



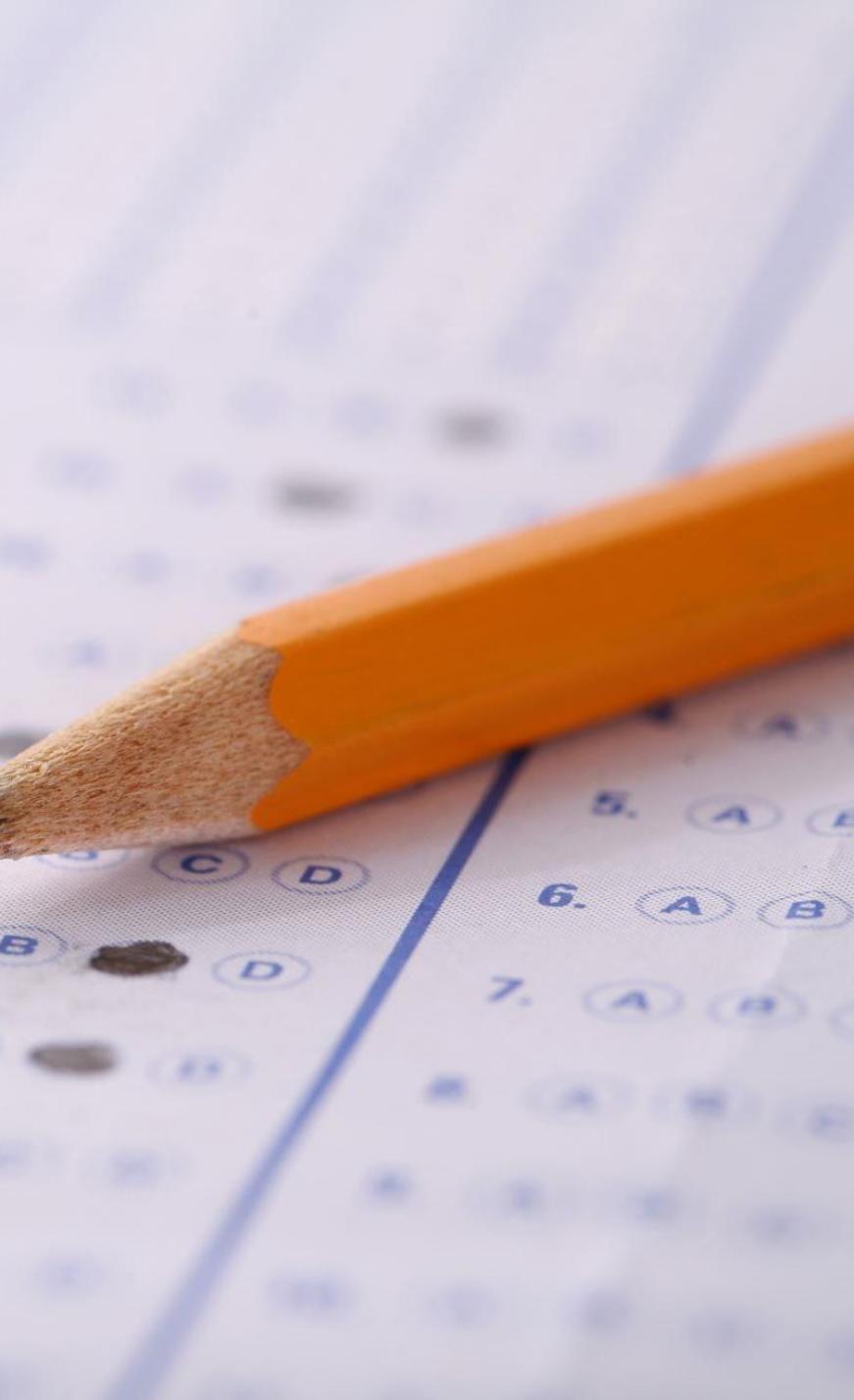
# Как могут помочь нейросети?

Горденко Мария

[mgordenko@gmail.com](mailto:mgordenko@gmail.com)

