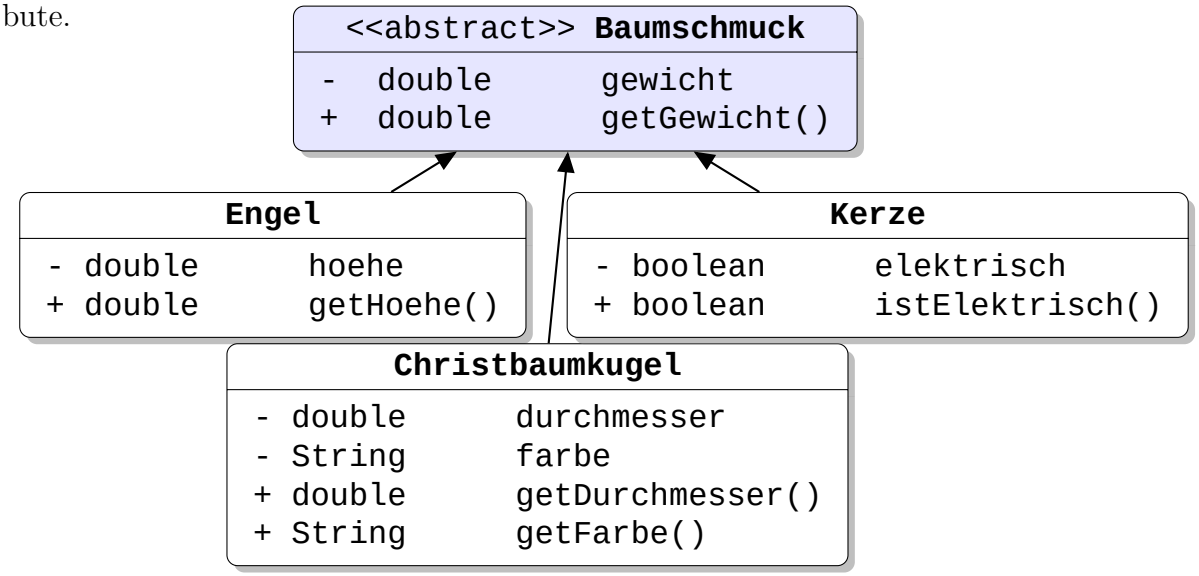
RWTH Aachen
Lehrgebiet Theoretische Informatik
Frohn–Reidl–Rossmanith–Sánchez–Ströder

Minipräsenzübung zur Vorlesung Programmierung

Aufgabe 8 (10 Punkte)

Betrachten Sie folgendes Klassendiagramm. Die Sichtbarkeiten + und - bedeuten `public` und `private`. Die Methoden in den Klassen sind Getter für die entsprechenden Attribute.



Implementieren Sie die untenstehende Methode in der Klasse `Baumschmuck`, welche den größten Durchmesser zurückgeben soll, der bei den im übergebenen Array enthaltenen Christbaumkugeln vorkommt. Sollten keine Christbaumkugeln vorhanden sein, soll die Methode `0.0` zurückgeben.

