Выполнение работы

**Задание 1.** Познакомиться с программами тестирования. Дать им краткую характеристику и выбрать на ваш взгляд 2 наиболее удобные, обосновав свой выбор по различным критериям.

Для выбора двух наиболее удобных программ для тестирования из предложенного списка рассмотрим каждую из них более подробно и выделим их основные характеристики:

1. Adit Testdesk:
   * Описание: Adit Testdesk — это многофункциональная система тестирования, предназначенная для создания, проведения и анализа тестов.
   * Основные характеристики: Разнообразие типов вопросов, поддержка множественных вариантов ответов, возможность создания тестов с разными уровнями сложности, удобный интерфейс для анализа результатов.
   * Обоснование выбора: Adit Testdesk предоставляет обширные возможности для создания и администрирования тестов, а также анализа результатов, что делает его удобным инструментом для образовательных целей.
2. iSpring QuizMaker:
   * Описание: iSpring QuizMaker — инструмент для создания интерактивных тестов и опросов.
   * Основные характеристики: Простой и интуитивно понятный интерфейс, многообразие типов вопросов, возможность встраивания тестов в презентации и онлайн-курсы.
   * Обоснование выбора: iSpring QuizMaker отличается простотой использования и подходит для быстрого создания интерактивных тестов и опросов, особенно в контексте обучения.
3. Конструктор тестов easyQuizzy:
   * Описание: easyQuizzy — инструмент для создания тестов и опросов с поддержкой разных типов вопросов.
   * Основные характеристики: Простой интерфейс, возможность создавать разнообразные вопросы, создание и сохранение базы вопросов.
   * Обоснование выбора: easyQuizzy может быть хорошим выбором для пользователей, которым нужен инструмент для быстрого создания простых тестов и опросов.

Итак, на основе предоставленных характеристик, для более продвинутых потребностей рекомендуется выбрать Adit Testdesk, а для простых и быстрых задач — iSpring QuizMaker.