TG: oduvan\_ja

---



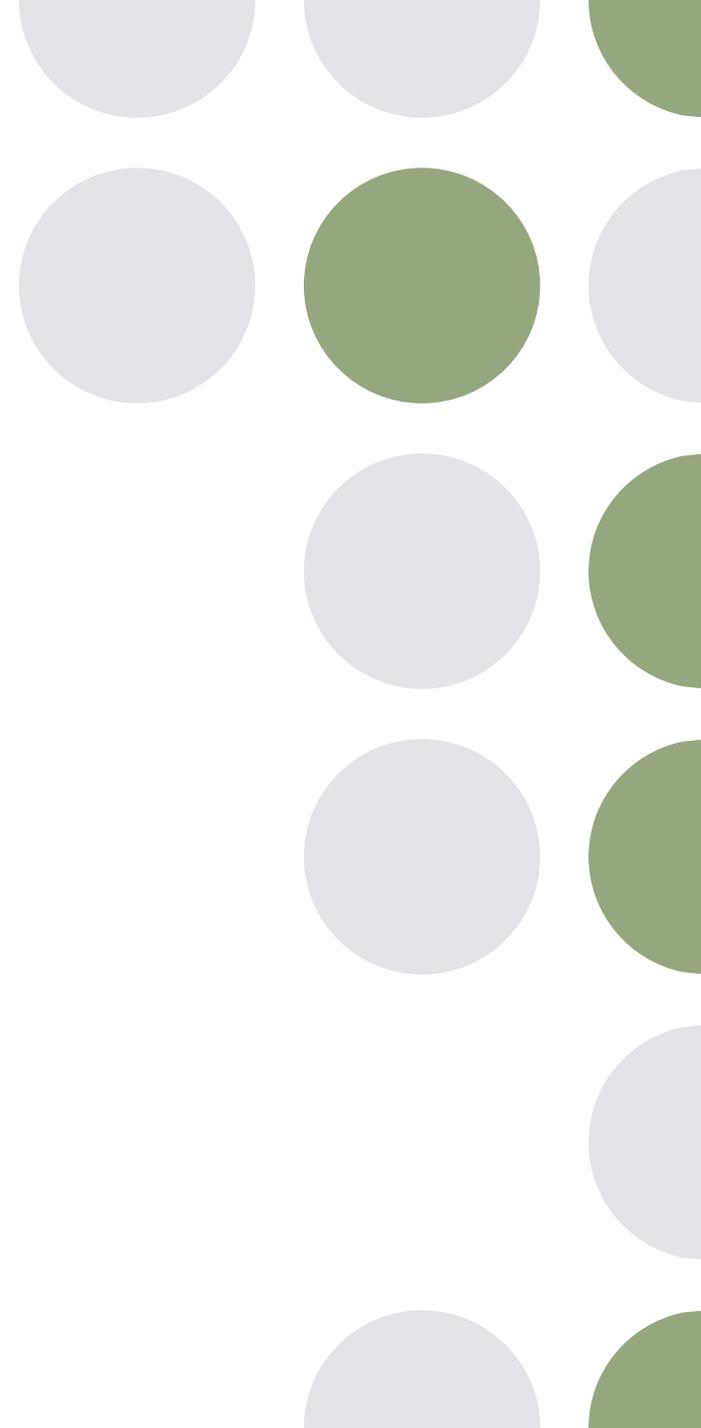
# **ChatGPT: запретить или использовать?**

