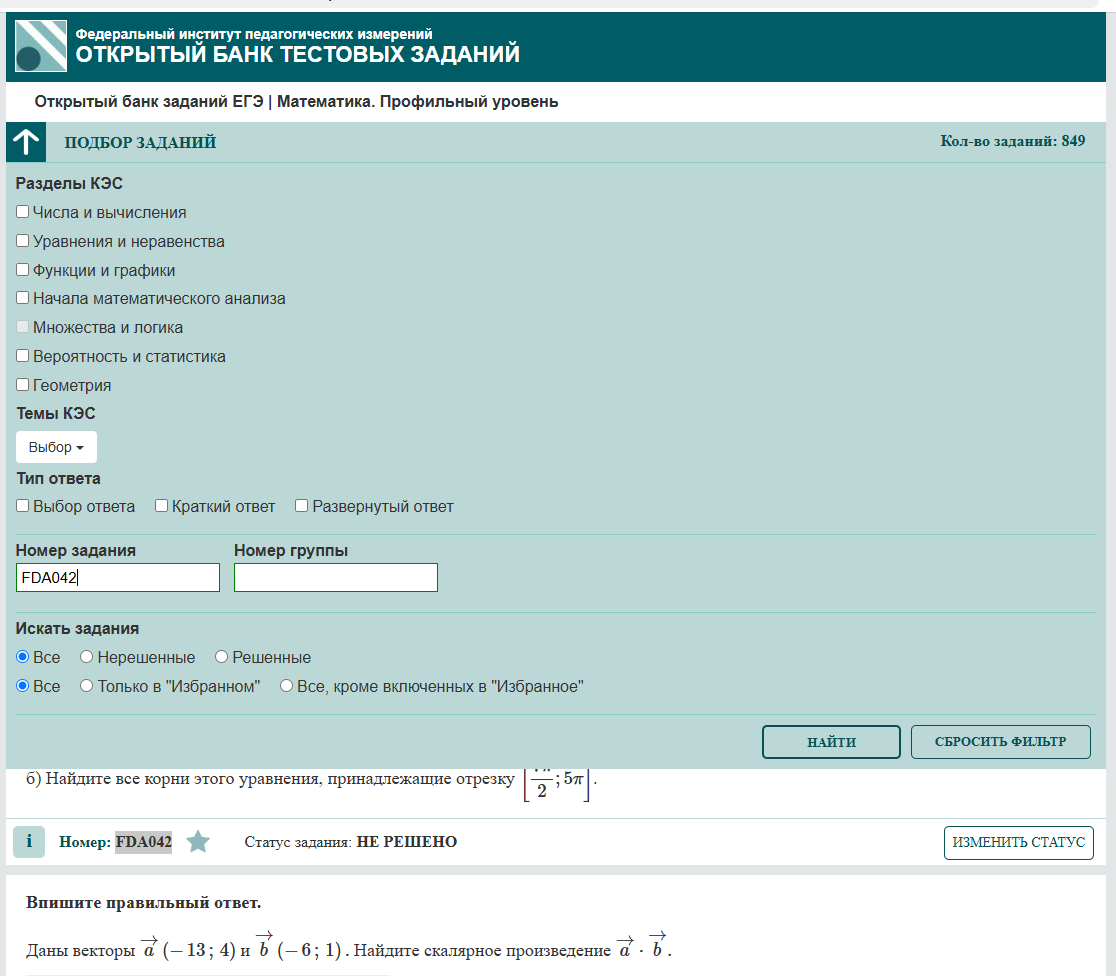


Щелкаем правой кнопкой по ссылке с тегом #document (в нашем случае ссылка https://ege.fipi.ru/bank/questions.php?proj=AC437B34557F88EA4115D2F374B0A07B&init\_filter\_themes=1) и открыть в новой вкладке, открываются все задания:



Копируем ссылку на данный раздел, она потребуется в нашей программе.

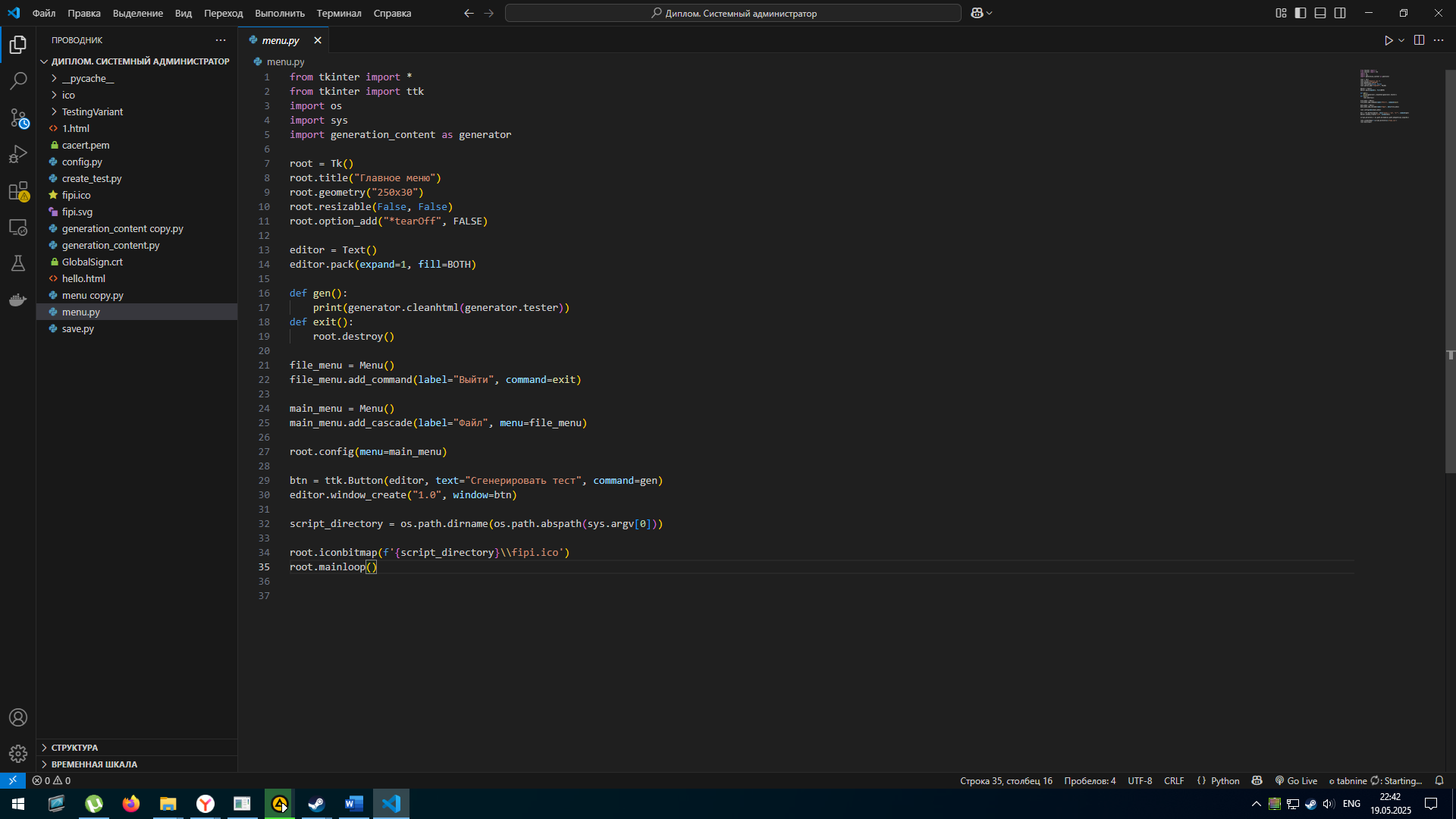
Возвращаемся к обычному окошку сайта, фильтруем в разделе Подбор задания задание по номеру, вводя номер любого задания и нажимая найти:



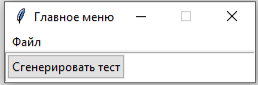
Повторяем пункт с исходным кодом, запоминаем исходную ссылку с одним заданием.

Так же делаем и с разделами, сохраняя полученные ссылки

Приступаем к написанию кода. Открываем Visual Studio Code и создаем python – файлы проекта:

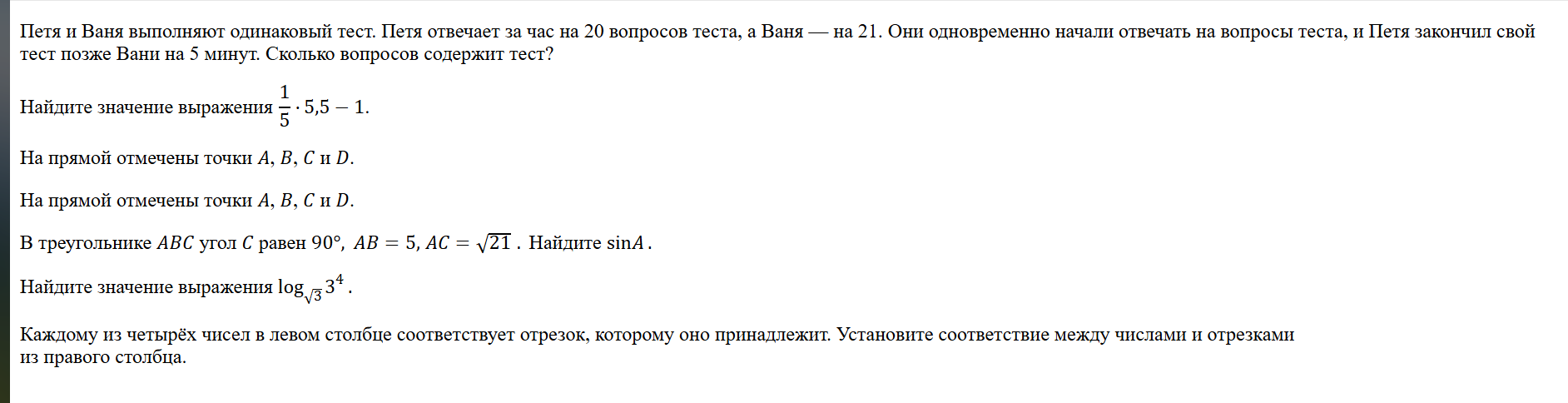


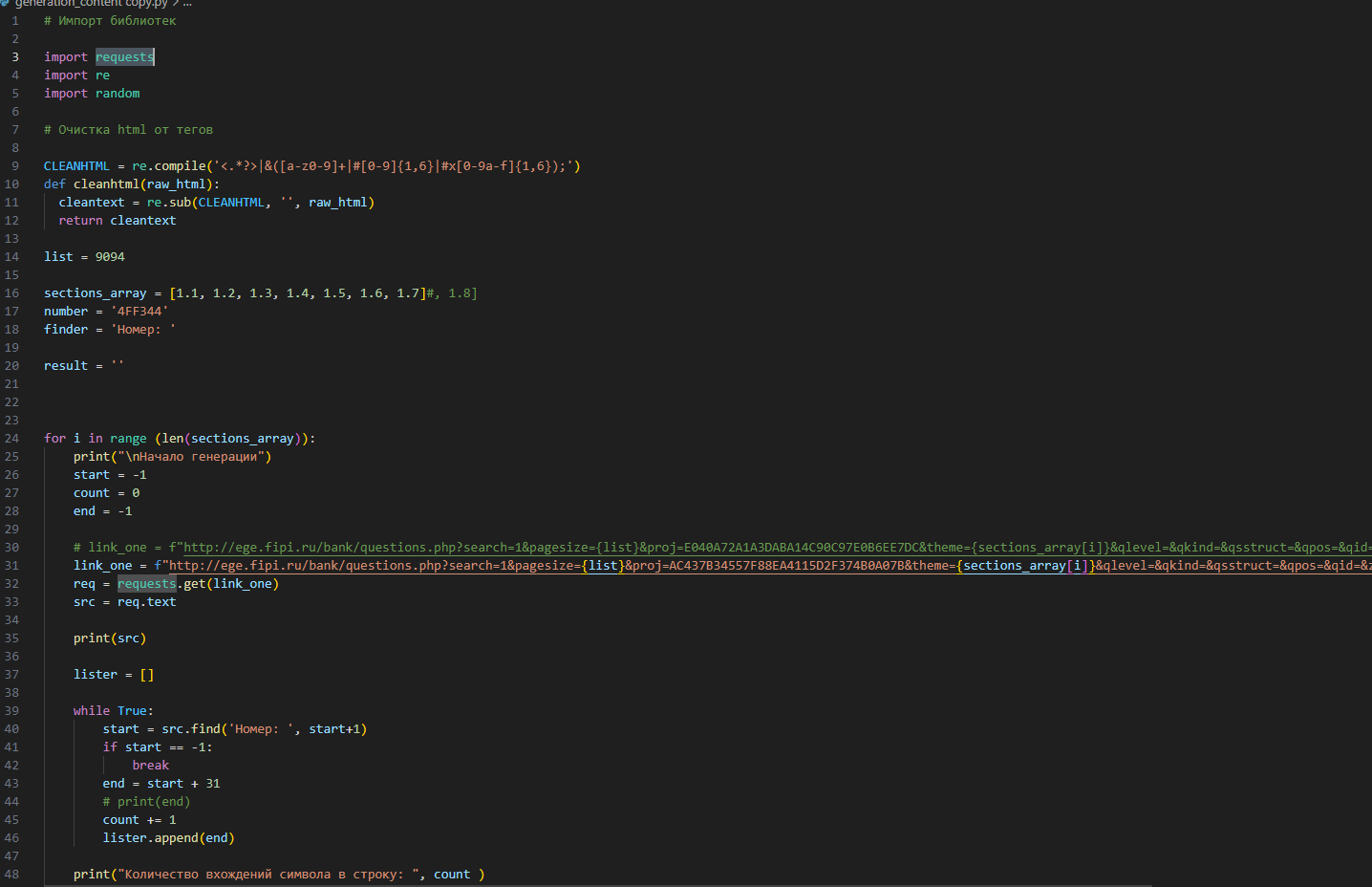
Создаем с помощью библиотеки tkinter меню:



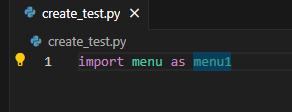
Создаем класс для генерации вариантов, который состоит из:

* импорт нужных библиотек requests, random, re
* функций генерации, очистки html от тегов и сохранении результатов в html-файл:





Создаём файл для запуска программы:



Исходный код программ можно найти в Приложении А.

