Objektorientierte Programmierung

HS-Bochum

WS 19/20

Übung 2

Aufgabe 1)
Auf diesem Übungsblatt betrachten wir den folgenden Quellcode für einen Brettspiele-Simulator:

```
public class Tuple2Dim<T extends BigDecimal> {
    public T m1 = null;
    public T m2 = null;
}
```

```
public class Entity<T extends BigDecimal, POS_TYPE extends Tuple2Dim<T>>
{
    public Boardfield<T, POS_TYPE> boardfield;
}
```

```
public abstract class Board<T extends BigDecimal, POS_TYPE extends Tuple2Dim<T>> {

    protected FieldPositionProjector<T, POS_TYPE> positionProjector;
    protected Map<String, Entity<T, POS_TYPE>> entities;
    public abstract void addEntityToBoard(Entity<T,POS_TYPE> e);}
```

```
public interface FieldPositionProjector<T extends BigDecimal, POS_TYPE extends
Tuple2Dim<T>> {

    public abstract String getPositionProjection(POS_TYPE pos);
}
```

```
public class Boardfield<T extends BigDecimal, POS_TYPE extends Tuple2Dim<T>> {

    protected POS_TYPE positionOnBoard;
    protected POS_TYPE sizeOnBoard;
}
```

I. Implementieren Sie mit einem Lambda Ausdruck das Interface ‚FieldPositionProjector'. Hierbei soll Ihre Implementierung die Variable ‚pos' bijektiv serialisieren.
II. Implementieren Sie die Klasse ‚Board'. Hierbei soll Ihre Implementierung den übergebenen Parameter in die Map einfügen und die Position ‚positionOnBoard' ausgeben.
III. Instanziieren Sie in einer Main-Methode eine ‚Board' Instanz mit ‚Entity' Instanzen, welche in einem 4x4 Gitter angeordnet sind.

Die obigen Klassen dürfen hierbei nicht verändert werden!

Aufgabe 2)
Beantworten Sie folgende Fragen:

I. Wozu dienen Generics?
II. Was bedeutet Upper-Bounded bzw. Lower-Bounded bei Generics?
III. Was ist Boilerplate-Code?
IV. Was sind Lambda-Ausdrücke und wann sollten sie verwendet werden?
V. Geben Sie ein Beispiel für ein Interface, welches nicht durch einen Lambda-Ausdruck implementiert werden kann.