---



# Использовать

- Адаптация курсов
  - Новые формы оценки
  - Учеба вместе с ИИ
- 

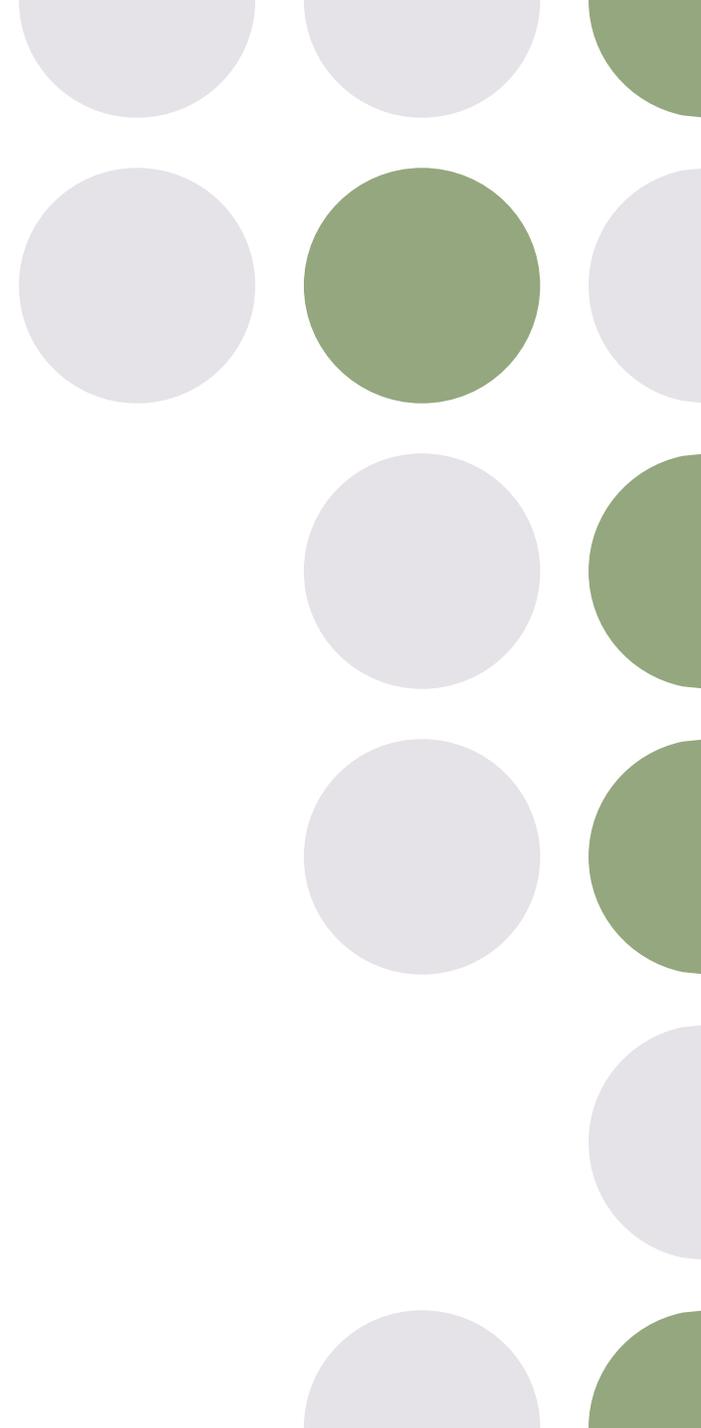


# ИИ на помощи у преподавателя

- Персонализация обучения
  - Виртуальные помощники
  - Оценка работ и обратная связь
  - Генерация образовательных материалов
-

# За чем следить?

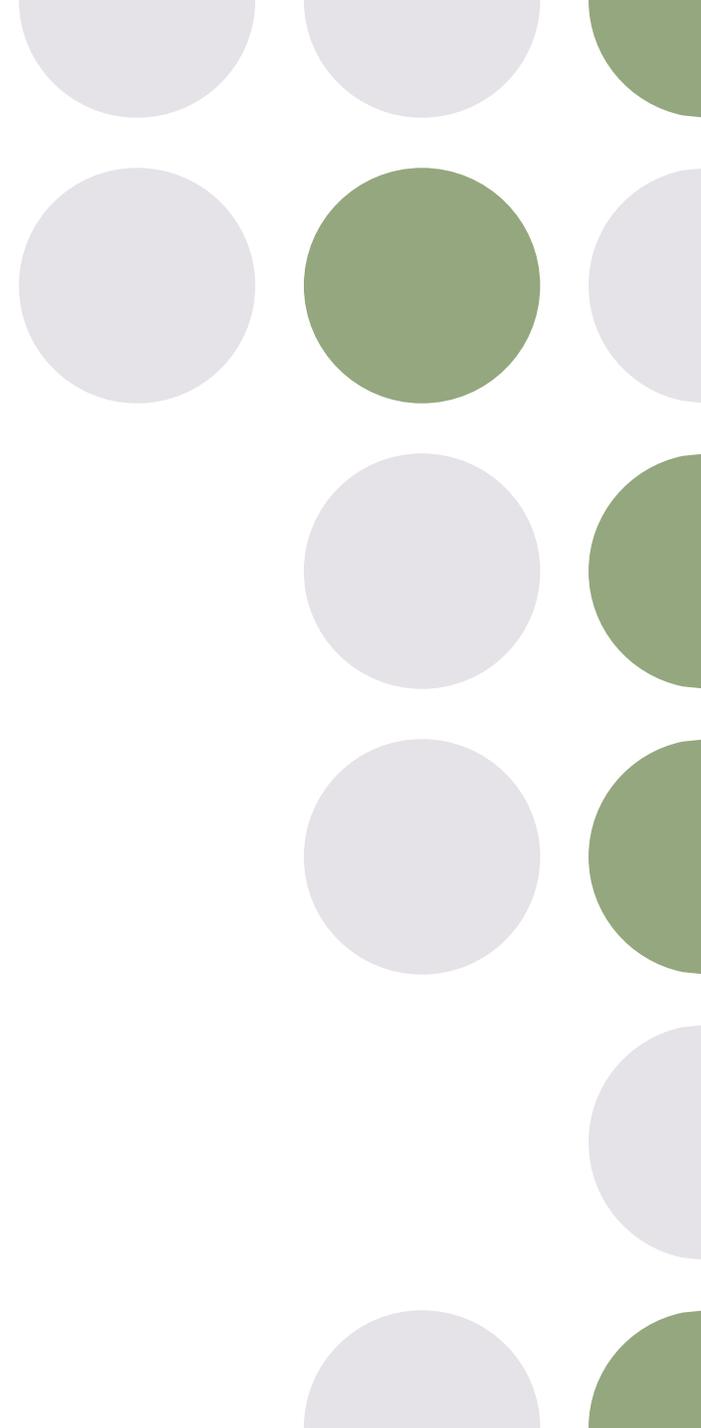
- Корректность материалов
  - Академическая честность
  - Баланс технологий и обучения
- 



# Мои примеры

## 1 кейс. Создание вариантов домашних заданий с ChatGPT

- ChatGPT может помочь создавать разнообразные варианты домашних заданий.
    - Определить тип задания (вопрос, эссе, математическая задача, задание на программирование и т.д.)
    - Разбить задачу на подзадачи. Возможно, будет необходимо составить один пример домашнего задания.
    - Прописать пошагово что вас интересует: тема, основные элементы задания, количество вопросов в задании.
- 



# Мои примеры

## 1 кейс.

### • Запрос:

- Составить 5 тестовых вопросов.
- 1. Тема: Циклы for и while в Python
- 2. Тип вопроса: вопросы с единственным ответом, 5 вариантов ответов.
- 3. К каждому вопросу делать пояснение (комментарий).
- 4. К каждому вопросу должен быть вариант ответа.



You

Составить 5 тестовых вопросов.

