Information Hiding, AST

Programmierung und Softwareentwicklung
Hörsaalübung

Klassen und Objekte

Aufgabe 2

Sie werden von der Universität beauftragt eine Software zu bauen, mit der Universitäten simuliert werden können. Dazu sollen Sie zuerst einen Entwurf des Systems erstellen.

a) Überlegen Sie sich mit Ihrem Nachbarn, welche Klassen die Universitätssimulation besitzen sollte
b) Legen Sie zu jeder Klasse Attribute und Operationen fest
c) Legen Sie die Beziehungen zwischen den Klassen fest
d) Überlegen Sie sich für die Uni Stuttgart die entsprechenden Objekte

Aufgabe 2 – Lösung

Gemeinsam an der Tafel

Lösung wird in digitaler Version dabei sein

Problem (I)

```java
public class Kreis {
    public float radius;
    public float flaeche;
    public float umfang;

    public Kreis() {
    }

    public void setzeRadiusUndBerechne(float radius) {
        this.radius = radius;
        this.berechneFlaecheUndUmfang();
    }

    public void berechneFlaecheUndUmfang() {
        float temp = (float)Math.PI * radius;
        this.flaeche = temp * radius;
        this.umfang = temp * 2.0f;
    }

    @Override
    public String toString() {
        return "Radius: " + this.radius + " | Umfang: "
            + this.umfang + " | Flaeche: " + this.flaeche;
    }
}
```

Implementierung einer Kreisklasse

- Allerdings kann folgende Ausgabe erzeugt werden…

```
C:\PSE>java Kreis
Radius: 5.0 | Umfang: 12.566371 | Flaeche: 12.566371
```

Information Hiding

Information Hiding

- Spezifizierung auf welche Operationen und Attribute von außen zugegriffen werden kann
- Verhindert unerlaubten Zugriff
  - Achtung: Keine IT Security!
- Muss sichergestellt werden, dass der Benutzer eines Objektes nur die öffentlichen Bestandteile nutzen und verändern kann

→ Wie kann dies erreicht werden?

Sichtbarkeiten (I)

Was sind Sichtbarkeitsmodifizierer?

- Innerhalb einer Klasse sind alle Operationen & Attribute für die Operationen sichtbar
- Ziel: Zugriff von außen auf Operationen und v.a. Attribute einschränken

Sichtbarkeiten (II)

Welche Sichtbarkeiten gibt es?

public

- von überall aus sichtbar und aufrufbar, keine Einschränkung

Protected

- in der eigenen Klasse, Klassen desselben Pakets und in davon abgeleiteten Kindklassen sichtbar

Default

- in Klassen desselben Pakets sichtbar

Private

- In der eigenen Klasse sichtbar

Sichtbarkeiten – Fragen

Welche Klasseninnereien sollten idR. public sein?

- Konstruktor
- getter- /setter – Operationen
- Schnittstellen (von außen zu verwendende Funktionen)

Welche Klasseninnereien sollten idR. private sein?

- alle Attribute
- Funktionen die nur für die interne Objektverwaltung benötigt oder genutzt werden sollen

Sichtbarkeiten (III) – Zusammenfassung

Modifier	Class	Package	Subclass	World
public				
protected				
default				
private				

Sichtbarkeiten (IV) – Beispiel und Aufgabe

```java
package de.unistuttgart.pse.vu1;

public class Foo {
    private static class Bar {
        public int corge;
        private int grault;
    }
    public static class Qax {
        public int garply;
        private int waldo;
    }

    private int foo;
    protected int bar;
    int qux;
    public int qax;
}
```

```java
package de.unistuttgart.pse.vu1.test;

public class Qux {
    Foo foo = new Foo();
    // (1) Auf was kann ich hier zugreifen?
}
```

```java
package de.unistuttgart.pse.vu1;

public class quux {
    Foo foo = new Foo();
    // (2) Auf was kann ich hier zugreifen?
}
```

