

Номинация: Преподавание учебного курса и технологии искусственного интеллекта
Автор: Горденко М.К., ФКН
Тема: Использование ChatGPT в учебном процессе (обучение программированию)

Важно! Все созданные материалы с помощью **ChatGPT** требуют проверки.

Оглавление

<i>1 кейс. Создание вариантов домашних заданий с ChatGPT.....</i>	2
Тестовые вопросы	2
Задача на программирование	3
Задача на анализ данных	4
<i>2 кейс. Домашнее задание с использованием ChatGPT.....</i>	7
Задание. Анализ данных на Python с использованием ChatGPT и надстройки для анализа данных.....	7
Критерии оценки задания.....	10
Подсказка. Как составлялись задания, чтобы в них ChatGPT ошибся и студенту пришлось дорабатывать код?.....	10
Задание. Написание чат-бота для телеграмм.....	11
Критерии оценки задания.....	14
Подсказка. На что обращать внимание при оценивании таких работ?.....	15
Как создать или переработать уже существующее задание, чтобы фокус был на рефлексии по поводу использования ChatGPT в учебных целях.....	16
<i>3 кейс. Оценивание домашних заданий с ChatGPT.....</i>	17
<i>4 кейс. Помощь в создании курсов</i>	19
Создание конспектов.....	19
<i>5 кейс. Помощь в наполнении программы учебной дисциплины</i>	23
Начальные требования	23
Запрос 1. Генерация общего описания по шаблону.....	24
Запрос 2. Генерация целей, результатов освоения дисциплины.....	24
Запрос 3. Содержание учебной дисциплины.....	25
Запрос 4. Оценивание.....	26
<i>Приложение 1. Шаблон ПУДа</i>	29

1 кейс. Создание вариантов домашних заданий с ChatGPT

ChatGPT может помочь создавать разнообразные варианты домашних заданий.

Вот пошаговый план, который может помочь составить домашние задания:

1. Определить тип задания (вопрос, эссе, математическая задача, задание на программирование и т.д.)
2. Разбить задачу на подзадачи. Возможно, будет необходимо составить один пример домашнего задания.
3. Прописать пошагово что вас интересует: тема, основные элементы задания, количество вопросов в задании.

Тестовые вопросы

Рассмотрим пример, как можно составить тестовые вопросы (на примере программирования на Python). Наш план и будет являться запросом для ChatGPT.

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.

Смотрим на ответ ChatGPT (приведен только первый вопрос):



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.

Подобным образом могут быть составлены вопросы с кратким ответом, с открытым вариантом ответа или с несколькими правильными вариантами ответа. Стоит только

поменять запрос и задать критерии этих вопросов. **Однако, всегда стоит проверять на верность созданные с помощью ChatGPT материалы.**

Задача на программирование

Студенты часто решают задачи на программирование и может потребоваться несколько похожих вариантов. Сделаем запрос, который позволит создать подобного вида задачу.

Запрос:

Составить задание по программированию на языке Python

1. Тема: Циклы for и while в Python.

2. Условие должно содержать: легенду, формат входных данных, формат выходных данных.

3. К заданию сделать примеры входных и выходных данных (3 штуки).

4. Написать решение задачи (включая считывание данных с клавиатуры).

Смотрим на ответ ChatGPT:



You

Составить задание по программированию на языке Python

1. Тема: Циклы for и while в Python.

2. Условие должно содержать: легенду, формат входных данных, формат выходных данных.

3. К заданию сделать примеры входных и выходных данных (3 штуки).

4. Написать решение задачи (включая считывание данных с клавиатуры).



ChatGPT

Задание по программированию: "Расчёт суммы ряда"

Легенда:

В магическом мире чисел есть интересная закономерность: сумма ряда чисел от 1 до N, возведённых в квадрат, рассчитывается по особой формуле. Ваша задача - написать программу, которая будет вычислять эту сумму для заданного числа N, используя циклы `for` и `while` в Python.

Формат входных данных:

Одно целое число N ($1 \leq N \leq 1000$).

Формат выходных данных:

Одно целое число - сумма квадратов чисел от 1 до N.

Примеры входных и выходных данных:

1. Входные данные: `3`
Выходные данные: `14`
Объяснение: $1^2 + 2^2 + 3^2 = 1 + 4 + 9 = 14$
2. Входные данные: `5`
Выходные данные: `55`
Объяснение: $1^2 + 2^2 + 3^2 + 4^2 + 5^2 = 1 + 4 + 9 + 16 + 25 = 55$
3. Входные данные: `2`
Выходные данные: `5`
Объяснение: $1^2 + 2^2 = 1 + 4 = 5$

Решение задачи:

```
python Copy code  
  
# Считывание данных с клавиатуры  
N = int(input("Введите число N: "))  
  
# Решение с использованием цикла for  
sum_for = 0  
for i in range(1, N + 1):  
    sum_for += i**2  
  
# Решение с использованием цикла while  
sum_while = 0  
i = 1  
while i <= N:  
    sum_while += i**2  
    i += 1  
  
# Вывод результатов  
print("Сумма с использованием цикла for:", sum_for)  
print("Сумма с использованием цикла while:", sum_while)
```

Было составлено верное задание. Однако, если посмотреть на решение, то в нем присутствуют некоторые несоответствия с форматом вывода, который прописан в условии. Однако, это не так страшно, можно поправить самостоятельно и сэкономить время.

Бывает, когда нужны более сложные задания, в этом случае, нужно предоставить пример (шаблон), по которому стоит создать задание.

Таким же образом могут быть созданы задачи в разных разметках (html, markdown) и с генераторами тестов (решений).

Задача на анализ данных

Сейчас данные генерируются в больших объемах и анализ данных помогает понять и интерпретировать эти данные, делая их полезными для принятия решений. Он позволяет компаниям и организациям принимать более обоснованные и точные решения, за счет анализа закономерностей, может помочь выявить узкие места и неэффективные процессы в операциях, повышая эффективность и сокращая издержки и т.д. Из-за этих и многих других причин анализ данных становится все более важным навыком в современном мире.

С помощью ChatGPT могут быть составлены задачи на анализ данных для решения с помощью различных инструментов (excel, python и т.д.).

Запрос:

Составить задание по анализу данных на Python.

1. Есть таблица из 1000 элементов, содержащая описание пользователей в социальной сети (id, username, name, surname, followers_count, country, city, salary, education).

2. По названию столбцов нужно сделать описание данных таблицы.

2. Сделай 3 задания на анализ и выявление закономерностей в данных.

