| Домашнее задание |
| --- |

**Результат**

ССЫЛКА на github с выполненным заданием

<https://github.com/makarova1507ana/python312/tree/OOP/leeson23> - последняя практика

рекомендуется использование декораторов

**ЗАДАНИЕ**

**Выбрать минимум 1 любую задачу**

**задача**

Определите класс Task с атрибутами title и status (например, "в процессе" или "завершено") и методом display\_info(), который выводит информацию о задаче.

Создайте дочерний класс AssignedTask, который наследует от класса Task. Добавьте атрибут assignee (назначенный сотрудник) и метод assign\_task(employee), который принимает объект сотрудника и назначает ему задачу.

Пример использования:

# Создание объектов

task1 = Task("Complete report", "в процессе")

task2 = AssignedTask("Code review", "ожидает", None)

# Вызов методов

task1.display\_info() # Выводит информацию о задаче

employee1 = User("Alice")

task2.assign\_task(employee1) # Назначение задачи сотруднику

task2.display\_info() # Выводит информацию о задаче, включая назначенного сотрудника

**задача**

Определите класс Book с атрибутами title, author и status (например, "в наличии" или "выдана") и методом display\_info(), который выводит информацию о книге.

Создайте дочерний класс BorrowableBook, который наследует от класса Book. Добавьте атрибут borrower (заимствовавший книгу) и методы borrow\_book(user) и return\_book(), которые позволяют пользователю взять книгу в библиотеке и вернуть ее после использования.

пример

# Создание объектов

book1 = Book("The Great Gatsby", "F. Scott Fitzgerald", "в наличии")

book2 = BorrowableBook("To Kill a Mockingbird", "Harper Lee", "в наличии", None)

# Вызов методов

book1.display\_info() # Выводит информацию о книге

user2 = User("Bob")

book2.borrow\_book(user2) # Заимствование книги пользователем

book2.display\_info() # Выводит информацию о книге, включая заимствовавшего пользователя

book2.return\_book() # Возвращение книги в библиотеку

book2.display\_info() # Выводит информацию о книге после возврата

**задача**

Определите класс Animal с атрибутами name (имя животного) и sound (звук, который издает животное) и методом make\_sound(), который выводит сообщение о звуке, издаваемом животным.

Создайте дочерний класс Bird, который наследует от класса Animal. Добавьте атрибут flight\_altitude (высота полета) и метод fly(), который выводит сообщение о полете птицы.

Пример использования:

# Создание объектов

lion = Animal("Lion", "Roar")

parrot = Bird("Parrot", "Squawk", 100)

# Вызов методов

lion.make\_sound() # Выводит сообщение о звуке льва

parrot.make\_sound() # Выводит сообщение о звуке попугая

parrot.fly() # Выводит сообщение о полете птицы

**задача**

Определите класс IDGenerator, который генерирует уникальные идентификаторы. Добавьте статический метод generate\_id(), который возвращает строку с уникальным идентификатором. Идентификатор может, например, состоять из текущей даты и случайного числа.

пример

# Использование IDGenerator

id1 = IDGenerator.generate\_id()

id2 = IDGenerator.generate\_id()

print(f"Generated ID 1: {id1}")

print(f"Generated ID 2: {id2}")

**задача**

Определите класс StringUtils, который содержит classmethod count\_characters(cls, input\_string). Метод должен принимать строку и возвращать словарь, в котором ключами являются символы, а значениями — количество их вхождений в строку.

Пример использования:

# Использование StringUtils

char\_count = StringUtils.count\_characters("hello world")

print(f"Character Count: {char\_count}")