Запускаем программу, нажимаем в меню Сгенерировать тест и смотрим результат в директории с программой.

К сожалению, пока подсчёт результатов и база данных в данном дипломе не реализована.

# **Таблица соответствия первичных баллов и баллов Итоговых аттестаций школьников**

Для того, чтобы узнать, какой балл по ЕГЭ и ОГЭ получат школьники, необходимо предварительно знать таблицу перевода первичных баллов за задания в итоговые баллы по ЕГЭ. Эта важная информация представлена в приложении С, где вы сможете найти таблицу перевода первичных баллов в итоговые, что помогает в понимании системы оценивания.  
Максимальный первичный балл в 2024 году составляет 32 балла, что, соответственно, эквивалентно 100 тестовым баллам, позволяя ученикам добиться значительных результатов.  
Следует отметить, что максимальный первичный балл меняется каждый год, что обусловлено обновлениями в заданиях и требованиях. Однако максимальный тестовый балл не меняется никогда и всегда остается на уровне – это 100 баллов, что обеспечивает стабильность и предсказуемость в системе оценки успеваемости школьников.

Минимальные баллы ЕГЭ 2025

[Минимальные баллы](https://ctege.info/ege-2025/minimalnyie-prohodnyie-ballyi-ege-2025.html)  в 2024 году изменились, в 2021 году их повышали, в 2022, 2023, 2025  не менялись.

**Обязательные предметы:**

русскому языку – 36 баллов; 24 - для получения аттестата

математике – 27 баллов; 27 - для получения аттестата

**По выбору предметы:**

биологии – 36 баллов

географии – 37 баллов

иностранному языку – 22 баллов

информатике – 40 балла

истории – 32 баллов

литературе – 32 баллов

обществознанию – 42 баллов (в 2020 году минимум был равен 45 баллам);

физике – 36 баллов

химии – 36 баллов

Шкала перевода баллов ЕГЭ 2025 по предметам

К таким предметам относятся:

**русский язык** – максимальный первичный балл изменён с 54 до **50 баллов**;

**математика (профиль)** – максимальный первичный балл увеличен с 31 до **32 баллов**;

**литература**– максимальный первичный балл работы изменён с 53 до **48 баллов**;

**физика**– максимальный первичный балл изменён с 54 до **45 баллов**;

**биология**– максимальный первичный балл изменён с 59 до **57 баллов**;

**география**– максимальный первичный балл изменён с 43 до **38 баллов**;

**иностранные языки** (английский, немецкий, французский, испанский) – максимальный первичный балл изменён с 86 до **82 баллов/**

Шкала перевода баллов в оценку ОГЭ–2025

Шкала перевода первичных баллов в итоговые отметки по пятибалльной системе используется для выставления оценок по результатам ОГЭ. Кроме того, образовательные учреждения могут использовать рекомендованные минимальные баллы при отборе учеников в профильные 10-е классы. Шкала ежегодно утверждается Рособрнадзором. В 2025 году она зафиксирована в письме министерства от 13.02.2025.

Теперь, чтобы получить «хорошо» и «отлично» по русскому языку, выпускникам придется набрать на 3 балла больше, чем в прошлом году. Также увеличили минимальный балл за грамотность с 6 до 9. Он влияет на получении отличной отметки за экзамен. В остальных предметах изменения незначительные.

Русский язык

«2»: 0–14

«3»: 15–25

«4»: 26–32 (не менее 6 баллов за грамотность по критериям ГК1–ГК4, иначе ставится «3»)

«5»: 33–37 (не менее 9 баллов за грамотность по критериям ГК1–ГК4, иначе ставится «4»)

Рекомендуемый балл для отбора в профильные классы – 28.

Математика

«2»: 0–7

«3»: 8–14 (не менее 2 баллов за геометрию, иначе ставится «2»)

«4»: 15–21 (не менее 2 баллов за геометрию, иначе ставится «2»)

«5»: 22–31 (не менее 2 баллов за геометрию, иначе ставится «2»)

Рекомендуемый балл для отбора в профильные классы:

для естественнонаучного профиля: 18 баллов, из них не менее 6 баллов по геометрии;

для экономического профиля: 18 баллов, из них не менее 5 баллов по геометрии;

для физико-математического профиля: 19 баллов, из них не менее 7 баллов по геометрии.

Физика

«2»: 0–9

«3»: 10–19

«4»: 20–29

«5»: 30–39

Рекомендуемый балл для отбора в профильные классы – 26.

Химия

«2»: 0–9

«3»: 10–20

«4»: 21–30

«5»: 31–38

Рекомендуемый балл для отбора в профильные классы – 27.

Биология

«2»: 0–12

«3»: 13–25

«4»: 26–37

«5»: 38–47

Рекомендуемый балл для отбора в профильные классы – 33.

География

«2»: 0–11

«3»: 12–18

«4»: 19–25

«5»: 26–31

Рекомендуемый балл для отбора в профильные классы – 23.

Обществознание

«2»: 0–13

«3»: 14–23

«4»: 24–31

«5»: 32–37

Рекомендуемый балл для отбора в профильные классы – 29.

История

«2»: 0–10

«3»: 11–20

«4»: 21–29

«5»: 30–37

Рекомендуемый балл для отбора в профильные классы – 26.

Литература

«2»: 0–15

«3»: 16–23

«4»: 24–31

«5»: 32–37

Рекомендуемый балл для отбора в профильные классы – 27.

Информатика

«2»: 0–4

«3»: 5–10

«4»: 11–16

«5»: 17–21

Рекомендуемый балл для отбора в профильные классы – 15.

Иностранный язык

«2»: 0–28

«3»: 29–45

«4»: 46–57

«5»: 58–68

Рекомендуемый балл для отбора в профильные классы – 55.