1. Тема: Циклы for и while в Python
2. Тип вопроса: вопросы с единственным ответом, 5 вариантов ответов.
3. К каждому вопросу делать пояснение (комментарий).
4. К каждому вопросу должен быть вариант ответа.



ChatGPT

Отлично, давайте создадим 5 тестовых вопросов на тему циклов `for` и `while` в Python.

К каждому вопросу я добавлю 5 вариантов ответов и пояснение.

### Вопрос 1

Что делает следующий код?

python

Copy code

```
for i in range(5):  
    print(i)
```

Варианты ответов:

1. Выводит числа от 1 до 5.
2. Выводит числа от 0 до 4.
3. Создаёт список из чисел от 0 до 4.
4. Выводит числа от 0 до 5.
5. Выводит числа от 1 до 4.

**Правильный ответ:** 2. Выводит числа от 0 до 4.

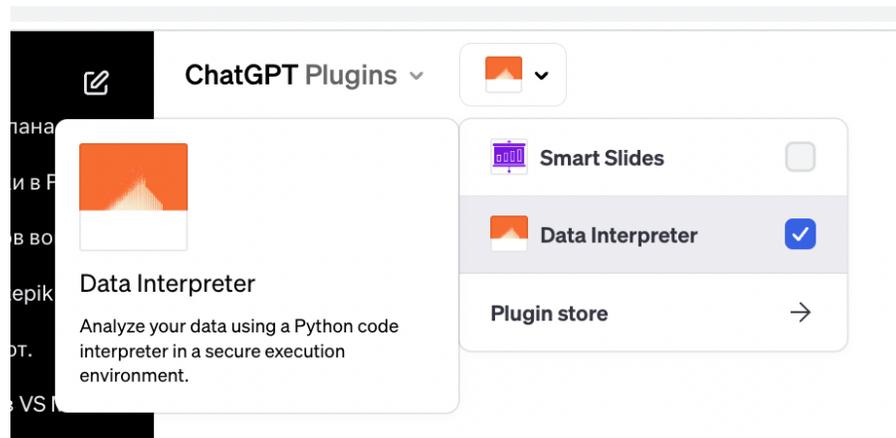
**Пояснение:** Цикл `for` использует функцию `range(5)`, которая генерирует последовательность от 0 до 4.

# Мои примеры

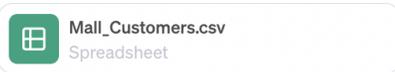
## 2 кейс. Домашнее задание с использованием ChatGPT

- Нейронные сети могут помочь в обучении программированию и в самом процессе программирования.
  - **Задача.**
    1. Дана таблица с данными (данные посетителей магазина: id, пол, возраст, доход, рейтинг трат). С помощью плагина DataInterpreter:
    2. Сделать описание таблицы данных.
    3. Найти зависимость и построить визуализацию зависимости дохода от возраста.
    4. Найти зависимость и построить визуализацию зависимости трат от дохода и возраста.
    5. Сделать выводы.
    6. Проверьте все выводы и данные, если есть ошибки, исправьте их.
-

# Мои примеры 2 кейс.



You



Сделать описание таблицы данных (на русском языке).

< 2 / 2 >

ChatGPT

Файл данных "Mall\_Customers.csv" содержит следующую структуру:

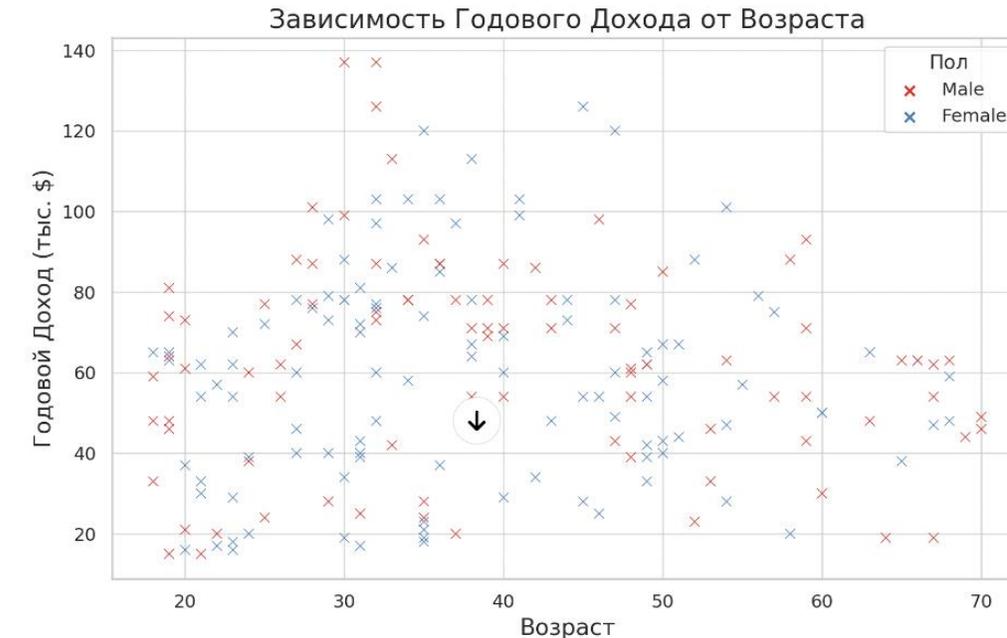
1. **CustomerID**: Уникальный идентификатор клиента.
2. **Genre**: Пол клиента (Male - мужчина, Female - женщина).
3. **Age**: Возраст клиента.
4. **Annual Income (k\$)**: Годовой доход клиента, выраженный в тысячах долларов.
5. **Spending Score (1-100)**: Оценка трат клиента, выраженная в баллах от 1 до 100.