**Задание 2.** Познакомиться с программой MyTestX и выполнить задания:

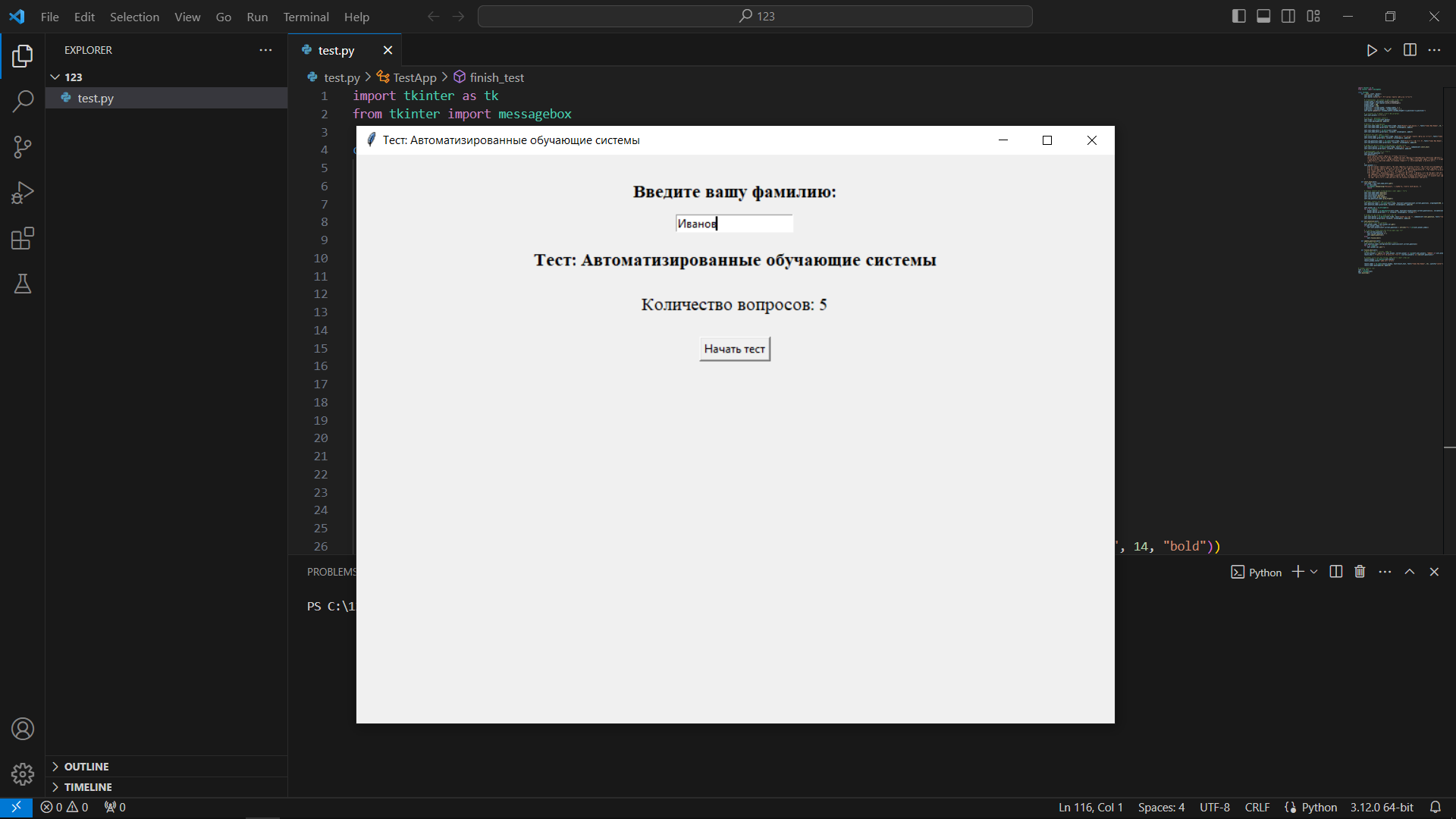
1. Назначение и состав программы MyTest X: Программа MyTest X предназначена для создания, проведения компьютерного тестирования и анализа результатов. Её компоненты включают:
   * Модуль тестирования (MyTestStudent): Позволяет тестируемым проходить тесты и получать результаты в соответствии с настройками теста.
   * Редактор тестов (MyTestEditor): Обеспечивает удобный интерфейс для создания, редактирования тестов и настройки параметров тестирования, таких как система оценивания и ограничение времени.
   * Журнал тестирования (MyTestServer): Позволяет организовывать тестирование, распространять тесты по сети, собирать результаты и проводить анализ.
2. Программное обеспечение для работы с MyTest X:
   * Поддерживаемые операционные системы: Windows XP, Vista, 7, 8, 8.1, 10. Для использования под Linux можно рассмотреть программу Wine для обеспечения совместимости с Windows-приложениями.
3. Задание характеристик с помощью модуля "Редактор тестов" (MyTestEditor):
   * Характеристики, такие как система оценивания, ограничение времени и режим тестирования, устанавливаются через модуль "Редактор тестов". Пользователь может настраивать параметры каждого теста, включая:
     + Систему оценивания: определение используемой системы (например, от 2-х до 100-бальной).
     + Ограничение времени: установка временных рамок для выполнения теста и/или отдельных заданий.
     + Режим тестирования: выбор режима, такого как обучающий, штрафной, свободный или монопольный, чтобы адаптировать тестирование к конкретным образовательным или оценочным целям.