3. Сделай 3 задания на построение графиков и диаграмм.

4. Каждое задание опиши подробно, а также сделай подсказки.

5. Предоставь эталонное решение.

Смотрим на ответ ChatGPT:



You

Составить задание по анализу данных на Python.

1. Есть таблица из 1000 элементов, содержащая описание пользователей в социальной сети (id, username, name, surname, followers_count, country, city, salary, education).

2. По названию столбцов нужно сделать описание данных таблицы.

2. Сделай 3 задания на анализ и выявление закономерностей в данных.

3. Сделай 3 задания на построение графиков и диаграмм.

4. Каждое задание опиши подробно, а также сделай подсказки.

5. Предоставь эталонное решение.



ChatGPT

Для создания задания по анализу данных на Python с использованием таблицы пользователей в социальной сети, давайте сначала опишем данные, а затем составим шесть заданий: три на анализ и выявление закономерностей и три на построение графиков и диаграмм.

Описание данных таблицы:

1. **id:** Уникальный идентификационный номер пользователя.
2. **username:** Имя пользователя, используемое в социальной сети.
3. **name:** Имя пользователя.
4. **surname:** Фамилия пользователя.
5. **followers_count:** Количество подписчиков у пользователя.
6. **country:** Страна, в которой находится пользователь.
7. **city:** Город, в котором находится пользователь.
8. **salary:** Зарплата пользователя.
9. **education:** Образование пользователя.

Задания на анализ и выявление закономерностей:

Задание 1: Анализ взаимосвязи образования и зарплаты.

- **Описание:** Исследуйте, как уровень образования влияет на среднюю зарплату пользователей.
- **Подсказки:** Используйте группировку по образованию и расчет среднего значения зарплаты.
- **Эталонное решение:**

```
python
```

[Copy code](#)

```
average_salary_by_education = df.groupby('education')['salary'].mean()
```

Задание 2: Анализ распределения пользователей по странам.

- **Описание:** Определите, как распределяются пользователи по странам и какие страны наиболее популярны.
- **Подсказки:** Используйте метод `value_counts()` для подсчета числа пользователей в каждой стране.
- **Эталонное решение:**

```
python
```

[Copy code](#)

```
users_per_country = df['country'].value_counts()
```

Задание 3: Корреляция между количеством подписчиков и зарплатой.

- **Описание:** Исследуйте, существует ли связь между количеством подписчиков и зарплатой пользователей.

2 кейс. Домашнее задание с использованием ChatGPT

Нейронные сети могут помочь в обучении программированию и в самом процессе программирования.

Задание. Анализ данных на Python с использованием ChatGPT и надстройки для анализа данных.

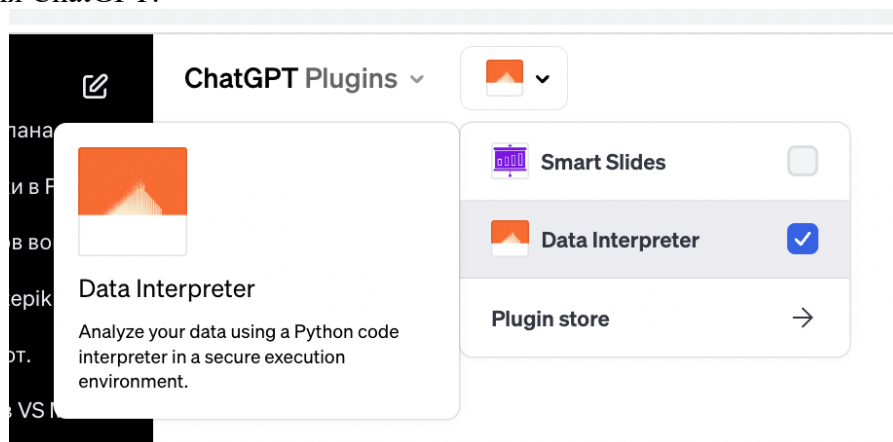
На курсе программирования (Программирование на Python) со студентами были изучены базовые возможности языка программирования, а также библиотеки pandas (для работы с таблицами) и matplotlib (для построения графиков). Библиотеки pandas и matplotlib были изучены на базовом уровне. После чего студентам было предложено использовать возможности ChatGPT для решения домашнего задания и изучения дополнительных возможностей библиотек.

Задача.

Дана таблица с данными (данные посетителей магазина: id, пол, возраст, доход, рейтинг трат). С помощью плагина DataInterpreter:

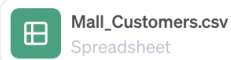
1. Сделать описание таблицы данных.
2. Найти зависимость и построить визуализацию зависимости дохода от возраста.
3. Найти зависимость и построить визуализацию зависимости трат от дохода и возраста.
4. Сделать выводы.
5. Проверьте все выводы и данные, если есть ошибки, исправьте их.

Пример использования ChatGPT.



Сначала загрузим датасет в ChatGPT и попросим описать его.

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.

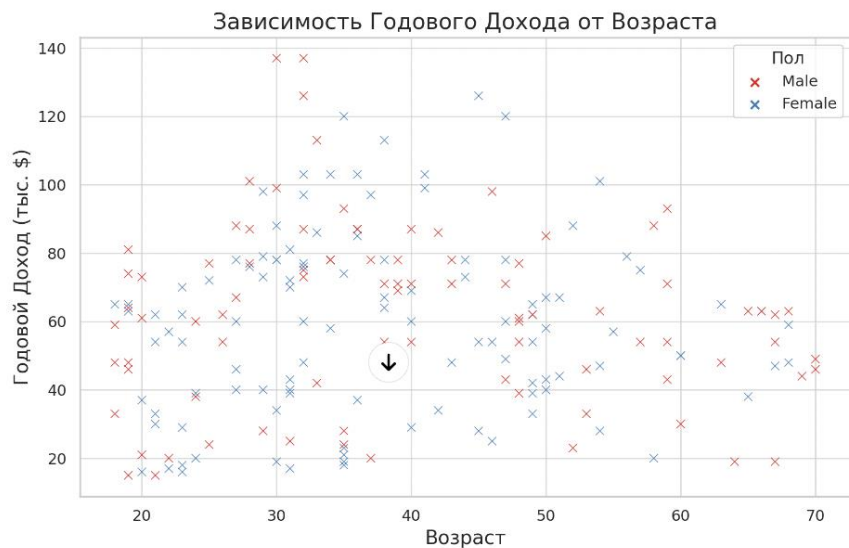
Эта таблица предназначена для анализа покупательского поведения клиентов торгового центра. Каждая строка представляет отдельного клиента и содержит информацию о его поле, возрасте, годовом доходе и оценке трат, что может быть полезно для маркетинговых исследований и целевой рекламы. [?]