Эта таблица предназначена для анализа покупательского поведения клиентов торгового центра. Каждая строка представляет отдельного клиента и содержит информацию о его поле, возрасте, годовом доходе и оценке трат, что может быть полезно для маркетинговых исследований и целевой рекламы. [-]

You

2. Найти зависимость и построить визуализацию зависимости дохода от возраста.

ChatGPT



На представленной визуализации показана зависимость годового дохода (выраженного в тысячах долларов) от возраста клиентов. Используются разные цвета для обозначения пола клиентов: мужчины и женщины.

Из графика можно сделать несколько наблюдений:

- Нет явной линейной зависимости между возрастом и годовым доходом. Данные распределены довольно равномерно по всему диапазону возрастов.
- Клиенты в возрастной группе от 20 до 40 лет имеют различный уровень дохода, от низкого до высокого.
- В возрастной группе от 40 лет и старше также наблюдается разнообразие в доходах, но максимальные значения дохода несколько ниже, чем у более молодых клиентов.

Эти наблюдения могут быть полезны для понимания потребностей и покупательской способности различных возрастных групп клиентов. [-]

# Мои примеры

## 2 кейс. Написание чат-бота для телеграмм.

**Задание: Создание Телеграм-бота для Управления Задачами**

**Цель:** Разработать телеграм-бота, который помогает пользователю управлять его личными задачами.

**Основные требования:**

- **Основные функции бота:**
    - **Добавление задачи:** Пользователь должен иметь возможность добавить новую задачу в свой список.
    - **Просмотр списка задач:** Пользователь может просмотреть все свои текущие задачи.
    - **Удаление одной задачи:** Пользователь может удалить конкретную задачу из списка.
    - **Удаление всех задач:** Пользователь может удалить все задачи из списка.
  - **Интерфейс:**
    - Реализуйте интерфейс с помощью клавиатурных кнопок в Телеграме.
    - Предусмотрите кнопки для каждой из основных функций (Добавление, Просмотр, Удаление одной задачи, Удаление всех задач).
-

**You**  
Создадим чат-бота в телеграмм (pyTelegramBotAPI). Должны быть команды /start и /info

**ChatGPT**  
Создание чат-бота в Telegram с использованием библиотеки `pyTelegramBotAPI` включает в себя несколько основных шагов. Вот краткое руководство по созданию такого бота с командами `/start` и `/info`:

### Шаг 1: Создание бота в Telegram

1. Откройте **Telegram** и найдите `@BotFather`.
2. Создайте нового бота, используя команду `/newbot`.
3. Следуйте инструкциям от `BotFather` для создания бота. Вам будет предоставлен токен, который необходим для доступа к API Telegram.

### Шаг 2: Установка pyTelegramBotAPI

Установите библиотеку `pyTelegramBotAPI`, используя pip:

```
bash Copy code
pip install pyTelegramBotAPI
```

### Шаг 3: Написание кода бота

Создайте Python-скрипт для вашего бота. Вот примерный код:

```
python Copy code
import telebot

TOKEN = 'ВАШ_ТОКЕН_БОТА'
bot = telebot.TeleBot(TOKEN)

@bot.message_handler(commands=['start'])
def send_welcome(message):
    bot.reply_to(message, "Привет! Я ваш тестовый бот.")

@bot.message_handler(commands=['info'])
def send_info(message):
    bot.reply_to(message, "Это бот для демонстрации. Создан с использованием pyTelegr

bot.polling()
```

Замените `'ВАШ_ТОКЕН_БОТА'` на токен, который вы получили от `BotFather`.

### Шаг 4: Запуск бота

Запустите скрипт, и ваш бот начнет работать. Отправьте команды `/start` или `/info` в чат с вашим ботом в Telegram, чтобы проверить его работу.

**You**  
Добавить клавиатуру (Добавление, Просмотр, Удаление одной задачи, Удаление всех задач) отображаемую по нажатию на команду start.