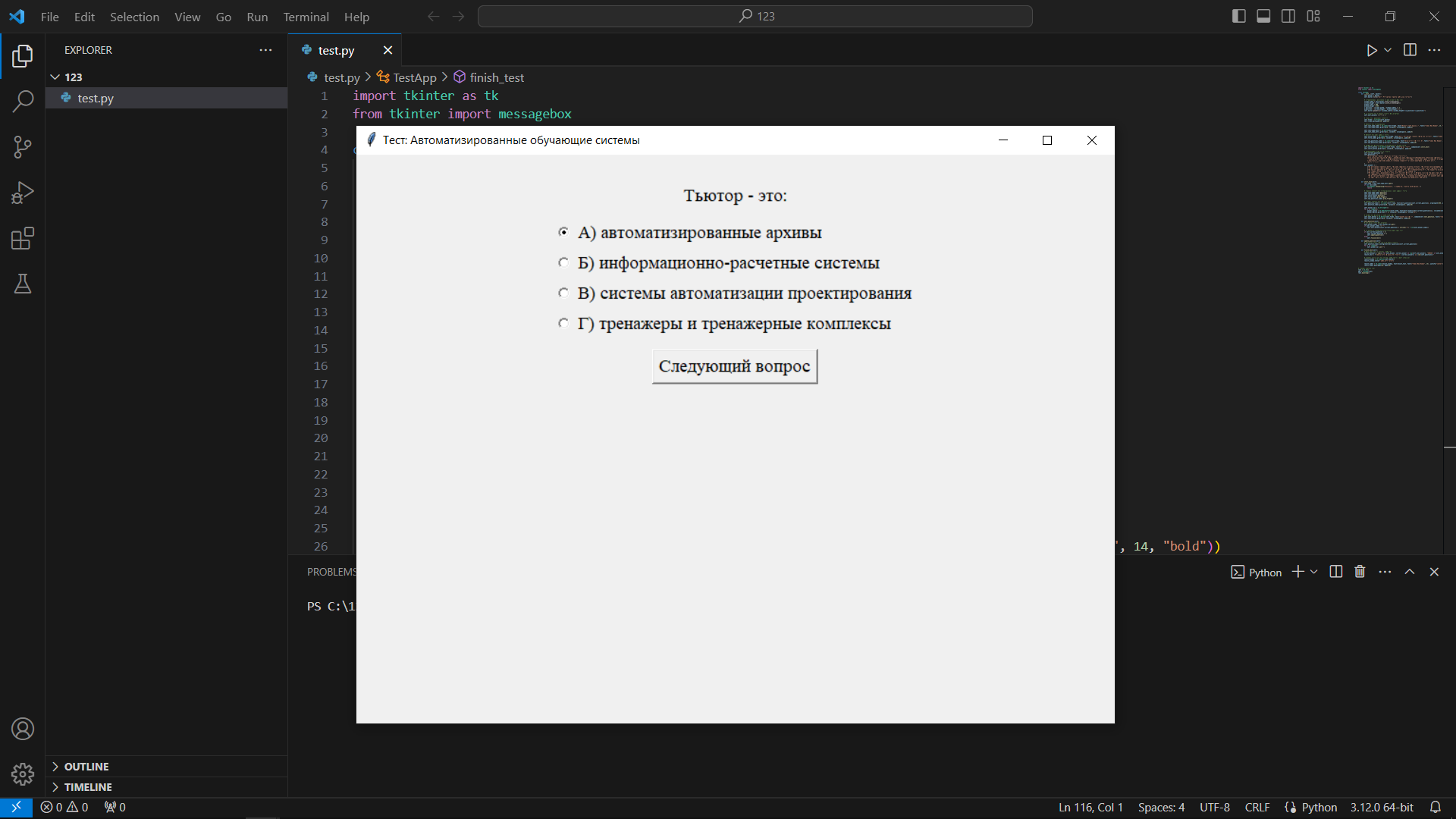
Таким образом, модуль "Редактор тестов" предоставляет пользователю гибкие инструменты для настройки параметров тестирования в соответствии с потребностями.