Пошагово выполним задания.

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. [?]



You

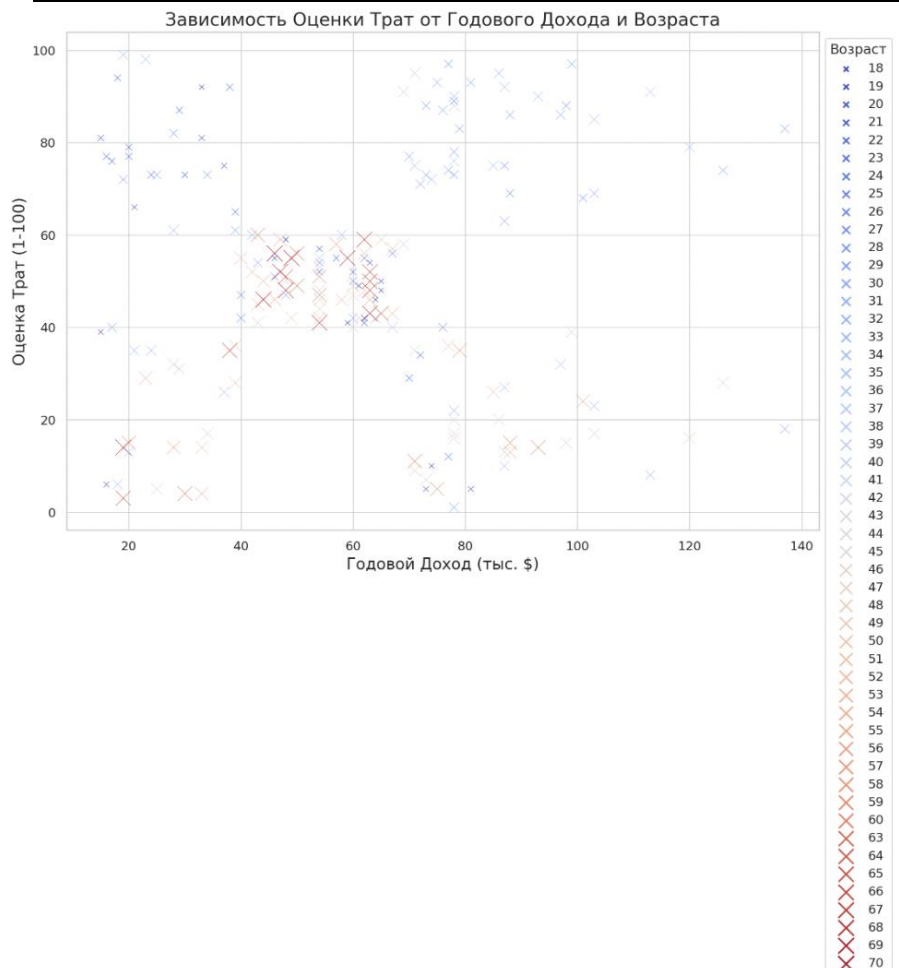
3. Найти зависимость и построить визуализацию зависимости трат от дохода и возраста. Оставь код на Python



ChatGPT

Finished analyzing

```
python Copy code  
  
# Creating a scatter plot to visualize the relationship between Spending Score, Annual  
plt.figure(figsize=(12, 8))  
  
# Using a scatter plot for visualization  
sns.scatterplot(x='Annual Income (k$)', y='Spending Score (1-100)', hue='Age', size=  
                sizes=(20, 200), data=data, palette="coolwarm", legend="full")  
  
# Adding titles and labels  
plt.title('Зависимость Оценки Трат от Годового Дохода и Возраста', fontsize=16)  
plt.xlabel('Годовой Доход (тыс. $)', fontsize=14)  
plt.ylabel('Оценка Трат (1-100)', fontsize=14)  
plt.legend(title='Возраст', bbox_to_anchor=(1, 1))  
  
# Displaying the plot  
plt.show()
```



На представленном графике показана зависимость оценки трат (баллы от 1 до 100) от годового дохода (выраженного в тысячах долларов), а также от возраста клиентов. Возраст клиентов отображается с помощью цветовой гаммы и размера точек: чем темнее и больше точка, тем старше клиент.

Из графика можно сделать несколько наблюдений:

- Существует группа клиентов с высокой оценкой трат (ближе к 100) и средним уровнем дохода (около 40-60 тысяч долларов). Эта группа в основном молодежь.
- Клиенты с высоким доходом (более 70 тысяч долларов) имеют различные уровни трат, независимо от возраста.
- Существует группа с низким уровнем дохода и низкой оценкой трат, в которой преобладают более старшие клиенты.

Эта визуализация полезна для понимания того, как годовой доход и возраст клиентов влияют на их расходные привычки.[-]

Видим, что с помощью DataInterpreter мы выполнили анализ данных, осталось только вставить сделанные части в IDE (среду исполнения кода) и запустить их. **Но не все так гладко, ChatGPT тоже ошибается, и задания составляются специально таким образом, чтобы студент мог столкнуться с этими ошибками и исправить их. Более того, инструмент позволяет сделать лишь поверхностный анализ. Для более сложных визуализаций и закономерностей требуется грамотно составлять уточняющие запросы, в т.ч. четко задавать какие инструменты стоит использовать.**

Обучающийся сдает не только написанный код, но и пошаговое описание своего взаимодействия с ИИ. Таким образом, должно быть сдано:

- Код
- Описание запросов нейросети (протокол запросов)
- Рефлексия с ответами на вопросы (рефлексия приведена в критериях и подсказках)

Критерии оценки задания

Оценка такого задания требует как оценку правильности выполнения поставленной задачи, так и оценку взаимодействия с ИИ (т.н. ИИ-грамотность).

При разработке критериев рекомендуется придерживаться следующей структуры:

1. Правильность и корректность ответов на вопросы.
2. Качество кода.
3. Оценка навыков ИИ-грамотности:
 - а. Оценка запросов (четкость, понятность, лаконичность)
 - б. Критическое осмысление результатов (достоверность, полнота, предвзятость)
4. Рефлексия (ответы на вопросы о рисках, достоинствах и недостатках применения ИИ в данном задании).

Подсказка. Как составлялись задания, чтобы в них ChatGPT ошибся и студенту пришлось дорабатывать код?

