Przykładowe kolokwium #2- Zestaw #1

Zadania mogą być wykonane jako skrypt lub notatnik jupytera. Rozwiązania należy umieścić:

- opcja 1: (jako kody) w prywatnym repozytorium na Githubie
- opcja 2: spakować jako zip i umieścić na pendrive.

Zadanie A.

Napisz klasę `Game`. Klasa powinna posiadać prywatne atrybuty instancyjne:

1. `name` typu `str` (pol. nazwa gry)
2. `genre` typu `str` (pol. gatunek)
3. `pegi` typu `int` (pol. klasyfikacja wiekowa – minimalny wiek)
4. `price` typu `float` (pol. cena)
5. `reputation` typu `float` (pol. średnia recenzja gier)

Zaimplementuj inicjalizator z pięcioma argumentami (zachowaj kolejność i nazwy jak wcześniej). Zadbaj również o to aby inicjalizator w razie podania `pegi` spoza przedziału [0,18], ustawiał wartość 18. Ponadto niezależnie należy sprawdzić, aby cena była dodatnia (w innym wypadku ustawić 50.99) oraz aby, atrybut związany recenzją może przyjmować wartości z przedziału [1,6] (w innym wypadku ustawić na 1.0).

Dodaj do klasy właściwości - (getter i setter). Jeśli dla setterów podane wartości argumentów nie spełniają założeń - wyrzuć wyjątek z odpowiednim komunikatem.

Zaimplementuj klasę `Strategy` dziedziczącą po klasie `Game`. Klasa ta powinna posiadać atrybuty instancyjne prywatne:

1. `strategy_type` (pol. rodzaj gry strategicznej), typu `str`
2. `players` typu `int` (pl. liczba graczy).

Rozszerz w tej klasie inicjalizator. Niezależnie sprawdź następujące warunki. Atrybut `strategy_game` może być napisem zawierającym co najmniej 3 znaki – w przeciwnym wypadku ustaw „RTS”. Liczba graczy musi być liczbą dodatnią, w przeciwnym wypadku ustaw ją na 5.

Zaimplementuj właściwości (setter i getter) dla każdego atrybutu. Jeśli dla setterów podane wartości argumentów nie spełniają założeń, wyrzuć wyjątek z odpowiednim komunikatem.

Poza wspomianymi wyżej klasami stwórz co najmniej 3 obiekty i przechwyć ew. wyjątki.

Zadanie B:

1. Stwórz listę zawierającą 10 imion (będących napisami). Następnie za pomocą wyrażeń lambda stwórz listę zawierającą imiona rozpoczynające się od samogłoski (dot. liter z alfabetu łacińskiego).

Zadanie C:

1. Napisz funkcję, która przyjmuje listę liczb całkowitych i zwraca sumę wszystkich elementów z listy, jeśli lista nie jest pusta, w przeciwnym razie rzuć wyjątek “IndexError” z odpowiednim komunikatem. Przetestuj działanie funkcji.

Zadanie D:

1. Stwórz klasę `Book` z prywatnymi atrybutami `author` i `title`. Dodaj w klasie metodę `__init__` z dwoma argumentami ustawiającą z nich odpowiednie atrybuty. Dodaj w klasie właściwości (getter i setter).

2. Stwórz poza klasą dekorator z parametrem limit i sprawdza, czy ilość wywołań funkcji nie przekroczyła tego limitu. Jeśli tak, to zwraca wyjątek komunikatem o błędzie.
3. Uzupełnij metodę `__init__` w klasie `Book` za pomocą dekoratora tak, aby była możliwość stworzenia maksymalnie trzech obiektów.
4. Poza klasą sprawdź działanie limitu na tworzenie obiektów i przechwyć ew. wyjątki.