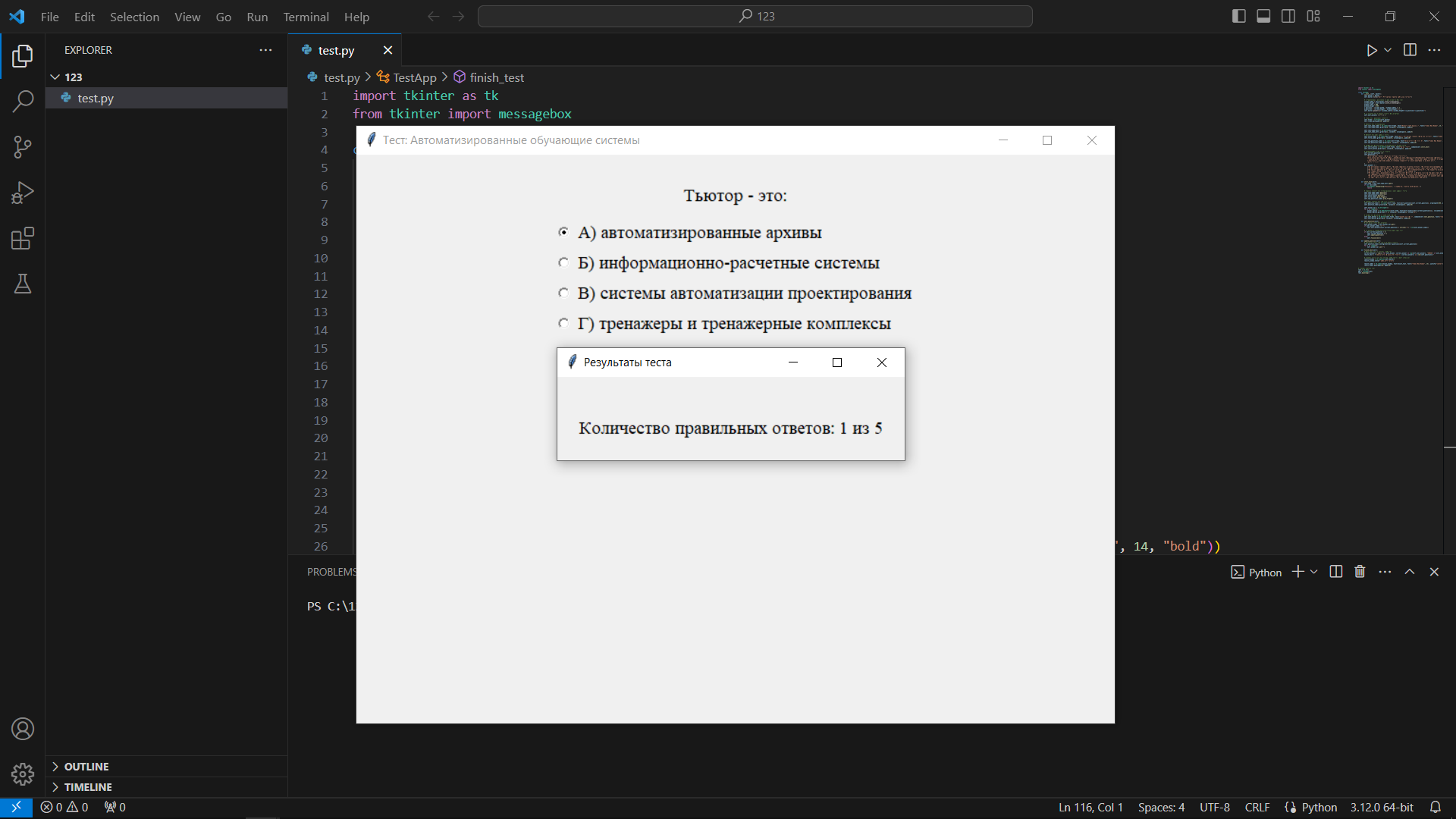
**Задание 3.** Разработайте программу для тестирования на одном из языков программирования.

Был выбран язык Python

Тест «Автоматизированные обучающие системы»







Код программы:

import tkinter as tk

from tkinter import messagebox

class TestApp:

    def \_\_init\_\_(self, master):

        self.master = master

        self.master.title("Тест: Автоматизированные обучающие системы")

        # Устанавливаем разрешение экрана и центрируем окно

        screen\_width = self.master.winfo\_screenwidth()

        screen\_height = self.master.winfo\_screenheight()

        window\_width = 800

        window\_height = 600

        x\_position = (screen\_width - window\_width) // 2

        y\_position = (screen\_height - window\_height) // 2

        self.master.geometry(f"{window\_width}x{window\_height}+{x\_position}+{y\_position}")

        # Переменные для хранения ответов пользователя

        self.user\_answers = [""] \* 5

        # Фрейм для размещения виджетов

        self.frame = tk.Frame(self.master)

        self.frame.pack(padx=20, pady=20)

        # Виджеты для ввода фамилии

        self.last\_name\_label = tk.Label(self.frame, text="Введите вашу фамилию:")

        self.last\_name\_label.grid(row=0, column=0, columnspan=2, pady=5)

        self.last\_name\_entry = tk.Entry(self.frame)

        self.last\_name\_entry.grid(row=1, column=0, columnspan=2, pady=5)

        # Виджеты для информации о тесте

        self.title\_label = tk.Label(self.frame, text="Тест: Автоматизированные обучающие системы", font=("Times New Roman", 14, "bold"))

        self.title\_label.grid(row=2, column=0, columnspan=2, pady=10)

        self.num\_questions\_label = tk.Label(self.frame, text="Количество вопросов: 5", font=("Times New Roman", 14))

        self.num\_questions\_label.grid(row=3, column=0, columnspan=2, pady=10)

        # Кнопка для подтверждения ввода фамилии и начала теста

        self.start\_button = tk.Button(self.frame, text="Начать тест", command=self.start\_test)

        self.start\_button.grid(row=4, column=0, columnspan=2, pady=10)