Abstrakter Syntaxbaum

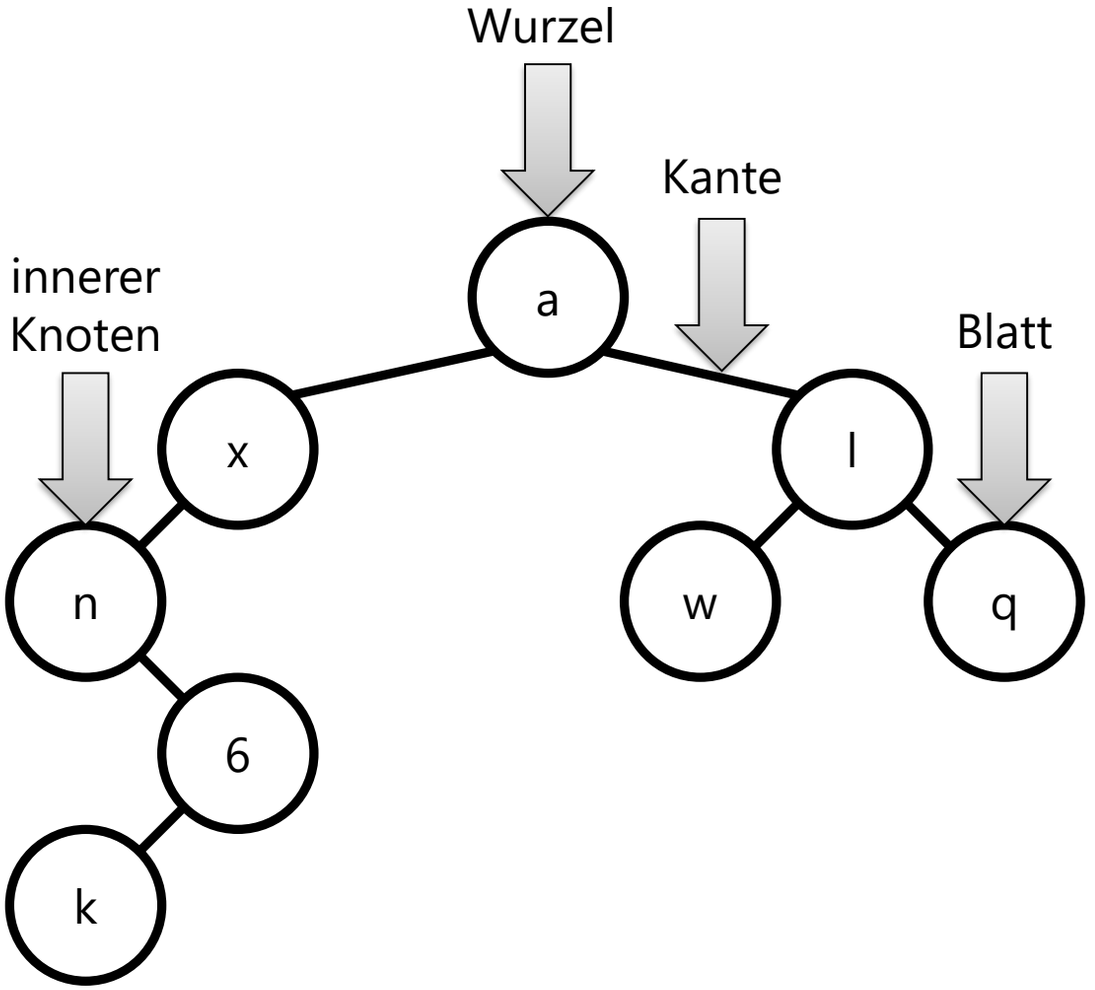
Abstract Syntax Tree, AST

Einschub: Bäume

