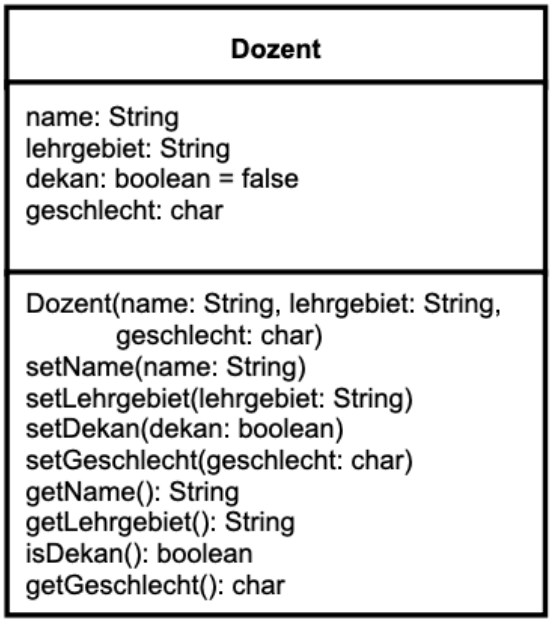
VL09-Aufgabe 3 (Praktikum)

1. Erstellen Sie in Eclipse ein neues Projekt mit dem Namen `EidP_VL09`.
2. Fügen Sie dem Projekt eine leere Java-Datei mit dem Namen `Dozent.java` hinzu und übertragen Sie die UML-Notation der Klasse `Dozent` – mit Attributen, Konstruktor und Methoden in eine Java-Klasse.



3. Fügen Sie dem Projekt eine zweite Klasse `DozentTest` mit einem Hauptprogramm (main-Methode) mit folgenden Eigenschaften hinzu:

   - Die beiden Referenzattribute `dieDozentin` und `derDekan` sollen deklariert werden.
   - Die beiden Objekte sollen erzeugt werden.
   - Alle Attribute des Dekans sollen durch Methodenaufruf gelesen und auf der Konsole ausgegeben werden.
   - Das Lehrgebiet der Dozentin soll durch Methodenaufruf geändert werden.

Die folgende Klasse enthält mehrere Methoden namens `hoppla`:

```
public class Signatur
{
	public static void hoppla(long x, long y, double z)
	{
		System.out.println("lld");
	}

	public static void hoppla(long x, long y, long z)
	{
		System.out.println("lll");
	}

	public static void hoppla(double x, long y, double z)
	{
		System.out.println("dld");
	}

	public static void main(String[] args)
	{
		long a = 333;
		double b = 4.44;

		hoppla(a,a,b); // Aufruf 1
		hoppla(b,b,b); // Aufruf 2
		hoppla(b,a,b); // Aufruf 3
		hoppla(b,a,a); // Aufruf 4
	}
}
```

Überlegen Sie, welche der vier Methodenaufrufe unzulässig sind. Geben Sie bei den zulässigen Aufrufen an, was auf dem Bildschirm ausgegeben wird.

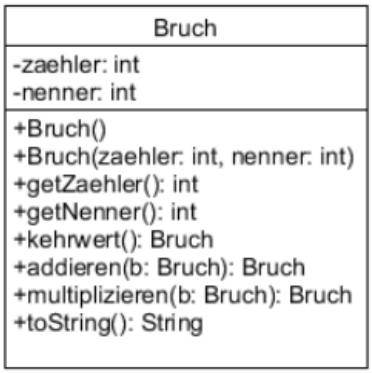
Betrachten Sie die folgende Java-Klasse.

Welche Attributdeklarationen und welche Zuweisungen sind fehlerhaft? Begründen Sie Ihre Antwort.

```
1   class Attribute
2   {
3           static int attribut1;
4           boolean attribut2;
5
6           void test()
7           {
8                   boolean attribut2;
9                   static double attribut3;
10                  double attribut4;
11
12                  attribut2 = true;
13                  {
14                          double attribut4;
15                          double attribut5;
16
17                          attribut5 = 39.5;
18                          attribut1 = attribut1 + 1;
19                  }
20                  attribut5 = 1.23;
21          }
22
23          void test2(boolean attribut2)
24          {
25                  boolean attribut2;
26                  attribut4 = 17.5;
27          }
28  }
```

Erstellen Sie in Eclipse ein neues Projekt mit dem Namen `EidP_VL10`. Fügen Sie dem Projekt zwei leere Java-Dateien mit den Namen `Bruch.java` und `BruchTest.java` hinzu.

a) Programmieren Sie die Klasse `Bruch` entsprechend dem folgenden UML-Klassendiagram:



- Der parameterlose Konstruktor soll Zähler und Nenner mit 1 initialisieren.
- Die Funktion `kehrwert` soll ein neues Bruch-Objekt, bei dem Zähler und Nenner vertauscht sind, als Ergebnis liefern.
- Die Methoden `addieren` und `multiplizieren` sollen jeweils ein neues Bruch-Objekt als Ergebnis liefern. Überlegen Sie, wie zwei Brüche addiert bzw. multipliziert werden. Das Ergebnis muss nicht gekürzt sein.
- Die Methode `toString` soll eine Zeichenkette bestehend aus dem Zähler, dem Zeichen / und dem Nenner erzeugen und zurückgeben.

b) Programmieren Sie die Klasse `BruchTest`, in deren Hauptprogramm (`main`) mehrere Objekte der Klasse `Bruch` erzeugt werden und alle Methoden mindestens einmal aufgerufen werden. Fügen Sie nach dem Aufruf jeder Rechenmethode Bildschirmausgaben auf das Konsolenfenster ein, um die Berechnungen zu überprüfen.

Hinweis: Da in der Klasse `Bruch` die öffentliche Methode `toString` implementiert ist, können Sie eine Variable `b` vom Typ `Bruch` direkt wie folgt auf der Konsole ausgeben: `System.out.println(b);`