Важно понимать, что языковые модели способны делать ошибки в самых неожиданных местах и этим можно пользоваться, чтобы обучающиеся столкнулись с ошибками и попытались их исправить, а также не верили безусловно нейросетям.

Вот свод правил, выявленных эмпирическим путем:

Правило 1.

Языковые модели несколько похожи на младших специалистов и могут делать ошибки в самых неожиданных местах и именно поэтому им нужны четкие и детальные инструкции

по тому, как сделать то или иное задание. Поэтому, чтобы у студента была возможность получить ошибку или же не получить сразу нужный результат, то задание должно быть четким, но не детализированным до мельчайших подробностей.

Правило 2.

На данный момент языковые модели не очень хорошо работают с творческими задачами, именно поэтому, если задание будет творческим или таким, где необходимо продумать архитектуру, взаимосвязи между компонентами и т.д. велик риск, что модель ошибется, а студенту придется самостоятельно проделать этот этап.

Правило 3.

Учитывая тот факт, что GPT4 обучен на данных до апреля 2023 года, то задания о событиях или библиотеках (их обновления), которые случились после этих сроков, модели не выдадут корректного результата.

Правило 4.

Языковые модели дают неплохие и даже очень хорошие результаты если задание простое, не требует творческого подхода и смешивания нескольких разных тем. Поэтому в комплексных заданиях можно часто получить ошибку.

Правило 5.

Всегда необходимо проверять, а сможет ли модель решить наше задание. Можно это делать следующим образом:

1. Проверить, что выдаст модель, если вставить результат целиком.
2. Что будет если вставлять задание по кусочкам.

Если модель не справилась, то велика вероятность, что для его решения обучающемуся придется потрудиться.

Правило 6.

Простые и не объемные задания модели решают легко. Если задание становится объемным, то модель очень часто пропускает определенные шаги или вовсе решает не ту задачу. Поэтому вместо трех заданий можно сделать всего одно, но большего объема.

Правило 7.

Существующие модели не могут составлять схем, картинок или иллюстраций должного уровня. Поэтому можно просить переработать полученную информацию в схему.

Задание. Написание чат-бота для телеграмм.

Исходя из предоставленного кода, задание для студента может быть сформулировано следующим образом:

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).
- **Обработка команд:**
 - Реализуйте обработчик команд /start, который будет отображать клавиатуру с опциями.
 - Реализуйте функцию обратного вызова для обработки выбора пользователя.
- **Обработка исключений:**
 - Убедитесь, что бот корректно обрабатывает ситуации, когда список задач пуст.
- **Интеграция с базой данных:**
 - Используйте SQLite для хранения данных.
 - Создайте таблицу planner с полями для хранения идентификатора пользователя и задачи.

Обучающийся сдает не только написанный код, но и пошаговое описание своего взаимодействия с ИИ. Таким образом, должно быть сдано:

- Код
- Описание запросов нейросети (протокол запросов)
- Рефлексия с ответами на вопросы (рефлексия приведена в критериях и подсказках)

Решение с помощью ChatGPT могло бы выглядеть следующим образом. Отметим, что можно на основе этого задания создать варианты, а также задание достаточно объемное и придется позадавать запросы, т.е. его сложно решить за один запрос, а также могут возникать ошибки, которые придется исправлять.



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 pyTelegramBotAPI.")

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
```

И т.д. Таким образом, пошагово может быть создан работоспособный Телеграм-бот и другие программы.

Критерии оценки задания

Оценка такого задания требует как оценку правильности выполнения поставленной задачи, так и оценку взаимодействия с ИИ (т.н. ИИ-грамотность). Кроме того, учитывая комплексность задания, могут возникать ошибки и очень важна рефлексия обучающегося по тому, почему эти ошибки возникли и можно ли было задать вопрос ИИ иначе, чтобы избежать ошибки.

При разработке критериев рекомендуется придерживаться следующей структуры:

1. Правильность и корректность ответов на вопросы.
2. Качество кода.
3. Рефлексия по поводу ошибочных ответов ИИ:
 - a. Почему возникла ошибка
 - b. Как исправить ошибку
 - c. Можно ли сделать запрос другим, чтобы ошибки не было
4. Оценка навыков ИИ-грамотности:
 - a. Оценка запросов (четкость, понятность, лаконичность)
 - b. Критическое осмысление результатов (достоверность, полнота, предвзятость)
5. Рефлексия (ответы на вопросы о рисках, достоинствах и недостатках применения ИИ в данном задании).

Подсказка. На что обращать внимание при оценивании таких работ?

Важно понимать, что даже ограничивая обучающихся в использовании ИИ при написании работ мы неизбежно будем сталкиваться с тем, что они его используют. Более того, это популярный инструмент, который пригодится обучающимся в будущем, поэтому важно научиться грамотно с ним работать и не ждать чудес.

Подобные подсказки были также разработаны и другими университетами (например, подсказки университета Австрии¹)

Подсказка 1. Задание должно оценивать как правильность его выполнения, так и то, как студент взаимодействовал с ИИ, включая:

- этику общения с ИИ;
- критическое мышление при взаимодействии с ИИ;

Подсказка 2. Можно попросить студентов описать какие риски, ограничения и преимущества при использовании ИИ в данном задании.

Так, например, при разработке чат-бота преимущества очевидны (ускорение написания кода, стилистически верный код и т.д.), но есть и существенный ряд минусов и недостатков связанный с неверной интерпретацией некоторых функций, неверным или ошибочным написанием кода, а также увеличением времени разработки (в случае, если ИИ создал код, в котором сложно найти ошибку).

Подсказка 3. Убрать живого человека из оценивания нельзя, более того, для таких работ рекомендуется проводить устные защиты или беседы, чтобы понять, насколько обучающийся разобрался с тем, что выдала нейросеть.

Подсказка 4. В сдаваемом материале обучающимся стоит показать то, как были предприняты основные решение, ход мыслей, подсказки нейросети. Т.е. студент сдает не только написанный код, но и пошаговое описание своего взаимодействия с ИИ.

Подсказка 5. Не все виды работы стоит выполнять с ИИ. Необходимо четко определить и разграничить где его допустимо, а где недопустимо использовать в рамках курса.

¹ <https://www.teqsa.gov.au/sites/default/files/2023-09/assessment-reform-age-artificial-intelligence-discussion-paper.pdf>

Как создать или переработать уже существующее задание, чтобы фокус был на рефлексии по поводу использования ChatGPT в учебных целях.

Подсказка 1. Прямо в конце задания попросить студентов провести рефлексию:

- Как вы использовали ChatGPT для выполнения этого задания?
- Какие преимущества и ограничения ChatGPT вы заметили в процессе работы?
- Какие стратегии оценки достоверности и качества информации, предоставленной ChatGPT, вы применяли?
- В каких областях ChatGPT оказался наиболее полезным, и где он не смог в полной мере помочь?
- Выявили ли вы какие-либо особенности при задании вопросов ChatGPT? Можете ли дать советы, как правильно задавать вопросы?
- Как вы считаете, какие навыки важно развивать для эффективного использования подобных технологий?

Подсказка 2. Попросить обучающегося записывать (протоколировать, копировать) вопросы и ответы для нейросети, а после выделить:

- Лучшие ответы.
- Худшие ответы (ответы с ошибками).

Подсказка 3. Стоит помнить, что лучше всего, чтобы задание было:

- комплексным (содержало несколько тем),
- творческим (где обучающемуся нужно принять ряд обоснованных решений),
- объемным и более сложным, чем базовые задачи курса, т.к. с базовыми простыми задачами нейросети справляются практически без подсказок.

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.

Рассмотрим пример.

Задание:

Создайте программу, которая выполняет бинарный поиск для определения количества элементов в отсортированном векторе, которые не превышают заданного значения.

Формат входных данных

В первой строке задано число n ($1 \leq n \leq 10^5$) - количество элементов вектора. В следующей строке через пробел записаны n целых чисел, которые составляют вектор (все числа отсортированы по возрастанию). В третьей строке задано число q ($1 \leq q \leq 10^5$) - количество запросов. В следующих q строках задаются числа, для которых нужно определить количество элементов в векторе, которые не превышают данное число.

Формат выходных данных

Для каждого запроса в отдельной строке выведите количество элементов в векторе, которые не превышают заданное число.

Sample Input:

