ZADANIE: System zarządzania modelami ai

📌 *Opis:*

Napisz klasę `ModelAI`, która pozwala na śledzenie liczby utworzonych modeli oraz umożliwia tworzenie obiektu na podstawie pliku json.

📌 *Wymagania:*

☑ **Atrybuty instancji:**

- `nazwa_modelu` (nazwa modelu AI)
- `wersja` (wersja modelu)

☑ **Atrybut klasowy:**

- `liczba_modeli` (przechowuje globalną liczbę utworzonych modeli)

☑ **Metody:**

- `nowy_model()`: Zwiększa licznik utworzonych modeli.
- `ile_modeli(cls)`: Zwraca liczbę utworzonych modeli.
- `z_pliku(cls, nazwa_pliku)`: Tworzy obiekt na podstawie pliku `.json`.

📌 Przykładowy plik `model.json`:

```
{
    “name”: “face_reco”,
    “version”: 2.0
}
```

Zadanie 2

Napisz klasę `Matrix`, która reprezentuje macierz 2x2. Przeciąż operatory:

- + → dla dodawania macierzy.
- * → dla mnożenia macierzy.

Wymagania:

- Konstruktor przyjmujący elementy macierzy `a, b, c, d`.
- Metody `__add__()` i `__mul__()` zwracające nową macierz.
- Metoda `__str__()` do wyświetlania macierzy.
- Metoda `__repr__()` do wyświetlania (reprezentacji) macierzy.

Przykład użycia:

>> m1 = Matrix(1, 2, 3, 4)

>> m2 = Matrix(2, 0, 1, 2)

>> m3 = m1 + m2

>> print(m3)

[3, 2;

4, 6]

m4 = m1 * m2

>> print(m4) # Wynik mnożenia macierzy

[4, 4;

10, 8]

>>print(repr(m4))

Matrix(4, 4, 10, 8)

Zadanie 3: Dziedziczenie – Hierarchia pracowników

Stwórz system zarządzania pracownikami w firmie.

1. Zaimplementuj klasę `Osoba`, która ma atrybuty:
    a. `imie` (str)
    b. `nazwisko` (str)
    c. `wiek` (int)
    d. metodę `przedstaw_sie()`, zwracającą `"Jestem {imie} {nazwisko}."`
2. Stwórz klasę `Pracownik`, dziedziczącą po `Osoba`, która dodatkowo ma:
    a. `stanowisko` (str)
    b. `pensja` (float)
    c. metodę `info_o_pracy()`, zwracającą `"Pracuję jako {stanowisko}, zarabiam {pensja} zł."`
3. Dodaj klasę `Manager`, dziedziczącą po `Pracownik`, z nowym atrybutem `zespol` (lista pracowników).
    a. Rozszerz metodę `przedstaw_sie()`, aby zawierała informację o liczbie podwładnych.
    b. Dodaj metodę `dodaj_do_zespolu(pracownik)`, która dodaje pracownika do zespołu.

Test: Stwórz instancje `Managera` oraz kilku `Pracowników`, dodaj ich do zespołu i wywołaj metody.

Zadanie 4: System biblioteczny

Stwórz hierarchię klas, która modeluje różne rodzaje książek w bibliotece.

Wymagania:

1. Utwórz klasę bazową `Ksiazka` z atrybutami:
    a. `tytul` (str)
    b. `autor` (str)
    c. `rok_wydania` (int)
    d. Metoda `opis()`, która zwraca podstawowy opis książki.
2. Utwórz klasy dziedziczące:
    a. `Ebook` – dodatkowy atrybut `rozmiar_pliku` (w MB). Metoda `opis()` powinna dodatkowo informować o rozmiarze pliku.
    b. `Audiobook` – dodatkowy atrybut `czas_trwania` (w minutach). Metoda `opis()` powinna zwracać informację o czasie trwania.
3. **Test:** Utwórz kilka obiektów obu typów i wypisz ich opisy.

Zadanie 5: Inteligentny telefon – wielodziedziczenie czy kompozycja?

Zaprojektuj system dla inteligentnego telefonu, który posiada funkcje komunikacji i rozrywki. Zadanie wymaga przemyślenia, czy lepiej zastosować wielodziedziczenie czy kompozycję.

Scenariusz:

- Każdy inteligentny telefon ma podstawowe cechy, takie jak `model` i `producent`.
- Funkcjonalność komunikacji obejmuje wysyłanie wiadomości.
- Funkcjonalność rozrywki obejmuje odtwarzanie muzyki.
- Istnieje telefon, który ma obie te funkcje.

Wymagania:

1. Utwórz klasę `Telefon` z atrybutami `model` i `producent`.
2. Utwórz osobne klasy:
    a. `Komunikacja` z metodą `wyslij_wiadomosc(odbiorca, tresc)`
    b. `Rozrywka` z metodą `odtworz_muzyke(utwor)`
3. Zastanów się, czy stworzyć klasę `Smartphone` dziedziczącą jednocześnie po `Telefon`, `Komunikacja` i `Rozrywka` (wielodziedziczenie) lub zastosować kompozycję, czyli w klasie `Smartphone` umieścić atrybuty będące obiektami klas `Komunikacja` i `Rozrywka`.
4. **Test:**
    a. Jeśli zdecydujesz się na wielodziedziczenie – zaimplementuj klasę `Smartphone` i przetestuj jej funkcjonalności.
    b. Jeśli wybierzesz kompozycję – stwórz obiekty funkcjonalności i przypisz je do telefonu, a następnie wywołaj odpowiednie metody.

Pytanie do przemyślenia:
W jakich sytuacjach wielodziedziczenie może prowadzić do problemów (np. konflikty metod), a kiedy lepiej zastosować kompozycję?