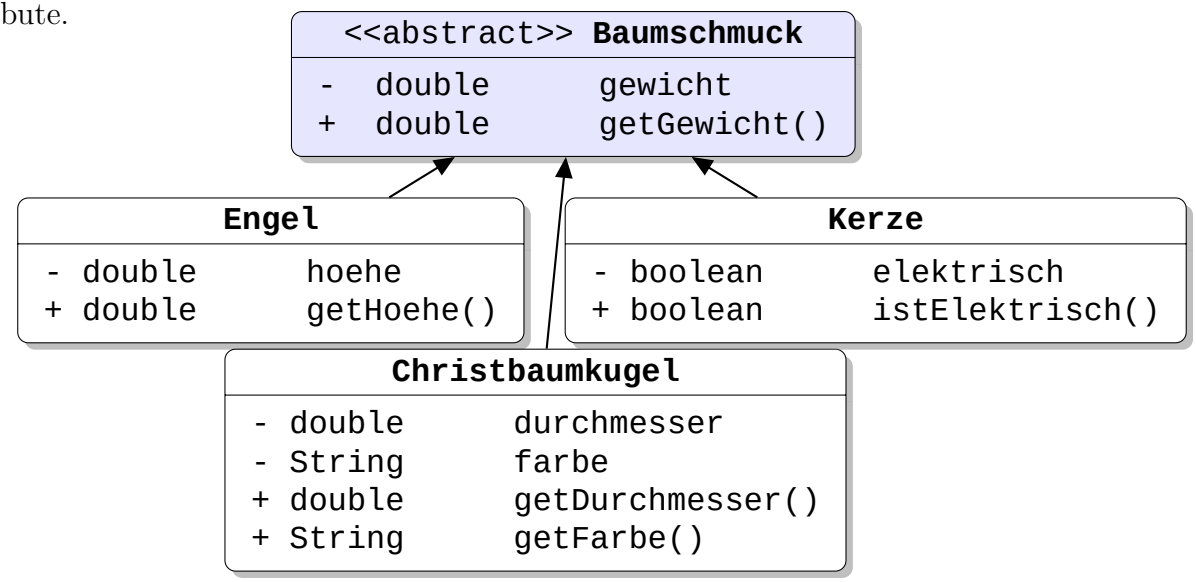
```
public static double maxDurchmesser(Baumschmuck[] schmuck) {

}
```

RWTH Aachen
Lehrgebiet Theoretische Informatik
Frohn–Reidl–Rossmanith–Sánchez–Ströder

Minipräsenzübung zur Vorlesung Programmierung

Aufgabe 8 (10 Punkte)

Betrachten Sie folgendes Klassendiagramm. Die Sichtbarkeiten + und - bedeuten `public` und `private`. Die Methoden in den Klassen sind Getter für die entsprechenden Attribute.



Implementieren Sie die untenstehende Methode in der Klasse `Baumschmuck`, welche die Höhe des größten Engels zurückgeben soll, der im übergebenen Array vorkommt. Sollten keine Engel vorhanden sein, soll die Methode `0.0` zurückgeben.

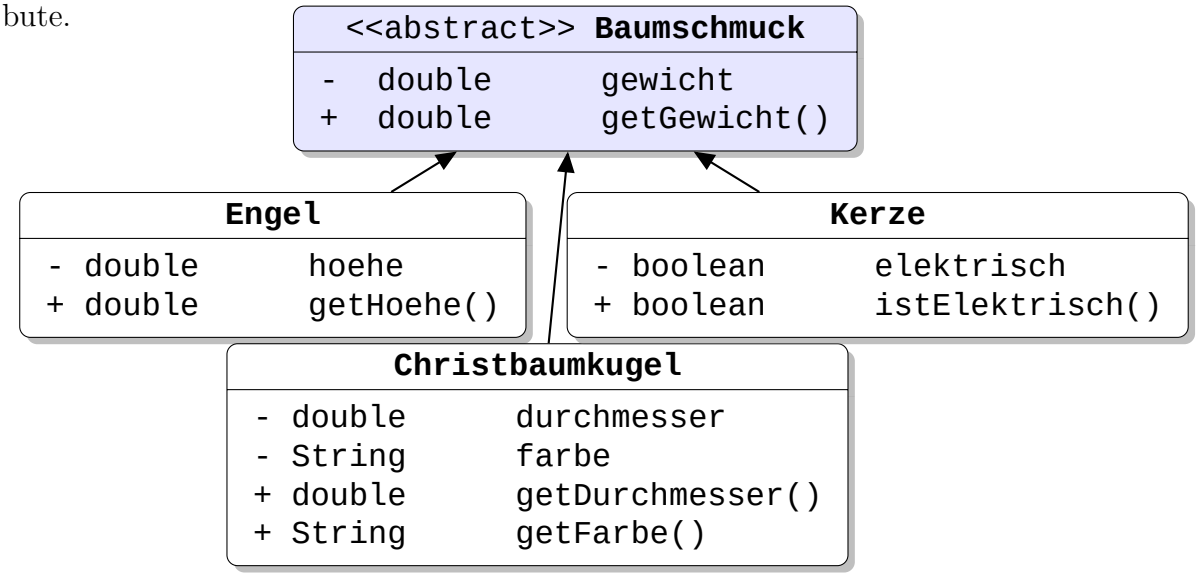
```
public static double maxEngelHoehe(Baumschmuck[] schmuck) {




}
```

