

## Paradygmaty programowania - ćwiczenia

### Lista 10

Wszystkie programy mają być napisane w języku Scala.

1. Klasa `GenericCellImm` kompiluje się jako klasa inwariantna i kowariantna.

```
class GenericCellImm[T] (val x: T) {  
}  
//defined class GenericCellImm  
  
class GenericCellImm[+T] (val x: T) {  
}  
//defined class GenericCellImm
```

Natomiast klasa `GenericCellMut` kompiluje się tylko jako klasa inwariantna.

```
class GenericCellMut[T] (var x: T) {  
}  
//defined class GenericCellMut
```

Wersja kowariantna powoduje błąd kompilacji.

```
class GenericCellMut[+T] (var x: T) {  
}  
//<console>:12: error: covariant type T occurs in contravariant position in type T  
//of value x_=
```

- a) Wyjaśnij powód tego błędu.
- b) Czy można się pozbyć tego błędu? Uzasadnij swoją odpowiedź.
- c) Czy wersja kontrawariantna skompiluje się? Uzasadnij swoją odpowiedź.

```
class GenericCellMut[-T] (var x: T) {  
}
```

2. Poniższa definicja powoduje błąd kompilacji.

```
abstract class Sequence[+A] {  
  def append(x: Sequence[A]): Sequence[A]  
}  
// <console>:14: error: covariant type A occurs in contravariant position in type  
// Sequence[A] of value x
```

Wyjaśnij przyczynę tego błędu. Czy można się go pozbyć?.

3. Zdefiniuj klasę generyczną dla kowariantnej kolejki niemodyfikowalnej, reprezentowanej przez parę list (patrz lista 7, zadanie 1b).

*Wskazówka.* Wzoruj się na klasie dla stosu z wykładu 10 (str. 6 i 24).

4. Zdefiniuj generyczną inwariantną metodę copy dla kolekcji modyfikowalnych na wzór programu napisanego w Javie (wykład, str. 27).

*Wskazówka.* Wykorzystaj metodę `foreach` z cechy `scala.collection.Traversable`, oraz metodę `update` z cechy `scala.collection.mutable.Seq` (patrz Scala API).