Java: Klasy generyczne, kolekcje - Ćwiczenia

Ćwiczenie 1

a) Utwórz generyczną klasę `Pair` (para) do przechowywania pary wartości dowolnego, ale tego samego typu, zawierającą:
   - właściwości: `left` i `right` (prywatne pola z publicznymi getterami i setterami)
   - 2-argumentowy publiczny konstruktor ustawiający właściwości `left` i `right` na podstawie wartości podanych jako parametry wywołania
   - Publiczną metodę `swap` zamieniającą wartości `left` i `right`
   - Publiczną metodę `toString` zwracającą tekstową reprezentację obiektu w formacie „[left;right]”

b) Przetestuj działanie klasy dla par łańcuchów znaków i liczb całkowitych (utworzenie, wyświetlenie, zamiana wartości i ponowne wyświetlenie)

c) Dodaj do klasy `Pair` publiczną metodę `max`() zwracającą większą z wartości `left` i `right` (do porównania wykorzystaj metodę `compareTo`). Uwaga: Po dodaniu tej metody konieczne będzie ograniczenie możliwych typów danych, których wartości może przechowywać klasa `Pair`.

d) Przetestuj działanie metody `max` w klasie `Pair` dla par łańcuchów znaków i liczb całkowitych.

Ćwiczenie 2

a) Utwórz abstrakcyjną klasę `Animal` (zwierzę) zawierającą:
   - właściwość: `species` (gatunek) typu `String` (w formie prywatnego pola z publicznym getterem i setterem)
   - 1-argumentowy publiczny konstruktor ustawiający `species` na podstawie wartości podanej jako parametr wywołania.

b) Utwórz dwie podklasy klasy `Animal`: `Mammal` (ssak) i `Bird` (ptak).
   - Posiadające 1-argumentowy publiczny konstruktor ustawiający `species` na podstawie wartości podanej jako parametr wywołania (pamiętaj o współpracy konstruktorów podklasy i nadklasy!).

c) W głównej klasie programu utwórz statyczną metodę `printList` do wyświetlania na konsoli listy (kolekcji bibliotecznej `List`) obiektów `Mammal` lub `Bird` (listy której elementy są instancjami dowolnej, i w obrębie listy tej samej, podklasy klasy `Animal`). Szkielet metody, z kompletnym ciałem, ale z nagłówkiem do uzupełnienie przedstawiony jest poniżej:

```
public static void printList( ... list)
{
    for (Animal m : list)
    {
        System.out.println(m.getSpecies());
    }
}
```

d) Dodaj do metody `main` kolejno następujące operacje:
   - Utwórz listę (`List`) obiektów `Bird` i dodaj do niej 2 ptaki.
   - Utwórz listę (`List`) obiektów `Mammal` i dodaj do niej 2 ssaki.
   - Wyświetl zawartość obu list na konsoli, przekazując każdą z nich jako parametr metody `printList`.

e) Doczytaj co oznaczają w kontekście typów generycznych pojęcia invariance, covariance i contravariance. Dlaczego metoda z parametrem typu `List<Animal>` nie przyjmuje przy wywołaniu wartości parametrów typu `List<Bird>` i `List<Mammal>`?