```
5
1 3 5 7 9
3
3
4
9
```

Sample Output:

```
2
2
5
```

Запрос:

Дано условие задачи и решение ученика.

1. Указать на ошибки код-стайла
2. Дать комментарии по тому, как улучшить код
3. Проверить код на правильность решения поставленной задачи

Смотрим на ответ ChatGPT:



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 запятой.

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. **Именованые переменные**: 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.

Был выдан верный ответ, который можно немного отредактировать и отправлять студенту задание на доработку.

4 кейс. Помощь в создании курсов

Выше было описано, как можно создавать задания и тесты.

Создание конспектов

Подобным образом могут быть составлены конспекты занятий в требуемом формате.

Стоит описать:

1. Тему
2. То, что необходимо включить в конспект
3. Дополнительные критерии

Запрос:

Составить конспект

1. Тема: Списки (list) в Python
2. Включить в конспект: создание списков, доступ к элементам списка, изменение списка, методы списков, генераторы списков
3. Приводи примеры кода на Python

Смотрим на ответ ChatGPT:



You

Составить конспект

1. Тема: Списки (list) в Python
2. Включить в конспект: создание списков, доступ к элементам списка, изменение списка, методы списков, генераторы списков
3. Приводи примеры кода на Python



ChatGPT

Конспект: Списки (list) в Python

Определение

Список в Python - это упорядоченная коллекция объектов, которые могут быть различных типов. Списки являются одними из наиболее часто используемых структур данных в Python благодаря их гибкости и простоте использования.

Создание списков

Списки можно создавать несколькими способами:

1. С помощью литералов списка:

```
python
```

Copy code

```
my_list = [1, 2, 3, "Python", 5.0]
```

2. С помощью конструктора `list()`:

```
python
```

Copy code

```
my_list = list([1, 2, 3, "Python", 5.0])
```

3. Создание пустого списка:

```
python
```

Copy code

```
my_list = []
```

Доступ к элементам списка

Элементы списка доступны по их индексу. Индексация начинается с 0.

- **Доступ к элементу:**

```
python Copy code  
  
my_list = [1, 2, 3, "Python", 5.0]  
print(my_list[3]) # Выведет 'Python'
```

- **Отрицательная индексация (доступ с конца списка):**

```
python Copy code  
  
print(my_list[-1]) # Выведет 5.0
```

- **Срезы (доступ к подписку):**

```
python Copy code  
  
print(my_list[1:3]) # Выведет [2, 3]
```

Изменение списка

Списки в Python - изменяемые объекты. Это означает, что их содержимое может быть изменено.

- **Изменение элемента:**

```
python Copy code  
  
my_list[3] = "Java"
```

- **Добавление элементов:**

- В конец списка: `my_list.append('new item')`
- На определенную позицию: `my_list.insert(1, 'new item')`

- **Удаление элементов:**

- Удаление по значению: `my_list.remove('Python')`
- Удаление по индексу: `del my_list[0]` или `my_list.pop(0)`

Методы списков

Списки в Python имеют множество полезных методов, вот некоторые из них:

- `append(x)`: Добавляет элемент в конец списка.
- `extend(iterable)`: Расширяет список, добавляя в конец все элементы из указанного итерируемого объекта.
- `insert(i, x)`: Вставляет элемент на указанную позицию.
- `remove(x)`: Удаляет первый элемент со значением x.
- `pop([i])`: Удаляет элемент по индексу и возвращает его.
- `clear()`: Очищает список.
- `index(x[, start[, end]])`: Возвращает индекс первого элемента со значением x.
- `count(x)`: Возвращает количество элементов со значением x.
- `sort(key=None, reverse=False)`: Сортирует элементы списка.
- `reverse()`: Разворачивает элементы списка.

Генераторы списков (List Comprehensions)

Генераторы списков предоставляют сжатый способ создания списков. Общий синтаксис:

```
python Copy code  
[expression for item in iterable if condition]
```

- **Пример:** Создание списка квадратов чисел от 1 до 5.