**ChatGPT**  
Чтобы добавить клавиатуру с кнопками (например, "Добавление", "Просмотр", "Удаление одной задачи", "Удаление всех задач") в вашего Telegram бота на Python с использованием библиотеки `pyTelegramBotAPI`, вам нужно использовать `telebot.types.ReplyKeyboardMarkup` и `telebot.types.KeyboardButton`. Вот как вы можете это сделать:

### Шаг 1: Импорт необходимых модулей

Убедитесь, что вы импортировали `ReplyKeyboardMarkup` и `KeyboardButton`:

```
python Copy code
from telebot import types
```

### Шаг 2: Создание клавиатуры

Измените обработчик команды `/start`, чтобы добавить клавиатуру:

```
python Copy code
@bot.message_handler(commands=['start'])
def send_welcome(message):
    # Создание клавиатуры
    markup = types.ReplyKeyboardMarkup(resize_keyboard=True, one_time_keyboard=True)
    btn1 = types.KeyboardButton('Добавление')
    btn2 = types.KeyboardButton('Просмотр')
    btn3 = types.KeyboardButton('Удаление одной задачи')
    btn4 = types.KeyboardButton('Удаление всех задач')
    markup.add(btn1, btn2, btn3, btn4)

    bot.send_message(message.chat.id, "Выберите действие:", reply_markup=markup)
```

### Шаг 3: Обработка нажатий на кнопки

Чтобы обработать нажатия на кнопки, вам нужно добавить обработчики текстовых сообщений. Вот пример:

```
python Copy code
@bot.message_handler(func=lambda message: message.text == "Добавление")
def handle_task_add(message):
    bot.reply_to(message, "Функция добавления задачи.")

@bot.message_handler(func=lambda message: message.text == "Просмотр")
def handle_task_view(message):
    bot.reply_to(message, "Функция просмотра задач.")

# Добавьте аналогичные обработчики для других кнопок
```

# Мои примеры

## 3 кейс. Оценивание домашних заданий с ChatGPT

- Оценивание домашних заданий с помощью ChatGPT может быть очень полезным и экономящим время. Рассмотрим пример того, как можно предоставить студенту обратную связь на написанный код (проверка на код-стайл и т.д.).
- Для начала стоит прописать критерии оценки и на что обратить внимание. ChatGPT стоит предоставить условие, решение и критерии проверки (оценки) работы. Не забудьте проверить результат.

### ChatGPT

Давайте проанализируем представленный код с точки зрения стиля, улучшений и правильности решения задачи.

#### Ошибки код-стайла

1. **Использование пространства имен `std`**: Хотя это распространенная практика, особенно в учебных материалах, лучше избегать `using namespace std;` в более крупных или серьезных проектах, чтобы избежать потенциальных конфликтов имен.
2. **Объявление массивов с переменной длиной**: В C++ стандартный способ создания массивов требует заранее известного размера. Использование `int b[a];` и `int q[k];`, где `a` и `k` являются переменными, не является стандартным C++. Вместо этого следует использовать `std::vector`.
3. **Форматирование**: Код должен быть оформлен единообразно для улучшения читаемости. Например, пробелы вокруг операторов (`for(int i = 0; i < a; i++)`) и после запятых.

#### Комментарии по улучшению кода

1. **Использование `std::vector`**: Вместо массивов фиксированной длины следует использовать `std::vector`, что позволит избежать проблем с выделением памяти и улучшит переносимость кода.
2. **Оптимизация поиска**: Текущая реализация использует двойной цикл для подсчета элементов, что приводит к временной сложности  $O(nk)$ . Можно оптимизировать это, используя бинарный поиск для каждого запроса, что снизит временную сложность до  $O(k \log n)$ .
3. **Именование переменных**: Использование более описательных имен переменных может улучшить читаемость кода. Например, вместо `a` и `b` использовать `vectorSize` и `elements`.

#### Проверка кода на правильность

Код неправильно решает задачу в соответствии с условием. В условии задачи требуется использовать бинарный поиск для определения количества элементов, что не реализовано в представленном коде. Вместо этого используется простой перебор, что значительно менее эффективно, особенно для больших векторов и большого количества запросов.

# Мои примеры

## 4 кейс. Программа дисциплины



ChatGPT

### Содержание Учебной Дисциплины "Python для начинающих"

1. Введение. Основы, print(), input() и типы данных

- **Планируемые результаты обучения:** Знание (1), Понимание (2)
- **Форма контроля:** Домашнее задание на основы синтаксиса Python, использование print() и input(), понимание различных типов данных.

2. Операторы (унарные, бинарные), условия

- **Планируемые результаты обучения:** Применение (3), Анализ (4)
- **Форма контроля:** Домашнее задание на использование операторов и условных конструкций.

3. Петли

- **Планируемые результаты обучения:** Применение (3), Анализ (4)
- **Форма контроля:** Домашнее задание на написание и анализ циклов.

4. Коллекции (список)

- **Планируемые результаты обучения:** Понимание (2), Применение (3)
- **Форма контроля:** Домашнее задание на работу со списками.

5. Коллекции (набор, кортеж, словарь)