# **Типы заданий на примере ЕГЭ по математики профильный уровень**

ЕГЭ по математике профильного уровня включает в себя разнообразные задания, проверяющие знания и умения в различных областях математики. Некоторые из этих типов заданий включают:

- \*\*Алгебра и преобразования\*\*: здесь рассматриваются задачи, связанные с упрощением выражений и разложением на множители, что помогает развивать навыки работы с алгебраическими выражениями. 2

- \*\*Функции и их графики\*\*: задания этой категории включают в себя анализ свойств функций, построение их графиков и исследование поведения различных типов функций, что является важной частью математического анализа. 2

- \*\*Тригонометрия\*\*: в этой секции представлены тригонометрические функции, а также уравнения и неравенства, которые требуют от учащихся хорошего понимания тригонометрических свойств и их применения. 2

- \*\*Логарифмы и показательные функции\*\*: задания заключаются в решении логарифмических и показательных уравнений, а также в изучении свойств логарифмов, что помогает укрепить знания о степени и логарифме. 2

- \*\*Геометрия на плоскости и в пространстве\*\*: здесь рассматриваются вопросы планиметрии, характеристики фигур, а также вычисление площадей и объёмов, что имеет прикладное значение в реальной жизни. 2

- \*\*Теория вероятностей и статистика\*\*: эта категория заданий ориентирована на вычисление вероятностей и ознакомление с основами комбинаторики, что особенно актуально в современных условиях анализа данных. 2

- \*\*Математический анализ\*\*: в этой области рассматриваются темы производной, первообразной и интеграла, что является важной базой для более глубокого изучения математического анализа. 2

- \*\*Стереометрия\*\*: здесь акцент делается на свойства объёмных фигур, а также расчёт площадей поверхностей и объёмов, что имеет большое значение в архитектуре и инженерии. 2

- \*\*Уравнения и неравенства\*\*: задания этой группы включают решения различных типов уравнений и неравенств, что является важным элементом математических навыков. 2

- \*\*Последовательности и прогрессии\*\*: в этой секции рассматриваются арифметические и геометрические прогрессии, а также формулы для нахождения общего члена и суммы, что помогает развивать логическое мышление. 2

ЕГЭ по математике профильного уровня состоит из двух частей: \*\*часть 1\*\* — с кратким ответом, где требуется дать ответ на поставленные вопросы, и \*\*часть 2\*\* — с развёрнутым ответом, в которой присутствует необходимость в подробном решении задач и обосновании сделанных выводов. 5

Таким образом, данный экзамен охватывает широкий спектр математических знаний и умений, что делает его важным этапом в образовательном процессе.

# **Технические характеристики компьютера для проведения экзамена**

Для простых сценариев \*\*на Python\*\* и веб-разработки, которые не требуют высоких мощностей и ресурсов, рекомендуются следующие характеристики:

- \*\*Процессор\*\*: рекомендуется выбирать двухъядерный процессор, с частотой 2,0 ГГц или выше, что позволит обеспечить приемлемую скорость обработки данных и выполнения задач, связанных с программированием. 1

- \*\*Оперативная память\*\*: оптимальным вариантом будет наличие 4 ГБ ОЗУ, что достаточно для запуска базовых приложений и выполнения простых скриптов, позволяя работать без значительных затруднений. 1

- \*\*Хранилище\*\*: необходим жёсткий диск объёмом 50 ГБ или SSD, так как это обеспечит достаточный объем для хранения операционной системы, программного обеспечения и необходимых файлов. 1

- \*\*Операционная система\*\*: следует использовать одну из современных операционных систем, таких как Windows, macOS или Linux, в зависимости от предпочтений пользователя и специфики задач.

Эти характеристики обеспечат стабильную работу устройства, что крайне важно для успешного прохождения экзамена и выполнения заданий.

# **Безопасность и экологичность проекта**

# **Анализ и оценка вредных факторов при разработке САКЗ**

Данная работа нацелена на помощь в контроле результатов учебного процесса. В результате внедрения разработанной системы автоматизированного контроля знаний (далее САКЗ) ожидается повышение качества и скорости проверки знаний студентов, а также облегчение труда преподавательского состава.

В настоящее время нашли широкое применение персональные компьютеры (далее ПК). В данном проекте использование ПК обязательно, т.к. запуск САКЗ и непосредственная работа с ней будет не возможна без ПК. Но компьютеры не только облегчают всем нам жизнь, но и при длительном использовании источник профессиональных заболеваний. Это предъявляет к каждому пользователю ПК требования - знать о вредном воздействии машин на организм человека и необходимых мерах защиты, чтобы сохранить здоровье и производительно работать на компьютере в течение рабочего дня.

В процессе работы человек вступает во взаимодействие с различными предметами, устройствами, другими людьми. Кроме этого на него воздействуют параметры производственной обстановки (температура, влажность, подвижность воздуха, шум и т.д.). Это характеризует определенные условия, в которых протекает работа человека. От условий работы в большой степени зависят здоровье и работоспособность человека, его отношение к работе.

При плохих условиях резко снижается производительность, и создаются предпосылки для возникновения травматизма и профессиональных заболеваний.

Производственный фактор называется опасным, если его воздействие на работающего в определенных условиях приводит к травматизму или другому резкому ухудшению здоровья человека в предельно короткий срок. Если же производственный фактор приводит к снижению работоспособности и как следствие, к заболеванию, то его считают вредным [6].

Система стандартов безопасности труда [7] содержит классификацию элементов условий труда, выступающих в роли опасных и вредных производственных факторов. Они подразделяются на 4 группы:

физические;

химические;

биологические;

психофизические.

В Табл. 8 приведены опасные и вредные факторы, влияющие на работников в служебных помещениях. Целью нашей работы является устранение или уменьшение воздействия вредных и опасных факторов на здоровье работников.

Табл. 8. Опасные и вредные факторы

|  |  |
| --- | --- |
| Факторы | Действие на человека |
| Высокое напряжение тока в электросети | Электротравмы (ожоги) |
| Вредные вещества | Общетоксическое, аллергическое, канцерогенное, раздражающее, нарушение тканевого дыхания, действие на центральную нервную систему |
| Неоптимальные метеорологические условия | Нарушение терморегуляции |
| Нерациональное освещение | Нарушение зрительной функции, снижение работоспособности |
| Неудобная рабочая поза | Патологическое изменение опорно-двигательного аппарата, снижение работоспособности |
| Напряженный умственный труд в условиях дефицита времени | Нервно-психическое, зрительное напряжение, переутомление, снижение работоспособности |
| Монотонный труд | Мышечное напряжение, утомляемость, снижение работоспособности и интереса к работе. |

Предлагаемые мероприятия направлены на защиту от вредных и опасных факторов в служебных помещениях. Они могут быть сгруппированы, как:

) Гигиенические:

нормирование факторов производственной среды, влияющих на здоровье и работоспособность;

оздоровление условий труда путем уменьшения и ликвидации вредных факторов производственной среды.

) Физиологические:

устройство рабочего места и использование оборудования в соответствии с физиологическими требованиями;

внедрение физиологически обоснованных режимов труда и отдыха;

уменьшение умственной и эмоциональной напряженности труда.

) Разработка и внедрение мероприятий по созданию благоприятного климата в коллективе, высокой заинтересованности в труде и его результатах.