RWTH Aachen
Lehrgebiet Theoretische Informatik
Frohn–Reidl–Rossmanith–Sánchez–Ströder

Minipräsenzübung zur Vorlesung Programmierung

Aufgabe 8 (10 Punkte)

Betrachten Sie folgendes Klassendiagramm. Die Sichtbarkeiten + und - bedeuten `public` und `private`. Die Methoden in den Klassen sind Getter für die entsprechenden Attribute.



Implementieren Sie die untenstehende Methode in der Klasse `Baumschmuck`, welche das Gesamtgewicht aller roten Christbaumkugeln (deren `farbe` Attribut den Wert "`rot`" hat) im übergebenen Array zurückgibt. Sie können davon ausgehen, dass das `farbe` Attribut niemals `null` ist.

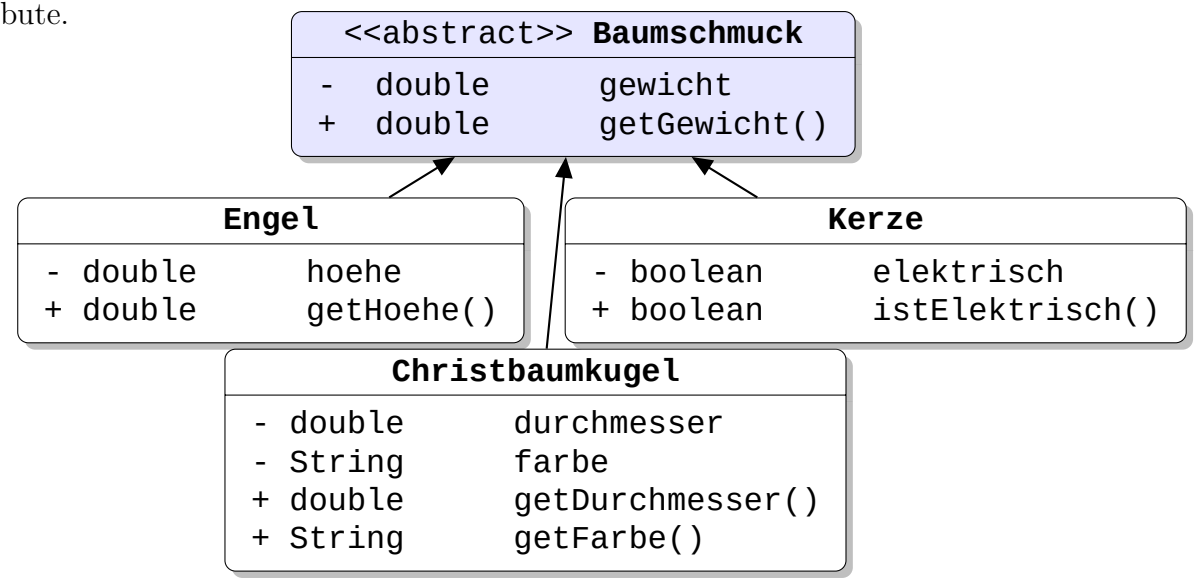
```
public static double gewichtRoterKugeln(Baumschmuck[] schmuck) {

}
```

RWTH Aachen
Lehrgebiet Theoretische Informatik
Frohn–Reidl–Rossmanith–Sánchez–Ströder

Minipräsenzübung zur Vorlesung Programmierung

Aufgabe 8 (10 Punkte)

Betrachten Sie folgendes Klassendiagramm. Die Sichtbarkeiten + und - bedeuten `public` und `private`. Die Methoden in den Klassen sind Getter für die entsprechenden Attribute.



Implementieren Sie die untenstehende Methode in der Klasse `Baumschmuck`, welche `true` genau dann zurückgeben soll, wenn im übergebenen Array eine nicht elektrische Kerze vorkommt. Ansonsten soll die Methode `false` zurückgeben.

