

Kolokwium nr 1 - Zestaw 3

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.

Dot zad. 1-3: Każdy obiekt należy stworzyć co najmniej jeden raz i każdą metodę/funkcję należy wywołać co najmniej jeden raz (o ile nie napisano inaczej). (8pkt)

1. (12pkt) Stwórz klasę **Course** a w niej następujące składniki:

- dodaj zmienne instancyjne: **destination** (**str**), **distance** (**int**), **passengers** (lista napisów), **date** (dowolny typ z modułu **datetime** umożliwiający przechowywanie daty).
- w inicjatorze nadaj wartości zmiennym z podanych odpowiednio argumentów
- upewnij się, że data przekazana jako argument w inicjatorze jest “przyszła”; w sytuacji gdy jest przeszła lub dzień bieżący - należy ustawić datę na dzień jutrzejszy od dnia bieżącego
- dodaj odpowiednią metodę magiczną zwracając napis z reprezentacją obiektu
- dodaj odpowiednią metodę magiczną umożliwiającą sortowanie obiektów typu **Course** wg klucza: najpierw sortowanie odbywa się wg daty - od najwcześniejszej do najpóźniejszej, przy równości sortowanie odbywa się wg kierunku - zgodnie z porządkiem leksykograficznym dla napisów.
- dodaj odpowiednią metodę magiczną odpowiadającą za sprawdzanie równości obiektów tak, aby obiekty były równe, gdy jedynie zmienne **destination** i **date** (w zakresie dnia) są takie same.

2. (6pkt) Stwórz klasę **Vehicle** dziedziczącą po **Course** ze składowymi:

- dodaj zmienne instancyjne **vehicleBrand** (**str**) oraz **capacity** (**int**)
- rozszerz **__init__** by ustawiał dodatkowe dwa pola z argumentów
- upewnij się, że pojemność ustawiana w inicjalizatorze jest liczbą dodatnią; jeśli argument jest niedodatni, to ustaw zmienną **capacity** jako 10.
- dodaj odpowiednią metodę magiczną zwracając napis z reprezentacją obiektu z uwzględnieniem zmiennych z klasy pochodnej
- rozszerz metodę odpowiedzialną za sortowanie tak, aby sortowanie odbywało się wg zasady dziedziczonych z klasy bazowej - ale dodatkowo przy równości sortowanie powinno się odbywać wg zmiennej **capacity** od największej do najmniejszej wartości.

3. (8pkt) Stwórz listę, w której będą po 3 obiekty typu **Course** oraz **Vehicle**. Następnie posortuj listę. Upewnij się, że “printując” listę wyświetlana jest reprezentacja obiektu pochodząca z odpowiedniej metody magicznej zdefiniowanej bezpośrednio we właściwej klasie.

4. (10pkt) Stwórz klasę **Smartphone** z następującymi składowymi:

- klasa posiada zmienną instancyjną **record** będą słownikiem, gdzie przechowywane są dane o smartfonie.
- inicjalizator powinien posiadać dwa argumenty ustawiający w polu **record** dwa elementy: jako klucz **brand** z pierwszego argumentu, jako klucz **model** z drugiego argumentu.
- dodaj do klasy **__getitem__** oraz **__setitem__** w taki sposób, aby dokonywały odpowiednich operacji na zmiennej **record** w tej klasie. Upewnij się dodatkowo, że wszystkie klucze składają się tylko z małych liter alfabetu łacińskiego.

- przesłoń metodę `__str__` tak by zwracała odpowiedni napis opisujący wszystkie informacje przechowywane w polu `record`
 - stwórz jeden obiekt i przetestuj działanie metod (każdą co najmniej jeden raz)
5. (6pkt) Stwórz klasę `Test` ze zmienną klasową `count` (`int`) z domyślną wartością zero. Zaimplementuj metodę klasową `change` z jednym argumentem `value` (`int`). Metoda ma powiększyć zmienną `count` o `value`. Przetestuj poza klasą działanie metody `change`.