12 grudnia 2025

Kurs rozszerzony języka Python

Lista 10.

Poniższe zadania proszę zaprogramować używając relacyjnej bazy danych (np. SQLite). Rozwiązania powinne spełniać warunki:

- program powinien zawierać przynajmniej trzy klasy (tabele), bez tabel dodatkowych służących do budowy związków typu *many–to–many*;
- tworzenie i dostęp do danych ma odbywać się za pomocą warstwy ORM, np. wykorzystując omówiony na wykładzie `SQLAlchemy`[1];
- związki między danymi (np. one–to–many) powinne być jawnie opisane w deklaracjach danych;
- definicje tabel powinne zawierać przynajmniej jedną walidację;
- przygotuj testowe dane (np. w formacie json), które będą automatycznie wciągane przy inicjowaniu nowej bazy danych;
- interfejsem użytkownika jest wywołanie programu z odpowiednimi argumentami, np.

  ```
  python3 mojprogram.py spotkania --dodaj --opis=Wyklady
  python3 mojprogram.py spotkania --wypisz
  ```

  Przyjmij, że dla każdego modelu jest możliwość co najmniej dodawania, aktualizacji i wyszukiwania.

  Do analizy argumentów wywołania programu wykorzystaj gotowe pakiety,[2] np. `argparse` czy `getopt`. Zadbaj też o informację o argumentach wywołania programu, np. po podaniu argumentu `--help`.

Zadanie 1.

Napisz program, który przechowujący informacje o posiadanych książkach (przynajmniej autor, tytuł, rok) oraz listę znajomych (przynajmniej imię, email), którzy od nas wypożyczają książki. Zaprogramuj także operacje dodawania nowych książek, wypożyczania książek i oddawania.

Zadanie 2.

Zaprogramuj system przechowujący informacje o filmach (mogą to też być inne utwory, np. muzyczne). Dane do przechowywania to opis samego filmu (tytuł, rok powstania) oraz osoby zaangażowane w jego powstanie (reżyser, operator, producent).

Zadanie 3.

Zaprogramuj własny kalendarz przechowujący wydarzenia (godz. rozpoczęcia i zakończenia, opis), np.zajęcia, spotkania etc, wraz z listą osób (imię, email), które są przypisane do wydarzenia. Przy dodawaniu wydarzeń informuj, czy wydarzenie nie koliduje z już zapamiętanym wydarzeniem.

Na zajęcia należy wykonać jedno z tych zadań. Każde zadanie jest warte 5 punktów. To zadanie będzie rozwijane na kolejnych listach zadań.

Marcin Młotkowski

[1] inne ORM'y: np. `peewee`, `Django ORM` czy `Pony ORM`

[2] wykorzystanie narzędzi do parsowana argumentów jest częścią zadania, za ręczną obsługę może zostać obniżona punktacja