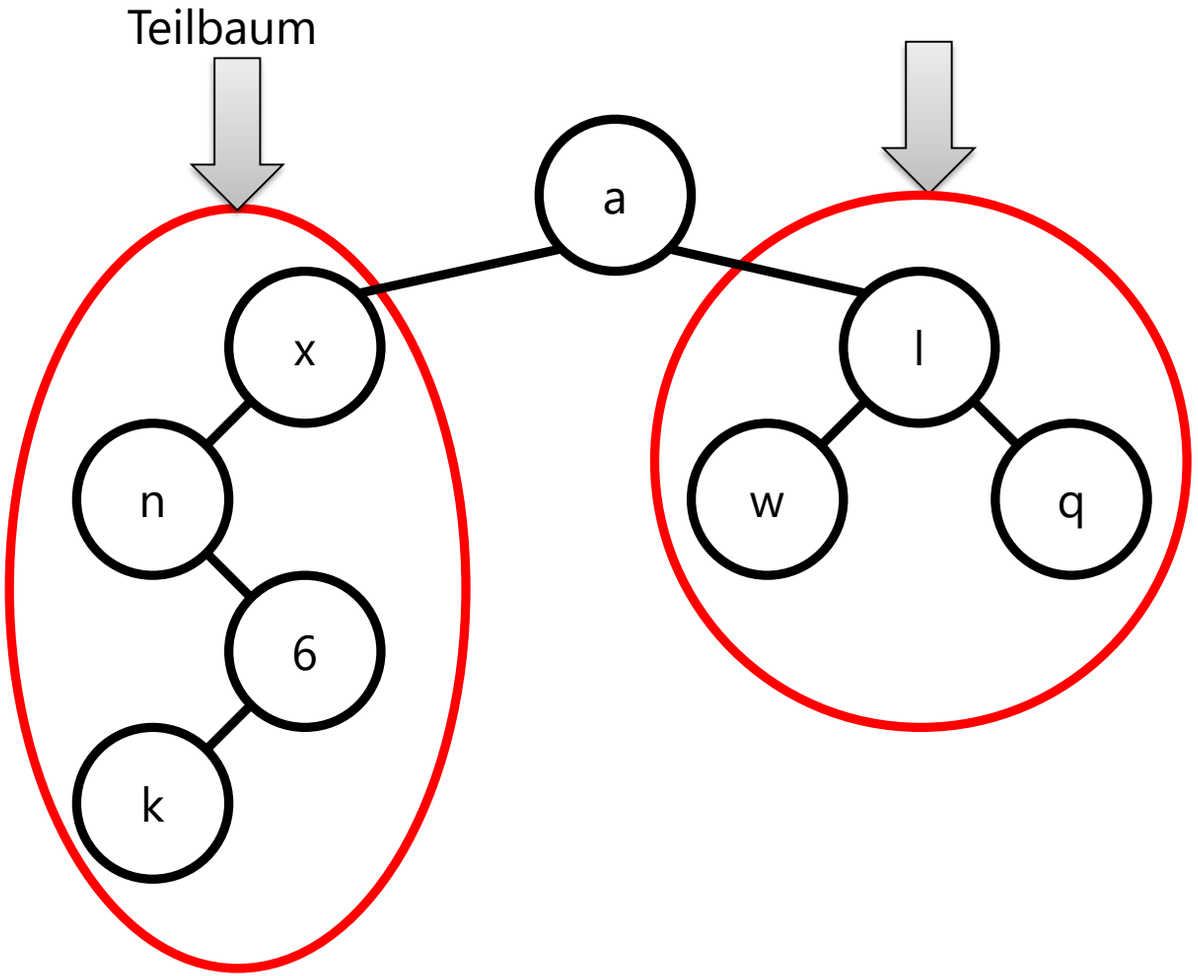
Baum

- Datenstruktur
- Speichert Werte, zum Beispiel Zahlen, Objekte, Texte
- Besitzt eine Ordnung