```
python Copy code  
squares = [x**2 for x in range(1, 6)]  
print(squares) # Выведет [1, 4, 9, 16, 25]
```

Конспект можно дополнять и расширять, а также попросить использовать требуемый стиль (html, markdown и другие разметки и т.д.). После проверки можно выкладывать в открытый доступ (например, Stepik).

Первое, с чем сталкивается учащийся, который хочет пройти курс на какой-либо платформе – его главная страница. По той информации, которая размещена на ней, учащийся будет принимать решение о прохождении курса. Написать такой текст – задача непростая, но, к счастью, ее можно делегировать искусственному интеллекту.

Генерируем «**Краткое описание**» и «**Чему вы научитесь**»

Запрос «Краткое описание»

Действуй как маркетолог онлайн-курса. Напиши Elevator Speech про следующий курс, используй 500 знаков.

Курс: Базовый курс по Python

Вы научитесь

- Разбираться в работе с переменными, типами данных, управляющими структурами, функциями
- Решать практические задачи и находить эффективные решения

Смотрим на ответ ChatGPT:



You

Действуй как маркетолог онлайн-курса. Напиши Elevator Speech про следующий курс, используй 500 знаков.

Курс: Базовый курс по Python

Вы научитесь

- Разбираться в работе с переменными, типами данных, управляющими структурами, функциями
- Решать практические задачи и находить эффективные решения



ChatGPT

"Хотите освоить Python и открыть для себя мир программирования? Наш базовый курс по Python - это ваш идеальный старт! Здесь вы познакомитесь с основами: переменными, типами данных, управляющими структурами и функциями. Наш курс не просто теория; он наполнен практическими задачами, которые научат вас мыслить как программист и находить эффективные решения. Подходит для начинающих, не требует предварительных знаний. Присоединяйтесь к нам и сделайте первый шаг в мир программирования с Python!"

Запрос «Чему вы научитесь»

Смотрим на ответ ChatGPT:



You

Об этом курсе составить описание "Чему вы научитесь"



ChatGPT

Чему Вы Научитесь в Базовом Курсе по Python

- Основы Python:** Понимание и использование основных концепций языка программирования Python, включая синтаксис и основные операции.
- Работа с Переменными:** Научитесь создавать и использовать переменные, понимать их типы и особенности в Python.
- Типы Данных:** Освойте различные типы данных в Python, такие как числа, строки, списки, кортежи, словари и множества, и узнаете, как и когда их использовать.
- Управляющие Структуры:** Изучите, как использовать условные операторы и циклы для контроля потока выполнения программы.
- Функции:** Научитесь определять и использовать функции для структурирования и упрощения кода, понимания концепций области видимости и передачи аргументов.
- Решение Практических Задач:** Получите практический опыт решения задач, что поможет закрепить теоретические знания и развить навыки программирования.
- Эффективные Решения:** Научитесь находить и применять эффективные подходы к решению задач, что является ключевым навыком в программировании.
- Основы Отладки:** Освойте базовые приемы отладки кода, чтобы быстро находить и исправлять ошибки.
- Введение в Продвинутые Темы:** Получите представление о более сложных аспектах Python, подготовив почву для дальнейшего обучения.

Этот курс даст вам твердую основу в программировании на Python, подготовив к изучению более сложных тем и технологий.

Вы можете попросить забавный или грустный текст, продающий или более строгий, ограничить количество символов, сгенерировать несколько вариантов. Попросить какую-то часть переписать и т.д.

5 кейс. Помощь в наполнении программы учебной дисциплины

Создание программы учебной дисциплины начинается с шаблона. За основу был взят ПУД (программа учебной дисциплины) из Приложения 1.

Начнем писать запросы в ChatGPT. Для начала **опишем сам курс** по которому будем составлять ПУД. Можно также использовать нейросеть (кейс 4).

Курс "Python для начинающих" предназначен для тех, кто хочет освоить основы программирования с нуля. В рамках курса участники познакомятся с базовыми концепциями Python: типы данных, операторы, условные конструкции и циклы, функции и модули. Отдельное внимание уделяется работе со строками, списками, словарями и объектно-ориентированному программированию. Курс включает теоретические материалы и практические задания, которые помогут закрепить полученные знания и научиться создавать простые программы на Python.

Цели курса:

1. Учащиеся научатся работать с базовыми элементами Python, включая переменные, типы данных, функции и классы.
2. Курс направлен на формирование у студентов способности к аналитическому и логическому мышлению через решение различных задач на программирование.
3. Учащиеся будут иметь возможность самостоятельно создавать небольшие программы на Python, используя приобретенные навыки.

Почему стоит выбрать этот курс:

1. **Актуальность:** Python является одним из самых популярных и востребованных языков программирования, используемых во многих отраслях, включая веб-разработку, науку о данных, искусственный интеллект и другие.
2. **Подход "с нуля":** Курс подходит даже для тех, кто никогда ранее не сталкивался с программированием.
3. **Практическая направленность:** Большой упор делается на практические задания, что позволяет быстро применять новые знания на практике.

Особенности курса:

1. **Гибкость:** Курс подойдет как тем, кто предпочитает самостоятельное обучение, так и тем, кто выбирает обучение с преподавателем.
2. **Индивидуальный подход:** В зависимости от предпочтений и целей учащегося, можно выбрать определенные модули или проходить весь курс целиком.
3. **Современные методики обучения:** Курс включает интерактивные уроки, тесты для самопроверки, а также автопроверяемые задания на написание кода, направленные на закрепление полученных знаний.

Начальные требования

1. **Базовые навыки работы с компьютером:** Умение пользоваться основными функциями ОС, файловыми системами.

2. **Математическая грамотность:** Понимание основ математики на уровне средней школы будет плюсом.
3. **Английский язык:** Умение читать техническую документацию на английском языке будет весьма полезным, но не обязательным.

Запрос 1. Генерация общего описания по шаблону.

На основе информации о курсе сгенерируй по шаблону аннотацию для общего описания курса.



ChatGPT

Курс "Python для начинающих" - это ваш путь в мир программирования, где вы с нуля освоите основы одного из самых востребованных языков. Этот курс идеально подходит для тех, кто хочет начать свой путь в программировании, а также для тех, кто стремится расширить свои технические навыки. Вы познакомитесь с ключевыми концепциями Python, включая типы данных, операторы, условные конструкции, циклы, функции и модули. Особое внимание уделяется работе со строками, списками, словарями и основам объектно-ориентированного программирования.