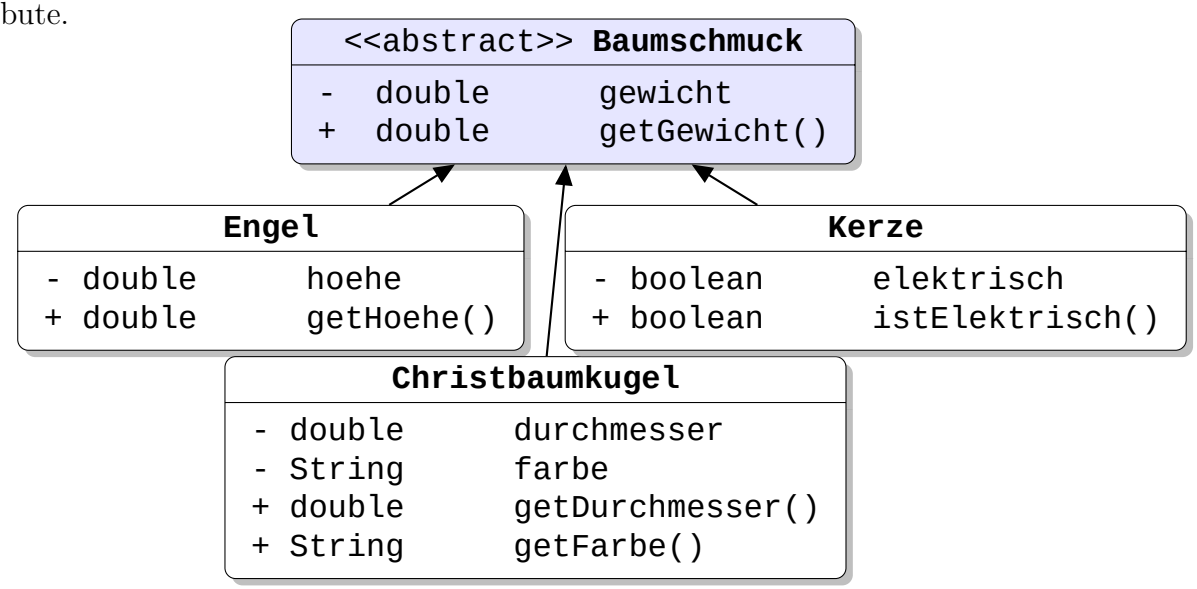
```
public static boolean istGefaehrlich(Baumschmuck[] schmuck) {

}
```

RWTH Aachen
Lehrgebiet Theoretische Informatik
Frohn–Reidl–Rossmanith–Sánchez–Ströder

Minipräsenzübung zur Vorlesung Programmierung

Aufgabe 8 (10 Punkte)

Betrachten Sie folgendes Klassendiagramm. Die Sichtbarkeiten + und - bedeuten `public` und `private`. Die Methoden in den Klassen sind Getter für die entsprechenden Attribute.



Implementieren Sie die untenstehende Methode in der Klasse `Baumschmuck`, welche die Anzahl an elektrischen Kerzen im übergebenen Array zurückgibt.