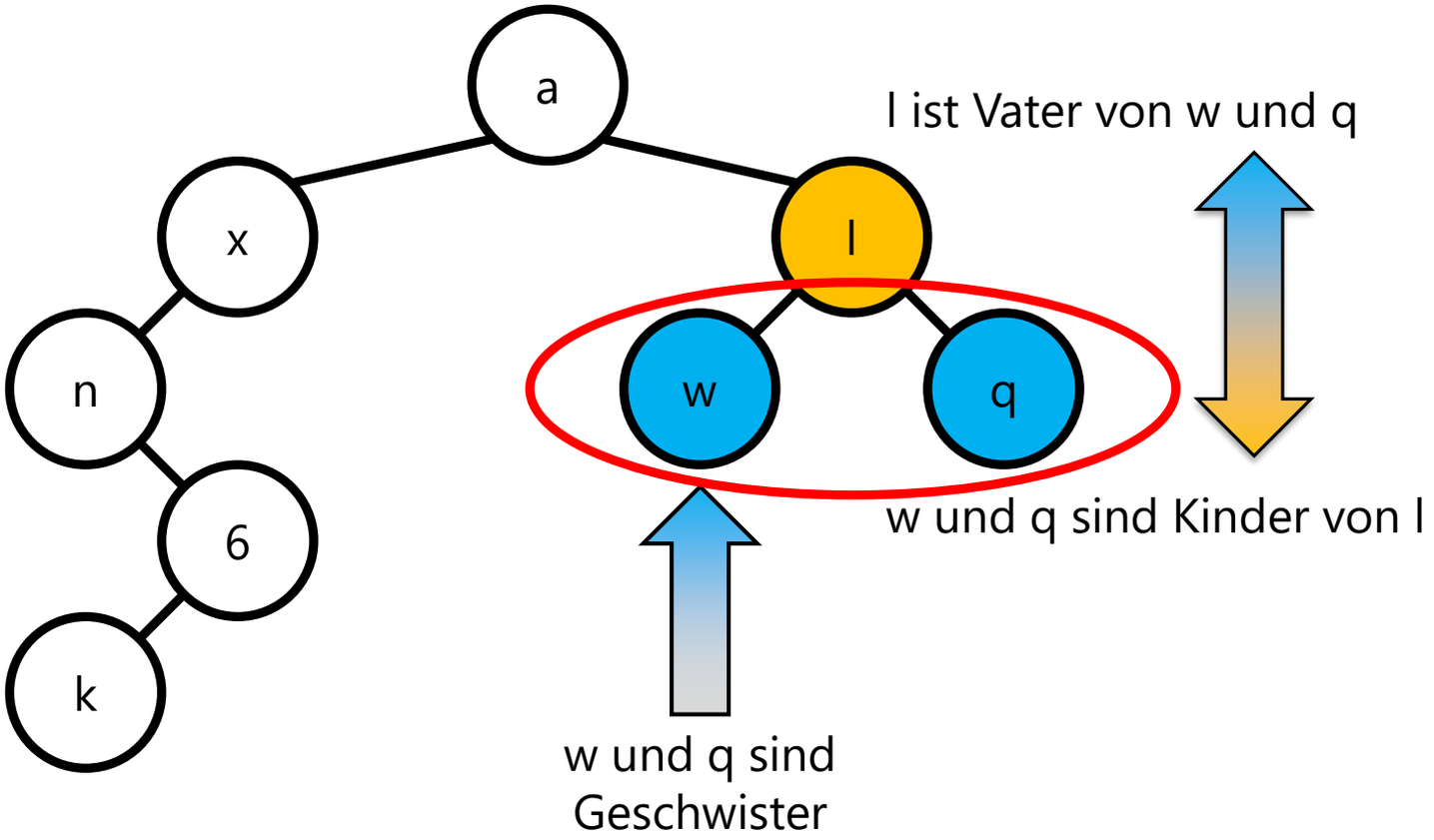
Bäume (I)