Запрос 2. Генерация целей, результатов освоения дисциплины.

Сгенерировать цель курса, а также планируемые результаты обучения (не менее 10 штук по таксономии Блума).

Сгенерировать цель курса, а также планируемые результаты обучения (не менее 10 штук по таксономии Блума).



ChatGPT

Цель Курса

Целью курса "Python для начинающих" является предоставление учащимся твердого фундамента в основах программирования на Python, включая понимание ключевых концепций и развитие навыков, необходимых для создания простых программ. Курс направлен на развитие аналитического мышления и практических навыков программирования, что позволит учащимся успешно применять Python в реальных проектах.

Планируемые Результаты Обучения (по таксономии Блума)

1. **Знание:**
 - Определение и понимание основных терминов и концепций в Python, таких как переменные, циклы, условные операторы и функции.
2. **Понимание:**
 - Интерпретация и объяснение работы различных типов данных и структур данных в Python.
3. **Применение:**
 - Разработка простых программ на Python, используя базовые конструкции языка.
 - Использование условных операторов и циклов для решения задач.
4. **Анализ:**
 - Разбор и анализ кода на Python для выявления логических ошибок и оптимизации.
 - Сравнение различных подходов к решению задач на Python.
5. **Синтез:**
 - Создание собственных функций и модулей для решения конкретных задач.
 - Интеграция различных концепций Python для создания комплексных программ.
6. **Оценка:**
 - Критический анализ и оценка эффективности различных программных решений.
 - Оценка собственного кода и кода других с точки зрения читаемости, эффективности и соответствия ●

Запрос 3. Содержание учебной дисциплины.

По темам курса составить содержание учебной дисциплины. Где для каждой темы предусмотреть описание планируемых результатов обучения (связать с ранее сгенерированными), а также форм контроля. Описать по пунктам.

Для справки. Курс будет состоять из тем:

1. Введение. Основы, print(), input() и типы данных
2. Операторы (унарные, бинарные), условия.
3. Циклы
4. Коллекции (список)
5. Коллекции (набор, кортеж, словарь)
6. Функции
7. Файлы
8. Регулярные выражения
9. Простой TelegramBot (pytelegrambotapi или aiogram)
10. Анализ данных (панды)
11. Построение графиков (matplotlib, pandas)
12. Веб-скрапинг

По каждой теме предусмотрена домашняя работа. В конце студентов ожидает экзамен.

Описание домашнего задания: набор автоматически проверяемых задач на написание кода.

Критерии оценивания: Код решает задачу, проходит тесты, работает при возможных ограничениях памяти и времени.

Экзамен: аналогично домашним работам. Проводится в сессионное время и на него отводится 120 минут.

Формула оценки итоговой: $0.3 * \text{Экзамен} + 0.7 * \text{Домашние_задания}$ (среднее значение).



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.

Запрос 4. Оценивание

Учитывая формулу: $0.3 * \text{Экзамен} + 0.7 * \text{Домашние_задания}$ (среднее значение).

По каждой теме предусмотрена домашняя работа. В конце студентов ожидает экзамен.

Описание домашнего задания: набор автоматически проверяемых задач на написание кода.

Критерии оценивания: Код решает задачу, проходит тесты, работает при возможных ограничениях памяти и времени.

Экзамен: аналогично домашним работам. Проводится в сессионное время и на него отводится 120 минут.

По шаблону сгенерировать

Формат оценивания

Промежуточная аттестация (2023/2024 учебный год 1 модуль)

Формула оценивания: (ПРИМЕР) $\text{Контрольная работа 1} * 0.3 + \text{Контрольная работа 2} * 0.3 + \text{Экзамен} * 0.4$

(ПРИМЕР ОДНОГО ИЗ ЭЛЕМЕНТОВ КОНТРОЛЯ)

- Контрольная работа 1 (0.3)

Проводится: 2 модуль 2023/2024, Учебный период, Оффлайн

Оценивается: 2 модуль 2023/2024

Пересдача: Возможна или Невозможна

Правила пересдачи: Пересдача возможна при пропуске по уважительной причине

Критерии оценивания:

Для каждого задания в работе указано, сколько баллов получает студент при его полностью корректном выполнении. Итоговая оценка за работу вычисляется как сумма набранных баллов или по правилам, прописанным в тексте работы, при их наличии. За задания могут выставляться частичные баллы в соответствии с долей выполненного задания, если критерии сформулированы в тексте задания.

Примеры заданий:

Построить ER-модель для проектирования БД кафедры, содержащую информацию, необходимую для закрепления тем курсового проектирования. Исходными данными является:

- Выписка из учебного плана с указанием, какие КП (по каким предметам) на каких потоках и в каком семестре должны быть выполнены в данном учебном году.
- Какие преподаватели могут руководить КП по данному предмету.
- Плановое число КП по каждой дисциплине на каждом потоке для каждого преподавателя.
- Численность студентов в каждой студенческой группе, в том числе платных и бесплатных.
- Темы КП по каждой дисциплине.

Проверяет планируемые результаты обучения:

- Владеть навыками проектирования структуры БД и ХД
- Владеть навыками формирования SQL-запросов к БД и MDX-запросов к ХД
- Знать архитектуру программных средств, работающих с БД и ХД
- Уметь строить запросы, используя различные языковые средства

* к Экзамену можно приложить примерные вопросы/билеты



ChatGPT

Формат Оценивания

Промежуточная Аттестация (2023/2024 Учебный Год 1 Модуль)

Формула оценивания: Домашние задания (среднее значение) * 0.7 + Экзамен * 0.3

Домашние Задания (0.7)

- **Проводится:** В течение всего модуля, Онлайн
- **Оценивается:** После каждой темы курса
- **Пересдача:** Невозможна
- **Критерии оценивания:**
 - Каждое домашнее задание оценивается на основе автоматически проверяемых задач на написание кода.
 - Код должен решать задачу, проходить тесты и работать в рамках ограничений памяти и времени.
 - Оценка за каждое задание вычисляется как сумма набранных баллов.
- **Примеры заданий:**
 - Создать программу на Python, которая считывает данные из файла и выводит статистику по данным.
 - Написать функцию, которая реализует алгоритм сортировки и протестировать её на различных входных данных.
- **Проверяет планируемые результаты обучения:** 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

Экзамен (0.3)

- **Проводится:** В конце 1 модуля 2023/2024, Сессионное время, Оффлайн
- **Оценивается:** В конце 1 модуля 2023/2024
- **Пересдача:** Возможна