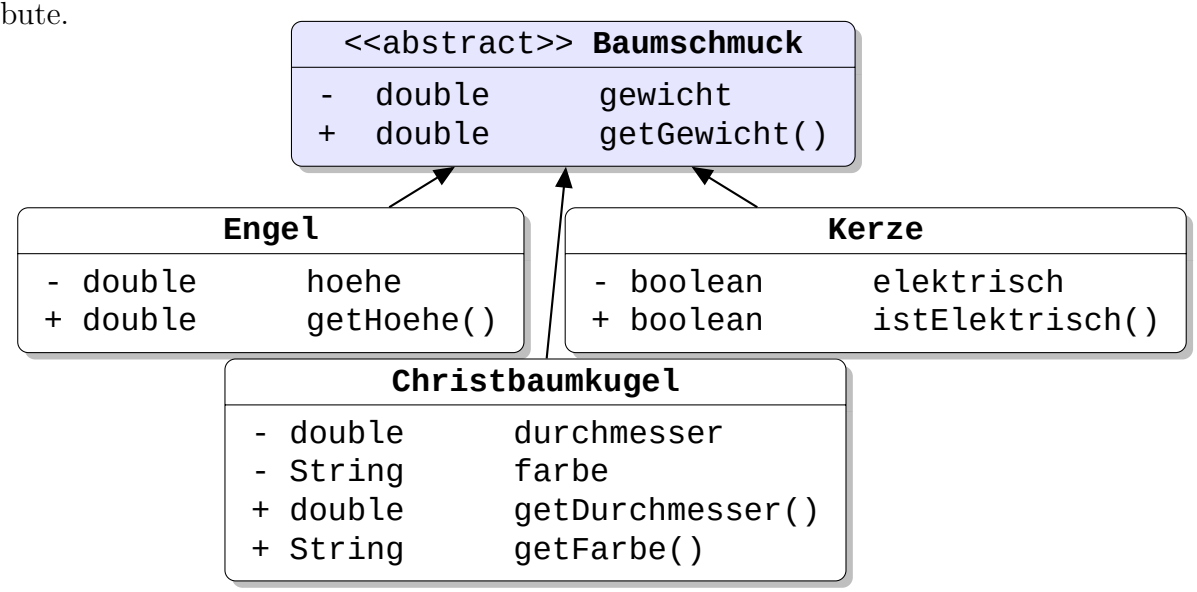
```
public static int anzElektroKerzen(Baumschmuck[] schmuck) {

}
```

RWTH Aachen
Lehrgebiet Theoretische Informatik
Frohn–Reidl–Rossmanith–Sánchez–Ströder

Minipräsenzübung zur Vorlesung Programmierung

Aufgabe 8 (10 Punkte)

Betrachten Sie folgendes Klassendiagramm. Die Sichtbarkeiten + und - bedeuten `public` und `private`. Die Methoden in den Klassen sind Getter für die entsprechenden Attribute.



Implementieren Sie die untenstehende Methode in der Klasse `Baumschmuck`, welche den größten Durchmesser zurückgeben soll, der bei den im übergebenen Array enthaltenen Christbaumkugeln vorkommt. Sollten keine Christbaumkugeln vorhanden sein, soll die Methode `0.0` zurückgeben.

Lösungsvorschlag:

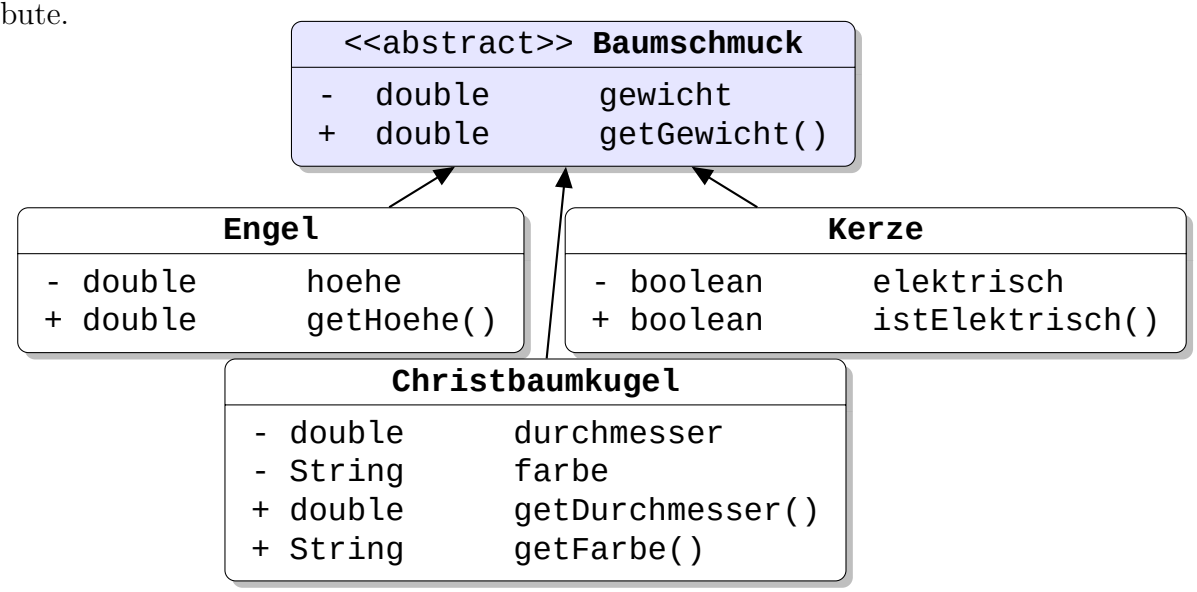
```
public static double maxDurchmesser(Baumschmuck[ ] schmuck) {
    double res = 0.0;
    for (int i = 0; i < schmuck.length; i++) {
        if (schmuck[i] instanceof Christbaumkugel) {
            Christbaumkugel kugel = (Christbaumkugel)schmuck[i];
            if (kugel.getDurchmesser() > res) {
                res = kugel.getDurchmesser();
            }
        }
    }
    return res;
}
```