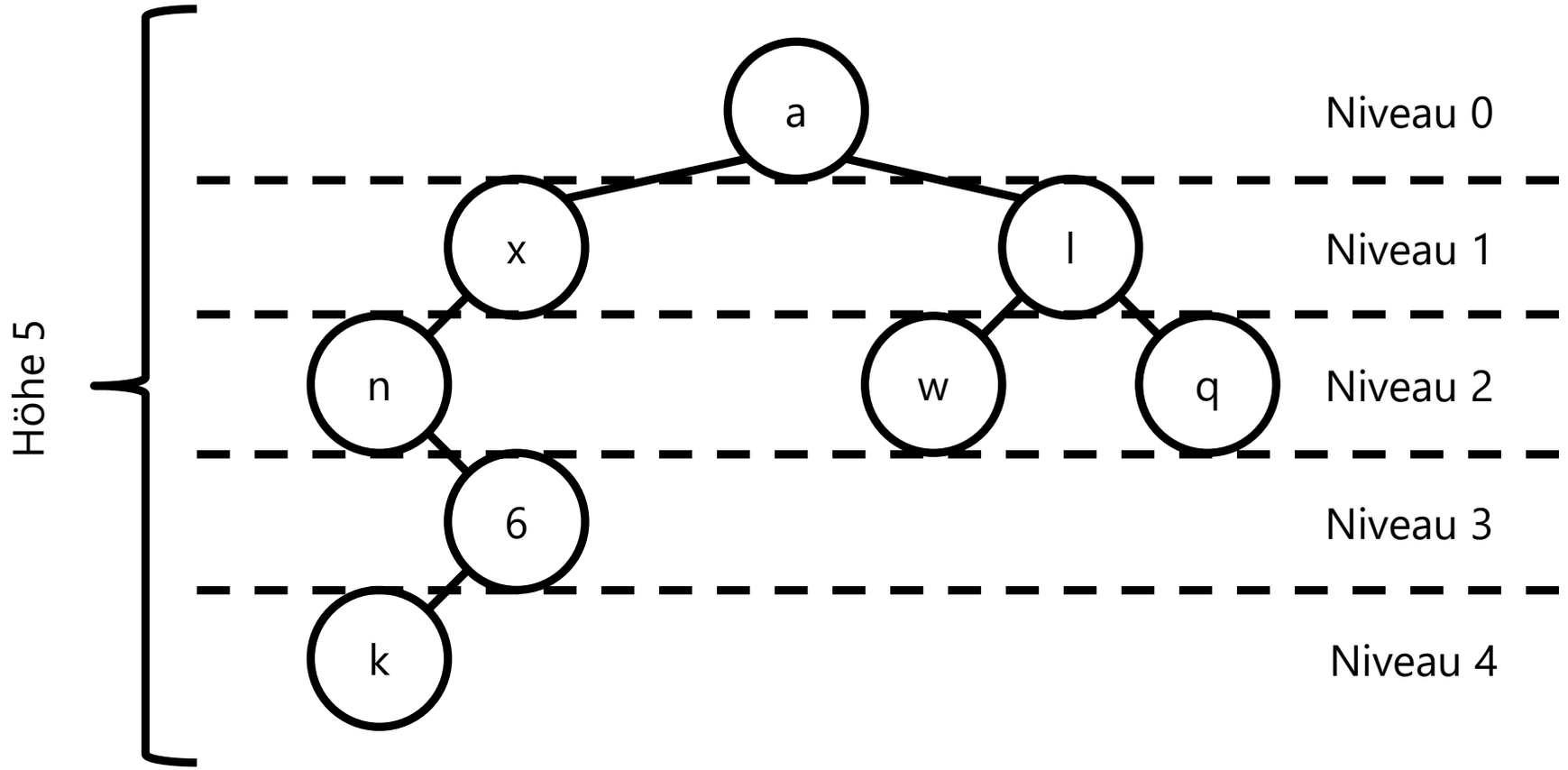
Bäume (II)



Bäume (III)


a
x
l
n
w
q
6
k
l ist Vater von w und q
w und q sind Kinder von l
w und q sind
Geschwister

Bäume (IV)



AST (I)

Abstract Syntax Tree (AST)

- Abstrakte Repräsentation eines Codes
- Syntaktische Details, wie Komma, etc. werden nicht dargestellt
- Darstellung in Form eines Baums
- Wurzel
  - Was wird betrachtet (Gesamtes Element)
- Innere Knoten
  - Die jeweiligen Unterkategorien oder Ausdrücke (Nichtterminale)
- Blätter
  - Bestandteile ohne weitere Verschachtelung (Terminale)

AST (II)

Abstract Syntax Tree (AST)

- Abstrakte Repräsentation eines Codes
- Syntaktische Details, wie Komma, etc. werden nicht dargestellt
- Darstellung in Form eines Baums
- Knoten vermeiden möglichst Ketten von einzelnen Kindknoten
- Operatoren, wie +, -, etc., werden als innere Knoten gespeichert