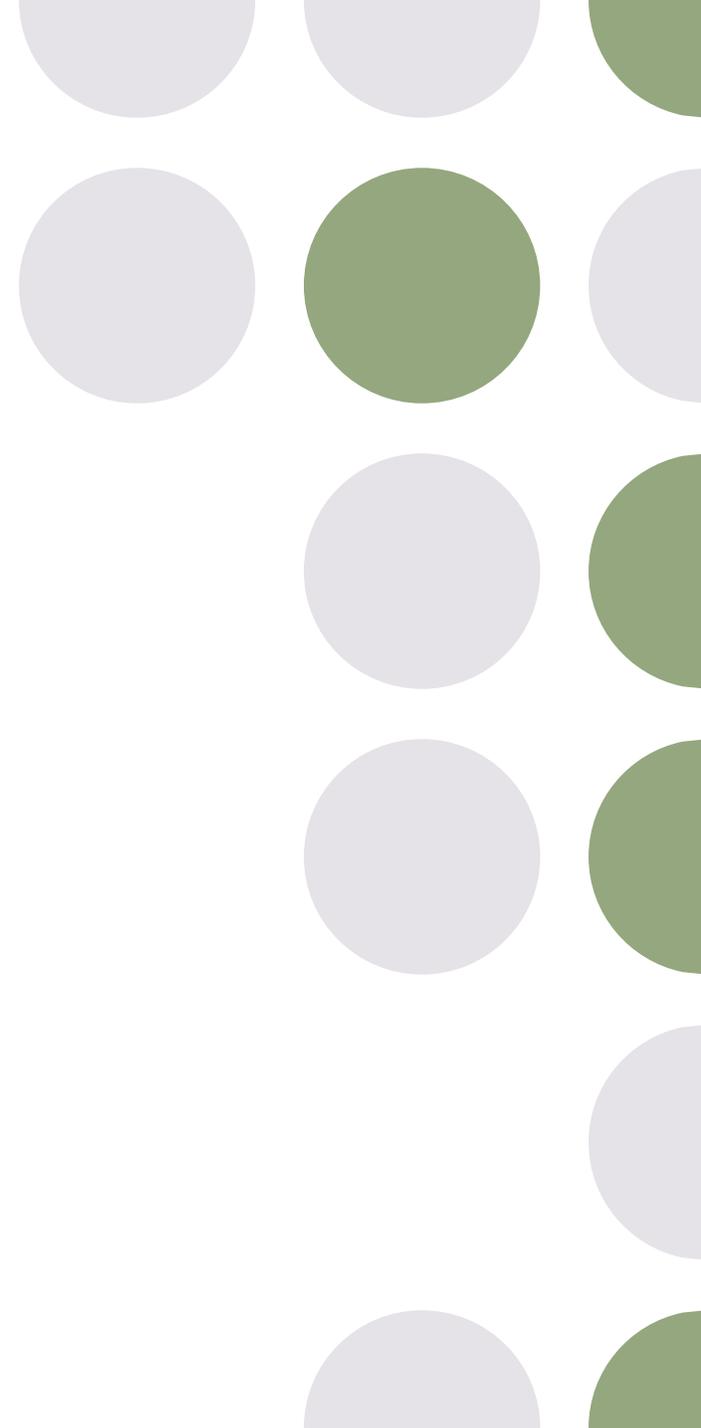
- **Планируемые результаты обучения:** Понимание (2), Применение (3)
- **Форма контроля:** Домашнее задание на работу с различными типами коллекций.

6. Функции

- **Планируемые результаты обучения:** Синтез (5), Анализ (4)
- **Форма контроля:** Домашнее задание на создание и использование функций.