RWTH Aachen
Lehrgebiet Theoretische Informatik
Frohn–Reidl–Rossmanith–Sánchez–Ströder

Minipräsenzübung zur Vorlesung Programmierung

Aufgabe 8 (10 Punkte)

Betrachten Sie folgendes Klassendiagramm. Die Sichtbarkeiten + und - bedeuten `public` und `private`. Die Methoden in den Klassen sind Getter für die entsprechenden Attribute.



Implementieren Sie die untenstehende Methode in der Klasse `Baumschmuck`, welche die Höhe des größten Engels zurückgeben soll, der im übergebenen Array vorkommt. Sollten keine Engel vorhanden sein, soll die Methode `0.0` zurückgeben.

Lösungsvorschlag:

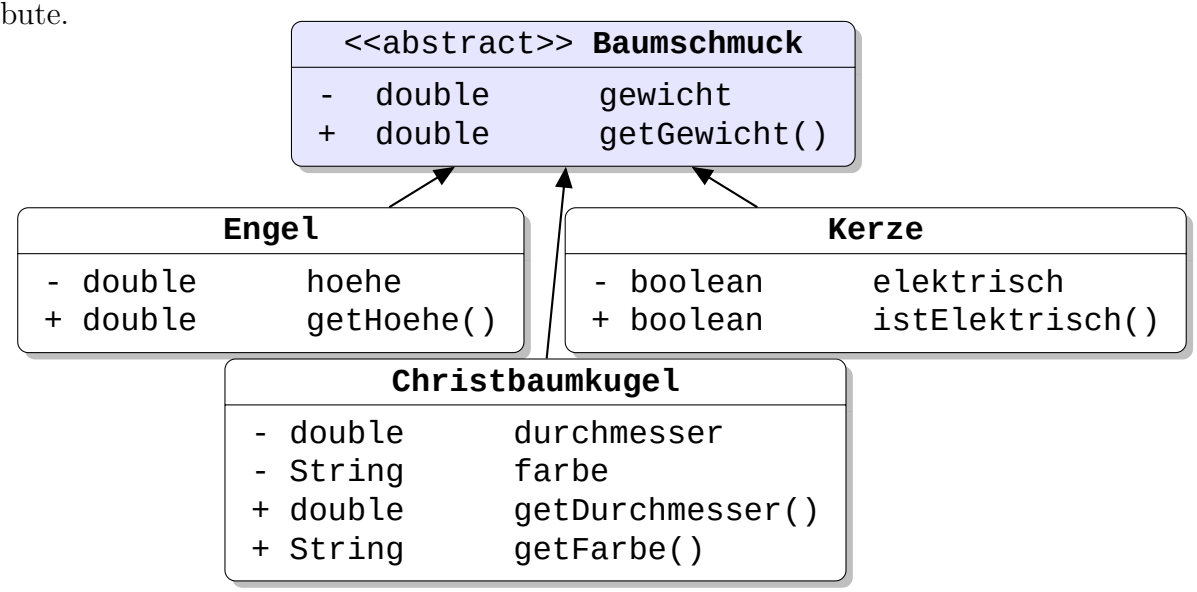
```
public static double maxEngelHoehe(Baumschmuck[ ] schmuck) {
    double res = 0.0;
    for (int i = 0; i < schmuck.length; i++) {
        if (schmuck[i] instanceof Engel) {
            Engel engel = (Engel)schmuck[i];
            if (engel.getHoehe() > res) {
                res = engel.getHoehe();
            }
        }
    }
    return res;
}
```