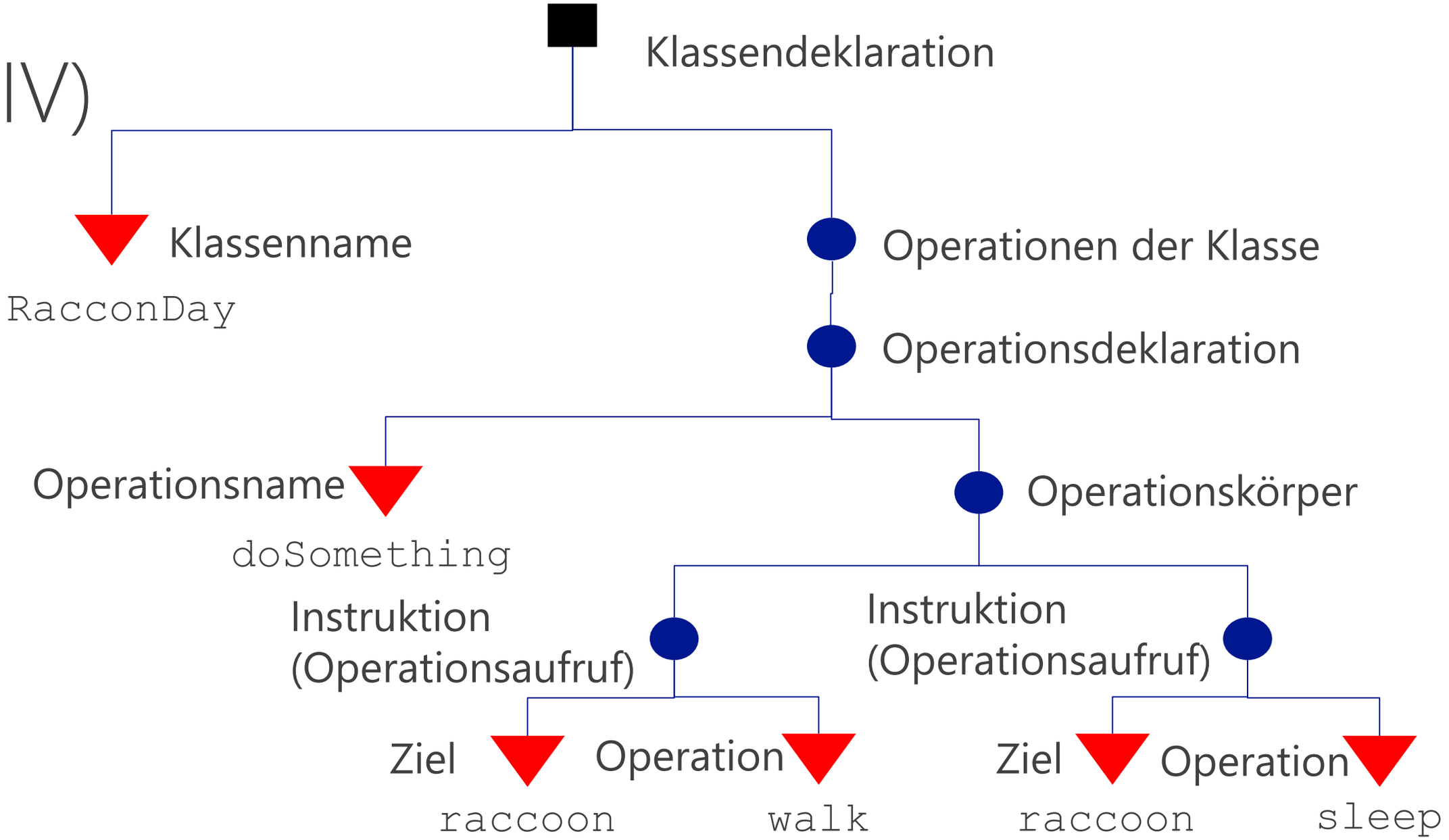
AST (III)

```
public class RaccoonDay {
    // ...

    public void doSomething() {
        raccoon.walk();
        // ...
        raccoon.sleep();
    }

}
```

AST (IV)


Klassendeklaration
Klassenname
RacconDay
Operationen der Klasse
Operationsdeklaration
Operationsname
doSomething
Operationskörper
Instruktion
(Operationsaufruf)
Instruktion
(Operationsaufruf)
Ziel
raccoon
Operation
walk
Ziel
raccoon
Operation
sleep

AST (V) – Aufgabe

Erstellen Sie den AST für die unten angegebene Java Klasse.

```
public class PSE extends Lecture {
    public void getLectureName() {
        printer.println("PSE");
    }
    public String getLecturersName(Lecturer lecturer)
     {
        return lecturer.getName();
    }
}
```

Die Besprechung erfolgt an der Tafel.