Funkcionalno programiranje

Vežbe

05 Crte

Zadatak 1

- Postoje dva standarda za električne komponente: standard po kojem električne komponente koriste mrežni napon od 110 V i standard po kojem koriste 220 V.
- Šuko utikači i utičnice su definisane za mrežni napon od 220 V.
- Usisivač A koristi mrežni napon od 220 V i šuko utikač.
- Obezbediti da usisivač B, koji koristi mrežni napon od 110 V ne može da poseduje šuko utikač.

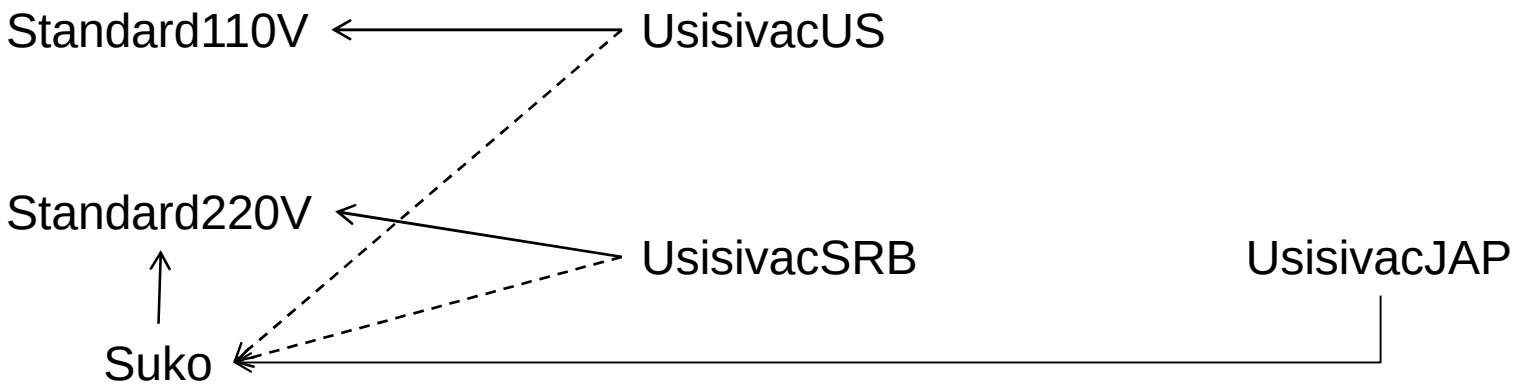
Zadatak 1

```
object elkomponente {
  class Standard110V
  class Standard220V
  trait Suko extends Standard220V

  class UsisivacSRB extends Standard220V with Suko

  class UsisivacUS extends Standard110V with Suko // greška

  class UsisivacJAP extends Suko // u redu???
}
```



Zadatak 1

- Problem – ne postoji ograničenje da uređaj mora da poštuje Standard 220 V.
- Implicitno ga poštuje, ali to može biti logička greška
  - Prevodilac ne detektuje
- Rešenje: crta mora da propiše nadklasu u koju će biti umetnuta

```
object elkomponente2 {
  class Standard110V
  class Standard220V
  trait Suko  {
    this: Standard220V =>
  }
  class UsisivacSRB extends Standard220V with Suko
  class UsisivacUS extends Standard110V with Suko // greska
  class UsisivacJAP extends Suko // greska
}
```

Zadatak 2

- Osobina pokretljivosti obezbeđuje pomeranje objekta u 2D ravni. Ova osobina zahteva da objekat može biti postavljen na proizvoljnu poziciju u 2D ravni.
- Napisati potrebne crte i klase tako da List i Kamen mogu imati osobinu pokretljivosti, ali Planina ne.

Zadatak 2

```
object movable {
  trait Position {
    protected var x: Int
    protected var y: Int
    def setPos(x: Int, y: Int) = {
      this.x = x
      this.y = y
    }
  }
  trait Movable extends Position {
    def move(x: Int, y: Int) = setPos(x,y)
  }
  class Leaf(protected override var x: Int,
             protected override var y: Int) extends Movable
  class Rock(protected override var x: Int,
             protected override var y: Int)  extends Movable
  class Mountain(protected override var x: Int,
                 protected override var y: Int)  extends Movable
  new Mountain(2,3)
  //> res0: w06.movable.Mountain = w06.movable$Mountain@f2a0b8e
}
```

Zadatak 2

```
object movable2 {

  trait Position {
    protected var x: Int
    protected var y: Int
    def setPos(x: Int, y: Int) = {
      this.x = x
      this.y = y
    }
  }

  trait Movable  {

    this: { def setPos(x: Int, y: Int): Unit } =>

    def move(x: Int, y: Int) = setPos(x,y)
  }
```

Zadatak 2

```
  class Leaf(protected override var x: Int,
             protected override var y: Int)
                                      extends Position with Movable

  class Rock(protected override var x: Int,
             protected override var y: Int)  extends Movable {
    def setPos(x: Int, y: Int) = { /*...*/ }
  }

  // greška
  class Mountain(protected override var x: Int,
                 protected override var y: Int) extends Movable
}
```