RWTH Aachen
Lehrgebiet Theoretische Informatik
Frohn–Reidl–Rossmanith–Sánchez–Ströder

Minipräsenzübung zur Vorlesung Programmierung

Aufgabe 8 (10 Punkte)

Betrachten Sie folgendes Klassendiagramm. Die Sichtbarkeiten + und - bedeuten `public` und `private`. Die Methoden in den Klassen sind Getter für die entsprechenden Attribute.



Implementieren Sie die untenstehende Methode in der Klasse `Baumschmuck`, welche das Gesamtgewicht aller roten Christbaumkugeln (deren `farbe` Attribut den Wert "`rot`" hat) im übergebenen Array zurückgibt. Sie können davon ausgehen, dass das `farbe` Attribut niemals `null` ist.

Lösungsvorschlag:

```
public static double gewichtRoterKugeln(Baumschmuck[ ] schmuck) {
    double res = 0.0;
    for (int i = 0; i < schmuck.length; i++) {
        if (schmuck[i] instanceof Christbaumkugel
            && ((Christbaumkugel)schmuck[i]).getFarbe().equals("rot")
        ) {
            res += schmuck[i].getGewicht();
        }
    }
    return res;
}
```

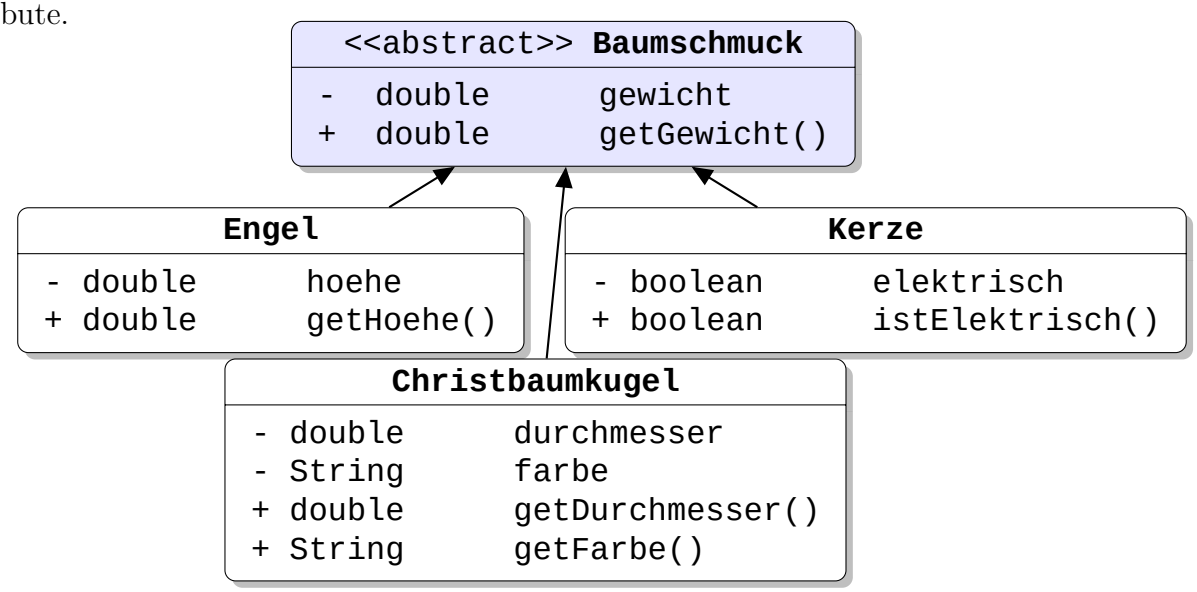
RWTH Aachen
Lehrgebiet Theoretische Informatik
Frohn–Reidl–Rossmanith–Sánchez–Ströder

WS 2013/14
Blatt 8 Gruppe D

Minipräsenzübung zur Vorlesung Programmierung

Aufgabe 8 (10 Punkte)

Betrachten Sie folgendes Klassendiagramm. Die Sichtbarkeiten + und - bedeuten `public` und `private`. Die Methoden in den Klassen sind Getter für die entsprechenden Attribute.