- **Правила пересдачи:** Пересдача возможна при неудовлетворительной оценке или пропуске по уважительной причине
- **Критерии оценивания:**
 - Экзамен состоит из автоматически проверяемых задач на написание кода, аналогично домашним заданиям.
 - Оценка за экзамен вычисляется как сумма набранных баллов за каждую задачу ●

Подобным образом с использованием уточняющих запросов можно сгенерировать текст программы учебной дисциплины. Стоит внимательно вычитывать содержание.

Приложение 1. Шаблон ПУДа

Общая информация по курсу

Наименование дисциплины	
Учебный год	2023/2024
Язык обучения	Обучение ведется на русском языке
Кредиты	
Кто читает:	
Статус:	
Когда читается:	
Формат изучения	без онлайн-курса
Охват аудитории	для своего кампуса
Контактные часы	
Преподаватели	
Аннотация	

Программа учебной дисциплины «»

Утверждена

Академическим советом ОП Протокол № от . .20

Разработчик	
Число кредитов	
Контактная работа (час.)	
Самостоятельная работа (час.)	
Курс, Образовательная программа	
Формат изучения дисциплины	Без использования онлайн курса

1. Цель, результаты освоения дисциплины и пререквизиты

Цель освоения: **(Пример)** *Формирование знаний, умений и навыков проектирования баз данных (БД), хранилищ данных (ХД), администрирования баз и хранилищ данных, разработки приложений (внешних интерфейсов) БД и ХД, использования средств многомерного анализа данных класса OLAP*

Планируемые результаты:

- **(Пример)** Владеть навыками проектирования структуры БД и ХД
- **(Пример)** Уметь строить запросы, используя различные языковые средства

2. Содержание учебной дисциплины

Тема (раздел дисциплины)	Объем в часах	Планируемые результаты обучения (ПРО), подлежащие контролю	Формы контроля
	ЛК		
	СМ		
	онл/ср		
(ПРИМЕР) Основы теории реляционных баз данных	2	(ПРИМЕР) Отличительные особенности реляционных моделей. Основные понятия: отношение, кортеж, атрибут, ключ (простой, составной; первичный, альтернативный). Домен. Связывание таблиц. Внешний ключ. Функциональные зависимости. Многозначные зависимости. Операции реляционной алгебры. Трёхзначная логика (3VL). Теория нормализации. Понятие 1НФ, 2НФ, 3НФ, 4НФ, 5НФ. Алгоритм нормализации. Достоинства и недостатки подхода.	
	2		
	14		
	4		

(ПРИМЕР) Проектирование баз данных – логическое и физическое моделирование	4	(ПРИМЕР) Общие сведения о даталогическом моделировании. Особенности даталогических моделей. Влияние особенностей модели на проектные решения. Факторы, влияющие на проектирование БД. Критерии оценки проекта БД. Проектирование структуры базы на основе использования ER-моделей. Влияние нотации ER-модели на сферу ее использования и алгоритм проектирования. Алгоритм проектирования (алгоритм перехода от базовой ER-модели к структуре реляционной базы). ERwin Data Modeler. Уточнение терминологии. Интерфейс системы. Создание логических и физических моделей. Методология проектирования в среде ERwin Data Modeler. Методология построения физических моделей. Типы данных. Физическое хранение данных (файлы, файловые группы, страницы, экстенды, индексы, секции). Влияние нормализации/денормализации.	
	18		

Часов по видам учебных занятий:			
Итого часов:			

3. Оценивание

Промежуточная аттестация (2023/2024 учебный год 1 модуль)

Формула оценивания: *(ПРИМЕР)* $\text{Контрольная работа 1} * 0.3 + \text{Контрольная работа 2} * 0.3 + \text{Экзамен} * 0.4$

(ПРИМЕР ОДНОГО ИЗ ЭЛЕМЕНТОВ КОНТРОЛЯ)

- Контрольная работа 1 (0.3)

Проводится: 2 модуль 2023/2024, Учебный период, Оффлайн

Оценивается: 2 модуль 2023/2024

Пересдача: Возможна или Невозможна

Правила пересдачи: Пересдача возможна при пропуске по уважительной причине

Критерии оценивания:

Для каждого задания в работе указано, сколько баллов получает студент при его полностью корректном выполнении. Итоговая оценка за работу вычисляется как сумма набранных баллов или по правилам, прописанным в тексте работы, при их наличии. За задания могут выставляться частичные баллы в соответствие с долей выполненного задания, если критерии сформулированы в тексте задания.

Примеры заданий:

Построить ER-модель для проектирования БД кафедры, содержащую информацию, необходимую для закрепления тем курсового проектирования. Исходными данными является: • Выписка из учебного плана с указанием, какие КП (по каким предметам) на каких потоках и в каком семестре должны быть выполнены в данном учебном году. • Какие преподаватели могут руководить КП по данному предмету. • Плановое число КП по каждой дисциплине на каждом потоке для каждого преподавателя. • Численность студентов в каждой студенческой группе, в том числе платных и бесплатных. • Темы КП по каждой дисциплине.

Проверяет планируемые результаты обучения:

- Владеть навыками проектирования структуры БД и ХД
- Владеть навыками формирования SQL-запросов к БД и MDX-запросов к ХД
- Знать архитектуру программных средств, работающих с БД и ХД
- Уметь строить запросы, используя различные языковые средства

* к Экзамену можно приложить примерные вопросы/билеты

4. Ресурсы

4.1. Рекомендуемая основная литература

№ п/п	Наименование
1	(ПРИМЕР) Adam Aspin. (2020). Pro Power BI Desktop : Self-Service Analytics and Data Visualization for the Power User: Vol. Third edition. Apress. http://search.ebscohost.com/login.aspx?direct=true&site=eds-live&db=edsebk&AN=2498686
2	
3	

4.2. Рекомендуемая дополнительная литература

№ п/п	Наименование
1	(ПРИМЕР) Гордеев С. И., Волошина В. Н. - ОРГАНИЗАЦИЯ БАЗ ДАННЫХ В 2 Ч. ЧАСТЬ 1 2-е изд., испр. и доп. Учебник для вузов - М.:Издательство Юрайт - 2021 - 310с. - ISBN: 978-5-534-04469-0 - Текст электронный // ЭБС ЮРАЙТ - URL: https://urait.ru/book/organizaciya-baz-dannyh-v-2-ch-chast-1-471758
2	
3	

4.3. Программное обеспечение

№ п/п	Наименование	Условия доступа/скачивания
1	(ПРИМЕР) R	(ПРИМЕР) Язык программирования для статистических вычислений и графики, Разработка ПО, Свободное лицензионное соглашение
2	(ПРИМЕР) RStudio	(ПРИМЕР) Среда разработки (IDE) для R, Разработка ПО, Свободное лицензионное соглашение
3		

4.4. Профессиональные базы данных, информационные справочные системы, интернет-ресурсы (электронные образовательные ресурсы)

№п/п	Наименование	Условия доступа/скачивания
1	(ПРИМЕР) Электронно-	(ПРИМЕР) https://urait.ru/

	библиотечная система Юрайт	Образовательная платформа Юрайт — образовательный ресурс, электронная библиотека и интернет-магазин, где читают и покупают электронные и печатные учебники авторов — преподавателей ведущих университетов для всех уровней профессионального образования, а также пользуются видео- и аудиоматериалами, тестированием и сервисами для преподавателей, доступными 24 часа 7 дней в неделю.
2		
3		