) Эстетические:

соблюдение требований технической эстетики при оформлении интерьеров, расположении оборудования, цветовом оформлении и т.п.

Назначение, технические данные, состав, устройство и работа измерителя ИП-50

Измеритель ИП-50 предназначен для измерения напряженности электрического и магнитного поля промышленной частоты при контроле за соблюдением ПДУ, касающихся гигиены труда и коммунально-бытовой гигиены, установленных в стандартах Российской Федерации [3].

Измеритель ИП-50 работает от источника постоянного напряжения 9В, типа «КРОНА», АА. И др.

Рабочие условия эксплуатации измерителя:

температура окружающего воздуха  К

относительная влажность воздуха 

атмосферное давление  кПа

Измеритель ИП-50 имеет следующие характеристики:

рабочая частота………………………………….................... Гц

диапазон измерение среднеквадратического значения напряженности магнитного поля…………………………………………………0,01-100 А/м

первый поддиапазон…………………………………….0,01-100 А/м

второй поддиапазон………………………………………...1-100 А/м

диапазон измерение среднеквадратического значения напряженности электрического поля………………………………................1 В/м-100 кВ/м

первый поддиапазон с ПИП N1………………………….0,01-1 кВ/м

первый поддиапазон с ПИП N2………………………0,001-0,1 кВ/м

второй поддиапазон с ПИП N1…………………………...1-100 кВ/м

второй поддиапазон с ПИП N2…………………………..0,1-10 кВ/м

погрешность измерения, не более……………………….…………...

габаритные размеры ИП-50 в упаковке……………………0,18х0,25х0,1 м

масса ИП-50, не более…………………………………………………..1,0 кг

электропитание - одна батарея типа «КРОНА», либо одна аккумуляторная батарея напряжением 9 В типа 7Д-0,115

время непрерывной работы, не менее……………………………...12 часов

В состав ИП-50 входят:

измеритель ИП-50…………………………………………………...…..1 шт.

ручка-держатель ИП-50…………….…………………………………..1 шт.

удлинитель ручки-держателя ИП-50….……………………………....2 шт.

ПИП электрического поля N1……………………………………….....1 шт.

ПИП электрического поля N2….………………………………………1 шт.

батарея типа «КРОНА»…………………………………………………1 шт.

техническое описание и инструкция по эксплуатации……………….1 шт.

формуляр...………………………………………………………………1 шт.

упаковочная коробка...………………………………………………….1 шт.

Принцип действия измерителя ИП-50 состоит в преобразовании измеряемой физической величины напряженности магнитного или электрического поля в постоянное напряжение с последующим его измерением АЦП интегрирующего типа.

Измеритель ИП-50 состоит из следующих основных частей:

ПИП напряженности магнитного поля и электри­ческого поля N1 и N2;

А1 - блок интеграторов;

А2 - входной делитель напряжения;

АЗ - измерительный усилитель;

АЗ - АЦП с жидкокристаллическим индикато­ром;

ПИП напряженности магнитного поля промышленной частоты (индукционного типа) расположен непосредственно в корпусе измерителя ИП-50 параллельно нижней части корпуса. Его выход, через входной разъем, подключен к входу усилителя.

ПИП напряженности электрического поля промышленной частоты N1 или N2 (емкостного типа) подключается ко входному разъему расположенному на верхней крышке ИП-50. При этом происходит автоматическое отключение ПИП напряженности магнитного поля от входа усилителя.

Блок интеграторов совместно с ПИП формирует амплитудно-частотную характеристику измерителя.

Входной делитель напряжения обеспечивает постоянный уровень входного сигнала для усилителя на различных пределах измерения.

Измерительный усилитель обеспечивает усиление сигнала поступающего с ПИП, частотную фильтрацию и детектирование.

АЦП используется для прямого преобразования напряжения, поступающего с выхода измерительного усилителя в соответствующий ему цифровой эквивалент. Цифровая информация па выходе АЦП представляется в семисегментном коде. Цифровой отсчет производится на 3,5 декадном жидкокристаллическом индикаторе.

Конструктивно измеритель ИП-50 выполнен в виде переносного прибора. Корпус измерителя разделен на отсеки, в которых размещены его функциональные узлы. На верхней панели корпуса измерителя расположен входной разъем. На передней панели корпуса находятся кнопка включения-выключения измерителя, кнопка переключения пределов измерения, цифровой индикатор и отверстие для подключения ручки-держателя ИП-50.

На задней панели корпуса измерителя, под крышкой, расположен батарейный отсек.

Измеритель, два ПИП ёмкостного типа, ручка-держатель, удлинитель ручки держателя, ТО и ИЭ, формуляр находятся в специальной упаковочной коробке для их перевозки и хранения.

Проведение электромагнитного мониторинга на рабочем месте пользователя ПК при разработке САКЗ

Инструментальный контроль электромагнитной обстановки на рабочих местах пользователей персонального компьютера (ПК) производится:

при вводе ПК в эксплуатацию и организации новых и реорганизации рабочих мест;

после проведения организационно-технических мероприятий, направленных на нормализацию электромагнитной обстановки при аттестации рабочих мест по условиям труда;

по заявкам предприятий и организаций.

Инструментальный контроль уровней ЭМП должен осуществляться приборами с допускаемой основной относительной погрешностью измерений ± 20%.

Для измерения уровней напряженности на экране ПК необходимо установить типичное для данного вида работы изображение (текст программы, графики и др.). При проведении измерений должна быть включена вся вычислительная техника, монитор и другое используемое для работы электрооборудование, размешенное в данном помещении. Измерения параметров электростатического поля проводить не ранее, чем через 20 минут после включения ПК.

Проведение измерений

Измерение уровней переменных электрических и магнитных полей, статических электрических полей на рабочем месте, оборудованном ПК, производится на расстоянии 50 см от экрана на трех уровнях на высоте 0,5 м, 1,0 м и 1,5 м. Результаты мониторинга приведены в Табл. 9.

Табл. 9. Результаты электромагнитного мониторинга

|  |  |  |
| --- | --- | --- |
| Высота | Напряженность магнитного поля, А/м | Напряженность электрического поля, В/м |
| 0,5 м | 0,19 | 0,9 |
| 1,0 м | 0,22 | 0,5 |
| 1,5 м | 0,28 | 0,6 |

Аттестация рабочего места пользователя ПК по электромагнитной безопасности при разработке САКЗ

Отнесение условий труда к тому или иному классу вредности и опасности при воздействии ЭМП и излучений осуществляется в соответствии с Табл. 10.