Implementieren Sie die untenstehende Methode in der Klasse `Baumschmuck`, welche `true` genau dann zurückgeben soll, wenn im übergebenen Array eine nicht elektrische Kerze vorkommt. Ansonsten soll die Methode `false` zurückgeben.

Lösungsvorschlag:

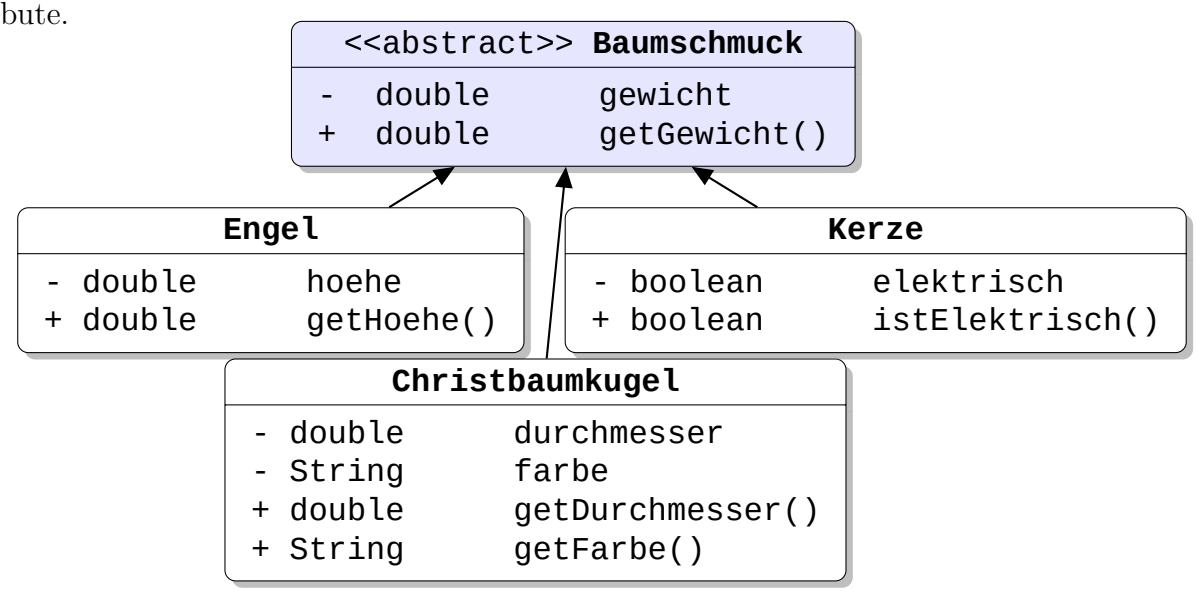
```
public static boolean istGefaehrlich(Baumschmuck[ ] schmuck) {
    for (int i = 0; i < schmuck.length; i++) {
        if (schmuck[i] instanceof Kerze
            &&!((Kerze)schmuck[i]).istElektrisch()
        ) {
            return true;
        }
    }
    return false;
}
```

RWTH Aachen
Lehrgebiet Theoretische Informatik
Frohn–Reidl–Rossmanith–Sánchez–Ströder

Minipräsenzübung zur Vorlesung Programmierung

Aufgabe 8 (10 Punkte)

Betrachten Sie folgendes Klassendiagramm. Die Sichtbarkeiten + und - bedeuten `public` und `private`. Die Methoden in den Klassen sind Getter für die entsprechenden Attribute.



Implementieren Sie die untenstehende Methode in der Klasse `Baumschmuck`, welche die Anzahl an elektrischen Kerzen im übergebenen Array zurückgibt.

Lösungsvorschlag:

```
public static int anzElektroKerzen(Baumschmuck[ ] schmuck) {
    int res = 0;
    for (int i = 0; i < schmuck.length; i++) {
        if (schmuck[i] instanceof Kerze
            && ((Kerze)schmuck[i]).istElektrisch()
        ) {
            res++;
        }
    }
    return res;
}
```