        # Инициализация вопросов и ответов

        self.current\_question = 0

        self.questions = [

            "К автоматизированным обучающим системам относятся:",

            "Конкретный материальный продукт, реализующий информационно-коммуникационные технологии обучения:",

            "Образовательные ресурсы, обеспечивающие возможность доступа к любой информации в локальных и глобальных сетях, удаленное интерактивное взаимодействие субъектов учебного процесса:",

            "Разновидность веб-конференции, проведение онлайн-встреч или презентаций через Интернет:",

            "Тьютор - это:"

        ]

        self.answers = [

            ["А) автоматизированные архивы", "Б) информационно-расчетные системы", "В) системы автоматизации проектирования", "Г) тренажеры и тренажерные комплексы"],

            ["А) автоматизированные обучающие системы", "Б) информационные системы электронного обучения", "В) электронный образовательный ресурс", "Г) электронный учебник"],

            ["А) демонстрационные ЭОР", "Б) контролирующие ЭОР", "В) коммуникативные ЭОР", "Г) диагностирующие ЭОР"],

            ["А) онлайн-семинар", "Б) коучинг", "В) вебкаст", "Г) вебинар"],

            ["А) видеоролик, размещенный в сети, который можно посмотреть в удобное время на выбранном вами устройстве отображения",

             "Б) специалист по индивидуализации в образовании, посредник между учеником и образовательной системой",

             "В) форма представления содержания учебного курса, основанная на использование современных информационных технологий",

             "Г) набор ИТ-сервисов, использующихся при проведении дистанционного обучения"]

        ]

    def start\_test(self):

        last\_name = self.last\_name\_entry.get()

        if not last\_name:

            messagebox.showwarning("Внимание", "Пожалуйста, введите вашу фамилию.")

            return

        # Удаляем виджеты для ввода фамилии и информации о тесте

        self.last\_name\_label.destroy()

        self.last\_name\_entry.destroy()

        self.start\_button.destroy()

        self.title\_label.grid\_forget()

        self.num\_questions\_label.grid\_forget()

        # Создаем виджеты для вопросов и ответов

        self.question\_label = tk.Label(self.frame, text=self.questions[self.current\_question], wraplength=400, justify="center", font=("Times New Roman", 14))

        self.question\_label.grid(row=0, column=0, columnspan=2, pady=10)

        self.answer\_var = tk.StringVar()

        for i in range(4):

            answer\_option = tk.Radiobutton(self.frame, text=self.answers[self.current\_question][i], variable=self.answer\_var, value=i, font=("Times New Roman", 14))

            answer\_option.grid(row=i + 1, column=0, columnspan=2, sticky="w")

        # Кнопка для перехода к следующему вопросу

        self.next\_button = tk.Button(self.frame, text="Следующий вопрос", command=self.next\_question, font=("Times New Roman", 14))

        self.next\_button.grid(row=6, column=0, columnspan=2, pady=10)

    def next\_question(self):

        # Сохраняем ответ пользователя

        user\_answer\_index = self.answer\_var.get()

        if user\_answer\_index != "":

            self.user\_answers[self.current\_question] = chr(ord('А') + int(user\_answer\_index))

        # Переходим к следующему вопросу или завершаем тест

        if self.current\_question < 4:

            self.current\_question += 1

            self.update\_question()

        else:

            self.finish\_test()

    def update\_question(self):

        # Обновляем текст вопроса и варианты ответов

        self.question\_label.config(text=self.questions[self.current\_question])

        for i in range(4):

            self.answer\_var.set("")

    def finish\_test(self):

        # Завершаем тест и выводим результат

        correct\_answers = sum([1 for user\_answer, correct\_answer in zip(self.user\_answers, "ГБВГБ") if user\_answer == correct\_answer])

        result\_text = f"Количество правильных ответов: {correct\_answers} из {len(self.questions)}"

        # Создаем новое окно для вывода результатов и центрируем его

        result\_window = tk.Toplevel(self.master)

        result\_window.title("Результаты теста")

        result\_label = tk.Label(result\_window, text=result\_text, font=("Times New Roman", 14), justify="center")

        result\_label.pack(padx=20, pady=20)

# Создаем главное окно

root = tk.Tk()

app = TestApp(root)

root.mainloop()