Табл. 10. Классы условий труда при действии неионизирующих ЭМП и излучений

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Показатель | Класс условий труда | | | | | | |
|  | оптимальный | допустимый | вредный | | | | опасный |
|  | 1 | 2 | 3.1 | 3.2 | 3.3 | 3.4 | 4 |
| Электромагнитные поля на рабочем месте пользователя ПК | - | ≤ ВДУ | >ВДУ | - | - | - | - |

Согласно СанПиН [2] временные допустимые уровни (ВДУ) воздействия напряжённости электрической составляющей ЭМП и плотность магнитного потока на расстоянии 50 см от экрана дисплея (40 см от центра клавиатуры портативного компьютера) не должны превышать соответственно значений 25 В/м и 250 нТл в диапазоне частот 5 Гц…2 кГц (табл. 4). Следует учитывать, что 250 нТл соответствует напряжённости магнитной составляющей ЭМП в воздухе Н = 0,2 А/м.

Табл. 11. ВДУ на человека электромагнитных полей

|  |  |
| --- | --- |
| Виды поля | ВДУ |
| Напряженность электрического поля в диапазоне частот 5 Гц…2кГц | 25 В/м |
| Плотность магнитного потока в диапазоне частот 5 Гц…2кГц | 0,2 А/м |

Сравнив результаты измерения ЭМП на рабочем месте (Табл. 9) с данными таблицы ВДУ ЭМП создаваемых ПК (Табл. 11) определяем, что класс условий труда на рабочем месте относится к классу 3.1 (вредные условия труда).

Оценка тяжести трудового процесса компьютеризированного рабочего места при разработке САКЗ

Тяжесть труда - характеристика трудового процесса, отражающая преимущественно нагрузку на опорно-двигательный аппарат и функциональные системы организма (сердечно-сосудистую, дыхательную и др.), обеспечивающие его деятельность [6].

Тяжесть трудового процесса оценивают по ряду показателей, выраженных в эргометрических величинах, характеризующих трудовой процесс, независимо от индиви­дуальных особенностей человека, участвующего в этом процессе. Основными показа­телями тяжести трудового процесса являются:

физическая динамическая нагрузка;

масса поднимаемого и перемещаемого груза вручную;

стереотипные рабочие движения;

статическая нагрузка;

рабочая поза;

наклоны корпуса;

перемещение в пространстве.

Физическая динамическая нагрузка (выражается в единицах внешней механической работы за смену - кг\*м)

Для подсчета физической динамической нагрузки (внешней механической работы):

Определяется масса груза (деталей, изделий, инструментов и т.д.), перемещаемого вручную в каждой операции и путь его перемещения в метрах.

Подсчитывается общее количество операций по переносу груза за смену.

Суммируется величина внешней механической работы (кг\*м) за смену в целом; по величине внешней механической работы за смену, в зависимости от вида нагрузки (региональная или общая) и расстояния перемещения груза, определяют к какому классу условий труда относится данная работа.

Расчет. Поскольку программист во время работы не производит никакой физической динамической нагрузки, то по данному пока­зателю класс его работы не оценивается.

Масса поднимаемого и перемещаемого груза вручную (кг)

Для определения массы груза (поднимаемого или переносимого работником на протяжении смены, постоянно или при чередовании с другой работой):

Его взвешивают на товарных весах.

Регистрируется только максимальная величина.

Массу груза можно также определить по документам.

Расчет. Поскольку программист во время работы не производит никакой физической нагрузки, то по данному пока­зателю класс его работы не оценивается.

Стереотипные рабочие движения (количество за смену, суммарно на две руки)

Понятие «рабочее движение» в данном случае подразумевает движение элементарное, т.е. однократное перемещение рук (или руки) из одного положения в другое. Стереотипные рабочие движения делятся на:

локальные - выполняются с участием мышц кистей и пальцев в быстром темпе (60-250 движений в минуту) и

региональные - выполняются с преимущественным участием мышц рук и плечевого пояса в более медленном темпе.

Время работы определяется путем хронометражных наблюдений или по фотографии рабочего дня.

Расчет. Программист, в среднем набирает код со скоростью 200 символов в минуту. Всего основная работа занимает 65 % рабочего времени, что при восьмичасовом рабочем дне составляет 312 минут. Таким образом, общее число вводимых символов за рабочий день - 62400.

Помимо нажатий клавиш на клавиатуре, он также в среднем производит 15 кликов мышкой в минуту. Следовательно, общее число кликов за рабочий день - 4680.

Суммируя значения, по данному пока­зателю его работу относят к классу 3,2.

Статическая нагрузка (величина статической нагрузки за смену при удержании груза, приложении усилий, кгс \* с)

Статическая нагрузка, связанная с удержанием груза (обрабатываемого инструмента или изделия) или приложением усилия (рукоятки, маховики, штурвалы), рассчитывается путем перемножения двух параметров:

величины удерживаемого усилия (веса груза) и

времени его удерживания.

Расчет. Программист во время работы удерживает в руке мышь весом 0,1 кгс и перемещает ее по поверхности стола в течение 35% рабочего времени, что при восьмичасовом рабочем дне составляет 10080 с. Величина статической нагрузки будет составлять 1008 кгс \* с (0,1 кгс \* 10080 с). Работа по дан­ному показателю относится к первому классу.

Рабочая поза

Характер рабочей позы определяется визуально. Рабочая поза бывает:

свободная - удобные позы сидя, которые дают возможность изменения рабочего положения тела или его частей: откинуться на спинку стула, изменить положение ног, рук;

неудобная - позы с большим наклоном или поворотом туловища, с поднятыми выше уровня плеч руками, с неудобным размещением нижних конечностей;

фиксированная - невозможность изменения взаимного положения различных частей тела относительно друг друга, например, при выполнении работ с использованием оптических увеличительных приборов: луп и микроскопов;

вынужденная - позы лежа, на коленях, на корточках и т.д.

Абсолютное время (в минутах, часах) пребывания в той или иной позе определяется на основании хронометражных данных за смену. Если по характеру работы рабочие позы разные, то оценку следует проводить по наиболее типичной позе для данной работы.

Расчет. Программист около 90 % рабочего времени проводит в фик­сированной позе - сидя работает за ПК, при этом может откинуться на спинку стула, изменить положение ног, рук. По этому показателю работу можно отне­сти к первому классу.

Наклоны корпуса (количество за смену)

Число наклонов за смену определяется: путем их прямого подсчета в единицу времени (несколько раз за смену), затем рассчитывается число наклонов за все время выполнения работы, либо определением их количества за одну операцию и умножением на число операций за смену.

Расчет. Поскольку работа программиста в основном протекает в положении сидя, при этом за весь рабочий день в редких случаях совершаются глубокие наклоны, то по этому показателю класс его работы не оценивается.

Перемещение в пространстве (переходы, обусловленные технологическим процессом) в течение смены по горизонтали или вертикали - по лестницам, пандусам и др., км

Самый простой способ определения этой величины - с помощью шагомера для определения шагов за смену. Количество шагов за смену следует умножить на длину шага и полученную величину выразить в км. Перемещением по вертикали можно считать перемещения по лестницам или наклонным поверхностям, угол наклона которых более  от горизонтали.

Расчет. Поскольку работа программиста в основном протекает в положении сидя, и даже с учетом перерыва на обед тяжесть труда относится ко первому классу.