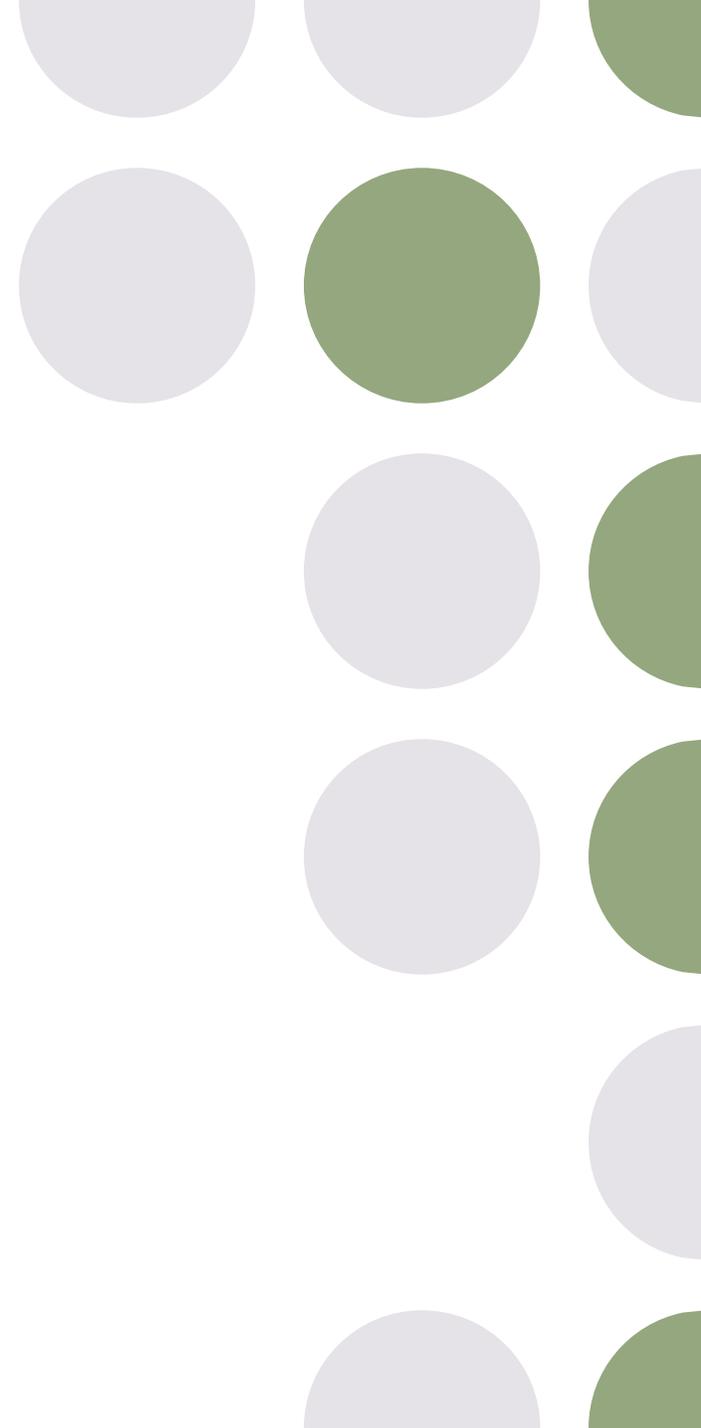
# Академическое мошенничество

- это действия, нарушающие принципы академической добросовестности, цель которых – получить незаслуженные академические преимущества.
    - Плагиат
    - Выполнение работы третьими лицами
    - Сочинение/подделка фактов и данных
    - Использование запрещенных материалов
    - Сдача одной и той же работы
    - и т.д.
- 



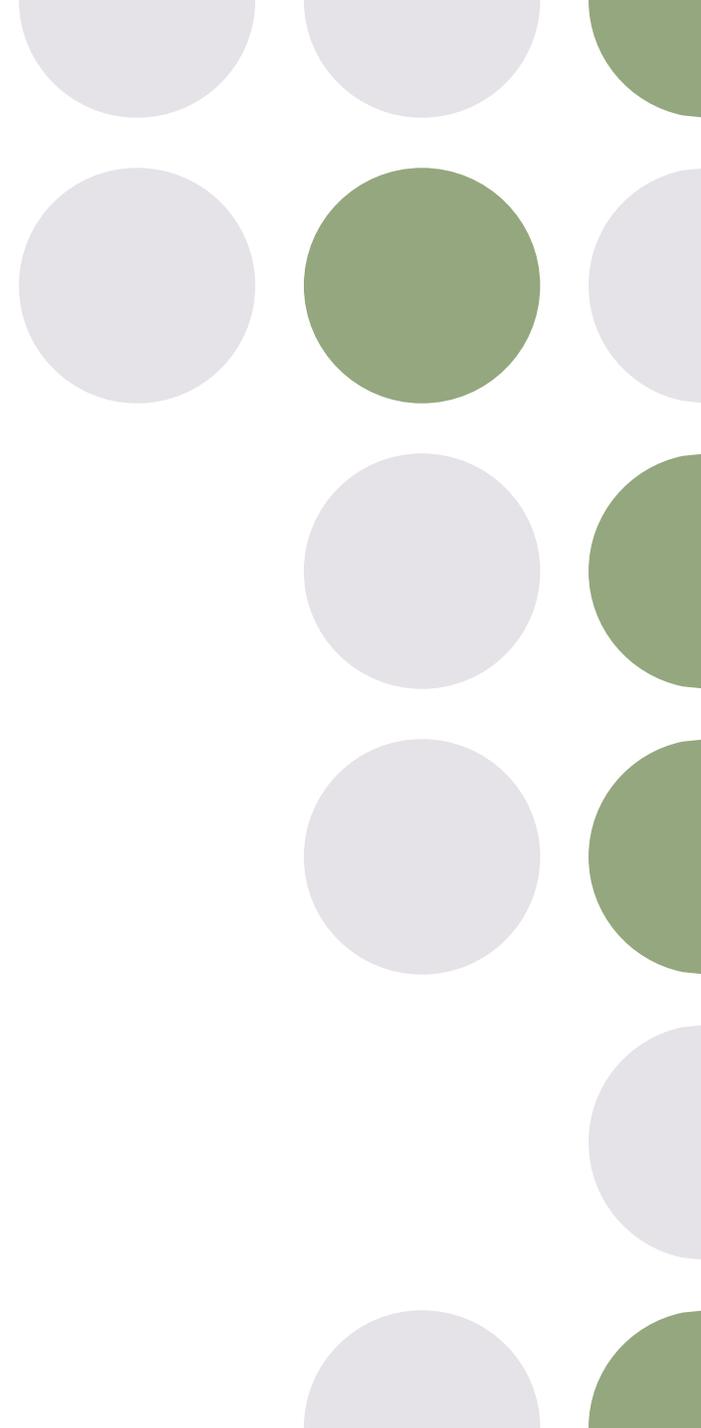
# Минусы

- Нарушение академической этики
  - Отсутствие обучения
  - Дисциплинарные последствия
  - Репутационные риски
  - И т.д.
- 



# Плюсы

- Экономия времени(???)
  - Новые технологии(???)
  - Обучение(???)
- 



# Технологии и мошенничество

- с развитием технологий способы академического мошенничества стали более изощренными, включая использование специализированного программного обеспечения

