KM PROGRAMS

ZARZĄDZANIE KOLEKCJĄ SAMOCHODÓW – KOMPOZYCJA

Zaimplementuj klasę **Engine**, która posiada pola składowe **type** (enum EngineType o wartościach DIESEL, GASOLINE, LPG) oraz **power** (liczba po przecinku nieujemna).

Zaimplementuj klasę **Wheel**, która posiada pola składowe **model** (napis składający się tylko i wyłącznie z dużych liter i białych znaków), **size** (liczba całkowita większa nieujemna) oraz **type** (enum TyreType o wartościach WINTER, SUMMER)

Zaimplementuj klasę **CarBody**, która posiada pola składowe **color** (enum CarBodyColor o wartościach BLACK, SILVER, WHITE, RED, BLUE, GREEN), **type** (enum CarBodyType o wartościach SEDAN, HATCHBACK, COMBI) oraz listę elementów wyposażenia nadwozia **components** (każdy napis powinien składać się tylko i wyłącznie z dużych liter oraz białych znaków).

Klasa **Car** posiada pola **model** (napis składający się tylko i wyłącznie z dużych liter oraz białych znaków), **price** (wartość nieujemna), **mileage** (liczba całkowita nieujemna), **engine** (referencja klasy Engine), **carBody** (referencja klasy CarBody) oraz **wheel** (referencja klasy Wheel). Klasa Car posiada konstruktor argumentowy, który przyjmuje jako argument nazwę pliku przechowującego dane samochodu w formacie JSON. Przykładowa postać pliku została przedstawiona poniżej.

Klasa **Cars** posiada pole składowe **cars**, które reprezentuje kolekcję elementów typu Car. Kolekcja nie posiada duplikatów. Klasa posiada konstruktor argumentowy. Argumentem konstruktora jest kolekcja napisów, która przechowuje nazwy plików JSON, z których zostaną pobrane dane dla kolejnych elementów kolekcji cars.

W klasie Cars przygotuj metody, które realizują następujące zadania:

- Metoda zwraca kolekcję samochodów posortowaną według kryterium podanego jako argument. Metoda powinna umożliwiać sortowanie według ilości komponentów, mocy silnika oraz rozmiaru opony. Dodatkowo metoda powinna umożliwiać sortowanie rosnąco oraz malejąco.
- Metoda zwraca kolekcję samochodów o określonym rodzaju nadwozia przekazanym jako argument (CarBodyType) oraz o cenie z przedziału <a, b>, gdzie a oraz b to kolejne argumenty metody.
- Metoda zwraca posortowaną alfabetycznie kolekcję modeli samochodów, które posiadają typ silnika (EngineType) przekazany jako argument metody.
- Metoda zwraca dane statystyczne dla podanej jako argument wielkości. Dopuszczalne wielkości to cena, przebieg oraz moc silnika. Dane statystyczne powinny zawierać wartość najmniejszą, wartość największą oraz wartość średnią.
- Metoda zwraca mapę, w której kluczem jest obiekt klasy Car, natomiast wartością jest liczba kilometrów, które samochód przejechał. Pary w mapie posortowane są malejąco według wartości.
- Metoda zwraca mapę, w której kluczem jest rodzaj opony (TyreType), a wartością lista samochodów o takim typie opony. Mapa posortowana jest malejąco po ilości elementów w kolekcji.
- Metoda zwraca kolekcję samochodów, które posiadają wszystkie komponenty z kolekcji przekazanej jako argument. Kolekcja posortowana jest alfabetycznie według nazwy modelu samochodu.

Błędne działanie programu należy przechwycić wyjątkiem. W wyniku wystąpienia wyjątku program nie powinien przerywać działania tylko obsłużyć sytuację wyjątkową i działać dalej.

Przykładowa postać pliku reprezentującego dane o samochodzie:

```
{
  "model": "AUDI",
  "price": 120,
  "mileage": 12000,
  "engine": {
    "type": "DIESEL",
    "power": 210.0
  },
  "carBody": {
    "color": "BLACK",
    "type": "HATCHBACK",
    "components": [
      "ABS",
      "AIR CONDITIONING"
    ]
  },
  "wheel": {
    "type": "SUMMER",
    "model": "PIRELLI",
    "size": 18
  }
}
```