Общая гигиеническая оценка условий труда при работе за ПК

Общая оценка по степени физической тяжести проводится на основе всех при­веденных выше показателей. При этом вначале устанавливается класс по каждому из­меренному показателю и вносится в таблицу, а окончательная оценка тяжести труда устанавливается по показателю, отнесенному к наибольшему классу. При наличии двух и более показателей класса 3.1 и 3.2 общая оценка устанавливается на одну степень выше.

|  |  |  |  |
| --- | --- | --- | --- |
| № | Показатели | Факт, значения | Класс |
| 1 | 2 | 3 | 4 |
| 1 | Физическая динамическая нагрузка (кг\*м): регио­нальная - перемещение груза до 1 м общая на­грузка: перемещение груза |  |  |
| 1.1 | от 1 до 5 м | - | - |
| 1.2 | более 5 м | - | - |
| 2 | Масса поднимаемого и перемещаемого вручную груза (кг): |  |  |
| 2.1 | при чередовании с другой работой | - | - |
| 2.2 | постоянно в течение смены | - | - |
| 2.3 | суммарная масса за каждый час смены: |  |  |
|  | с рабочей поверхности | - | - |
|  | с пола | - | - |
| 3 | Стереотипные рабочие движения (кол-во): |  |  |
| 3.1 | локальная нагрузка | 67080 | 3,1 |
| 3.2 | региональная нагрузка | - | - |
| 4 | Статическая нагрузка (кгс \* с) |  |  |
| 4.1 | одной рукой | 1080 | 1 |
| 4.2 | двумя руками | - | - |
| 4.3 | с участием корпуса и ног | - | - |
| 5 | Рабочая поза | сидя 90 % | 1 |
| 6 | Наклоны корпуса (количество за смену) | - | - |
| 7 | Перемещение в пространстве (км): |  |  |
| 7.1 | по горизонтали | до 1 | 1 |
| 7.2 | по вертикали | - |  |
| Окончательная оценка тяжести труда | |  | 3,1 |

Итак, поскольку оценка тяжести труда устанавливается по показателю, отнесенному к наибольшему классу, то окончательная оценка тяжести трудового процесса программиста - класс 3,1.

Вредные условия труда (3 класс, 1 степень) характеризуются такими отклонениями уровней вредных факторов от гигиенических нормативов, которые вызывают функциональные изменения, восстанавливающиеся, как правило, при более длительном (чем к началу следующей смены) прерывании контакта с вредными факторами и увеличивают риск ухудшения здоровья.

# Планы на будущее.

Для работы в дальнейшем, в самых ближайших перспективах, планируется создание клиент-серверной архитектуры проекта, где серверная часть, как основополагающий элемент, будет находиться на компьютере преподавателя, а генерироваться вариант - на машине учащегося, что, безусловно, очень важно. При этом, в процессе работы, html страница теста будет передаваться через различные запросы, что значительно упростит взаимодействие между пользователями.

Помимо этого, в будущем, в наиболее отдаленных перспективах, планируется создание реляционной базы данных, которая будет служить важным инструментом для записи результатов каждого учащегося на машине преподавателя, что, безусловно, будет способствовать лучшему контролю и анализу успеваемости.

# ЗАКЛЮЧЕНИЕ

Подводя итоги данной работы, можно смело сказать, что все поставленные цели для создания приложения для тестирования учащихся были успешно выполнены. При разработке приложения был использован мощный и универсальный язык программирования Python, который позволяет создавать эффективные решения. Для создания веб-страницы теста использован язык гипертекста HTML, который обеспечивает необходимую структуру и визуальное представление информации. В качестве базы данных была выбрана реляционная база данных MySQL, которая была написана на языке запросов SQL, что обеспечивает надежное хранение и обработку данных.

К сожалению, несмотря на все усилия, от человеческого фактора в процессе проверки избежать не удается, поскольку существует блок заданий, которые можно проверить только с помощью преподавателя. Это актуальный момент, который требует внимательного подхода. Однако, основная задача программы заключалась в том, чтобы сэкономить время на проверке тестовой части экзамена, и, можно с уверенностью сказать, что данная программа успешно справилась с этой задачей, значительно облегчая процесс оценки знаний учащихся.

Внедрение и совершенствование системы ЕГЭ в России началось как эксперимент в 2001 году, привлекая первоначально несколько регионов. С течением времени организация и контент заданий постоянно совершенствовались для обеспечения объективности. Официальное введение ЕГЭ в 2009 году и последующие годы привели к постоянному обновлению и усовершенствованию экзаменационной системы. Президент России выразил необходимость улучшения данной системы в 2023 году, чтобы обеспечить равные возможности для всех выпускников страны и обеспечить доступ к ведущим вузам. Реформы включали в себя создание различных вариантов заданий, внедрение устной части в экзамены по иностранным языкам, а также разделение математических заданий на базовый и профильный уровни, что способствует более глубокому и комплексному оцениванию знаний выпускников.

Внедрение и развитие компьютерного формата экзаменов с использованием нейросетей и искусственного интеллекта представляет собой перспективное направление совершенствования системы образования. Первые шаги в этом направлении были сделаны еще в 2018 году, когда была проведена апробация компьютерного формата ЕГЭ по информатике. Однако, существуют опасения относительно разнообразия компьютерного оборудования и необходимости подготовленных специалистов.

Компьютерный формат экзаменов открывает перед образовательной системой новые возможности в оценке знаний и навыков школьников. С помощью интерактивных заданий, а также включения видео- и аудиофрагментов, можно создать более разнообразные и эффективные тестовые задания по различным предметам, таким как география, история, астрономия, биология, физика и химия.

Использование ИИ-алгоритмов для проверки письменных и устных ответов на иностранные языки предоставляет возможность более точной и объективной оценки знаний экзаменуемых. Программы могут анализировать произношение, использование лексики и правильность ответов, что способствует улучшению качества оценки.

Несмотря на потенциальные преимущества компьютерного формата экзаменов, вопросы о целесообразности и надежности данного подхода требуют серьезного исследования. Оценка творческих работ, нестандартных решений и аналитических навыков по-прежнему остается сложной задачей, требующей дальнейшего изучения и анализа.Начало формы

# Список литературы

1. <https://ru.wikipedia.org/wiki/Программирование> (дата обращения: 05.05.2025)
2. <https://ctege.info/ege-2025/shkala-perevoda-ballov-ege-2025.html> (дата обращения: 19.05.2025)
3. <https://fipi.ru/>

# Приложение А

**Листинг приложения**

**Класс create\_test.py**

```python

from tkinter import \*

from tkinter import ttk

import os

import sys

import generation\_content as generator

# Создаем главное окно приложения

root = Tk()

# Устанавливаем заголовок окна

root.title("Главное меню")

# Устанавливаем размеры окна

root.geometry("250x30")

# Запрещаем изменение размеров окна

root.resizable(False, False)

# Отключаем возможность отсоединения меню

root.option\_add("\*tearOff", FALSE)

# Создаем текстовое поле для редактора

editor = Text()

editor.pack(expand=1, fill=BOTH)

# Определяем функцию для генерации теста

def gen():

# Отправляем в консоль сгенерированный контент

print(generator.cleanhtml(generator.tester))

# Определяем функцию выхода из приложения

def exit():

# Закрываем главное окно и завершаем работу программы

root.destroy()

# Создаем меню "Файл"

file\_menu = Menu()

# Добавляем в меню команду "Выйти" с привязкой функции exit

file\_menu.add\_command(label="Выйти", command=exit)

# Создаем главное меню и добавляем в него подменю "Файл"

main\_menu = Menu()

main\_menu.add\_cascade(label="Файл", menu=file\_menu)

# Применяем главное меню к главному окну

root.config(menu=main\_menu)

# Создаем кнопку для генерации теста

btn = ttk.Button(editor, text="Сгенерировать тест", command=gen)

# Размещаем кнопку в текстовом редакторе

editor.window\_create("1.0", window=btn)

# Получаем директорию скрипта

script\_directory = os.path.dirname(os.path.abspath(sys.argv[0]))

# Устанавливаем иконку для главного окна приложения

root.iconbitmap(f'{script\_directory}\\fipi.ico')

# Запускаем главный цикл приложения

root.mainloop()

```

В этом коде приведена реализация простого графического интерфейса с использованием библиотеки Tkinter. Программа открывает главное окно с заголовком "Главное меню", в котором предоставляется пользователю возможность сгенерировать тест, а также выйти из приложения через меню. Напоминанием о структуре кода служит наличие кнопки для генерации и текстового поля. Обратите внимание, что комментарии добавлены для более полного понимания каждого этапа работы с приложением.

**Класс generation\_content.py**

# Импорт библиотек

import requests

import re

import random

# Очистка html от тегов

CLEANHTML = re.compile('<.\*?>|&([a-z0-9]+|#[0-9]{1,6}|#x[0-9a-f]{1,6});')

def cleanhtml(raw\_html):

cleantext = re.sub(CLEANHTML, '', raw\_html)

return cleantext

# Основные переменные

list = 9094

sections\_array = [1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7] # Включает разделы задач

number = '4FF344' # Уникальный ID варианта

finder = 'Номер: ' # Искать номер

result = '' # Изначальный результат

# Цикл по массиву разделов

for i in range(len(sections\_array)):

start = -1

count = 0

end = -1

# Формируем ссылку для запроса

link\_one = f"http://ege.fipi.ru/bank/questions.php?search=1&pagesize={list}&proj=E040A72A1A3DABA14C90C97E0B6EE7DC&theme={sections\_array[i]}&qlevel=&qkind=&qsstruct=&qpos=&qid=&zid=&solved=&favorite=&blind="

req = requests.get(link\_one)

src = req.text

print(src) # Выводим исходный текст ответа

lister = [] # Список для хранения позиций найденных номеров

# Цикл для поиска всех номеров задач

while True:

start = src.find('Номер: ', start + 1)

if start == -1:

break

end = start + 31

count += 1

lister.append(end)

print("Количество вхождений символа в строку: ", count)

print(f'{sections\_array[i]} {list}')

test = ''

# Цикл для создания массива ответов

for i in range(len(lister)):

for j in range(6):

test += req.text[lister[i] + j]

test += ' '

test = test.split(" ")

rand = random.randint(0, count - 1) # Случайный выбор номера

print(test[rand])

number = test[rand] # Запоминаем случайный номер

# Формирование второй ссылки для получения конкретного вопроса

link\_two = f"http://ege.fipi.ru/bank/questions.php?search=1&pagesize=10&proj=E040A72A1A3DABA14C90C97E0B6EE7DC&theme=&qlevel=&qkind=&qsstruct=&qpos=&qid={number}&zid=&solved=&favorite=&blind="

req = requests.get(link\_two)

src = req.text

tester = '' # Переменная для хранения найденного теста

# Поиск текста вопроса в HTML ответе

tester = re.search('<p class="MsoNormal">(.+?)</p>', src)

if tester:

tester = tester.group(1)

else:

tester = re.search('<P class=Basis><SPAN>(.+?)</P>', src).group(1)

tester = tester.replace('m:', '')

tester = tester.replace('math', 'math xmlns="http://www.w3.org/1998/Math/MathML"')

# Сбор результатов для финального вывода

result += f'<p class="MsoNormal">{tester}</p>\n'

text = f'<!DOCTYPE html>\n<html lang="en">\n<head>\n<meta charset="UTF-8">\n<meta name="viewport" content="width=device-width, initial-scale=1.0">\n<title>Document</title>\n</head>\n<body>\n{result}\n</body>\n</html>'

# Запись результата в HTML файл

with open("hello.html", "w", encoding="utf-8") as file:

file.write(text)

print("Файл записан")

**Пояснение к коду**

Данный код реализует парсер для извлечения данных из открытого банка задач ФИПИ и формирования HTML-документа с вопросами. Сначала программа получает список задач из сайта, анализирует их, а затем сохраняет результаты в файле hello.html.

# Приложение B

Баллы ЕГЭ 2025 по заданиям на примере ЕГЭ по математики профиль

|  |  |
| --- | --- |
| **Номер задания по математике** | **Первичный балл** |
| 1 | 1 |
| 2 | 1 |
| 3 | 1 |
| 4 | 1 |
| 5 | 1 |
| 6 | 1 |
| 7 | 1 |
| 8 | 1 |
| 9 | 1 |
| 10 | 1 |
| 11 | 1 |
| 12 | 1 |
| 13 | 2 |
| 14 | 3 |
| 15 | 2 |
| 16 | 2 |
| 17 | 3 |
| 18 | 4 |
| 19 | 4 |
| **ИТОГО** | 32 |

# Приложение С

Таблица перевода первичных баллов на примере ЕГЭ по математики профиль

|  |  |
| --- | --- |
| Первичный балл | Тестовый балл |
| 1 | 6 |
| 2 | 11 |
| 3 | 17 |
| 4 | 22 |
| 5 | 27 |
| 6 | 34 |
| 7 | 40 |
| 8 | 46 |
| 9 | 52 |
| 10 | 58 |
| 11 | 64 |
| 12 | 70 |
| 13 | 72 |
| 14 | 74 |
| 15 | 76 |
| 16 | 78 |
| 17 | 80 |
| 18 | 82 |
| 19 | 84 |
| 20 | 86 |
| 21 | 88 |
| 22 | 90 |
| 23 | 92 |
| 24 | 94 |
| 25 | 95 |
| 26 | 96 |
| 27 | 97 |
| 28 | 98 |
| 29 | 99 |
| 30 | 100 |
| 31 | 100 |
| 32 | 100 |

Где:

 - это минимальный порог для получения аттестата и поступления в вузы

 - это минимальный порог для поступления в подведомственные образовательные учреждения Минобрнауки