Große Übung
Praktische Informatik 1

2005-12-08

Holger Füßler
fuessler@informatik.uni-mannheim.de
http://www.informatik.uni-mannheim.de/pi4/people/fuessler

1: Announcements / Orga

Weihnachtsklausur

- zählt als Übungsblatt, d.h. die Punkte werden sowohl im Zähler, als auch im Nenner gutgeschrieben.
- 22ter Dezember, A3, 17:15 - 18:45
- gr. Übung am 15.12. entfällt leider

Organisatorisches

- nächste große Übung entfällt
- d.h. keine große Übung mehr vor der Klausur
- Blatt 8 wird erst im Januar abzugeben sein

2: Relevantes in Java

.class Files

- .class files enthalten alles, was man braucht.
- es gibt Tools, um aus .class files wieder source-code zu machen
- wenn man ein .class File ohne Implementierung hat, gibt `javap <ClassName>` das Interface der Klasse aus.

javap Beispiel

```
[fuessler@proteus pi1]$ javap List
Compiled from "List.java"
public interface List{
    public abstract java.lang.Object insertfirst(java.lang.Object);
    public abstract java.lang.Object insertlast(java.lang.Object);
    public abstract java.lang.Object removefirst();
    public abstract java.lang.Object removelast();
    public abstract java.lang.Object search(java.lang.Object);
    public abstract java.lang.String toString();
}

[fuessler@proteus pi1]$
```

Exceptions

- Möglichkeiten der Fehlerbehandlung
  - Methodenrückgabewert (schlecht, weil man den auch für anderes braucht)
  - Ausnahmen (spezieller Algorithmus)

Exception

- Exception “auswerfen” mit
  - throw Throwable, z.B. throw new Exception;
- Exception nach oben weiterreichen oder abfangen.

Exception Beispiel

```
//...
    public static int doublePositiveInt(int n) throws Exception {
        if (n < 1)
            throw new Exception("positive Int required");
        return 2*n;
    }

    public static void catchEx(int n) {
        try {
            int result = doublePositiveInt(n);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public static void handUpEx(int n) throws Exception {
        int result = doublePositiveInt(n);

    }
//...
```

Programming by Contract

- Die Signatur einer Methode sagt dem "Benutzer" alles, was er wissen muss
  - rückgabewert
  - parameter
  - Exceptions, die ausgeworfen werden können

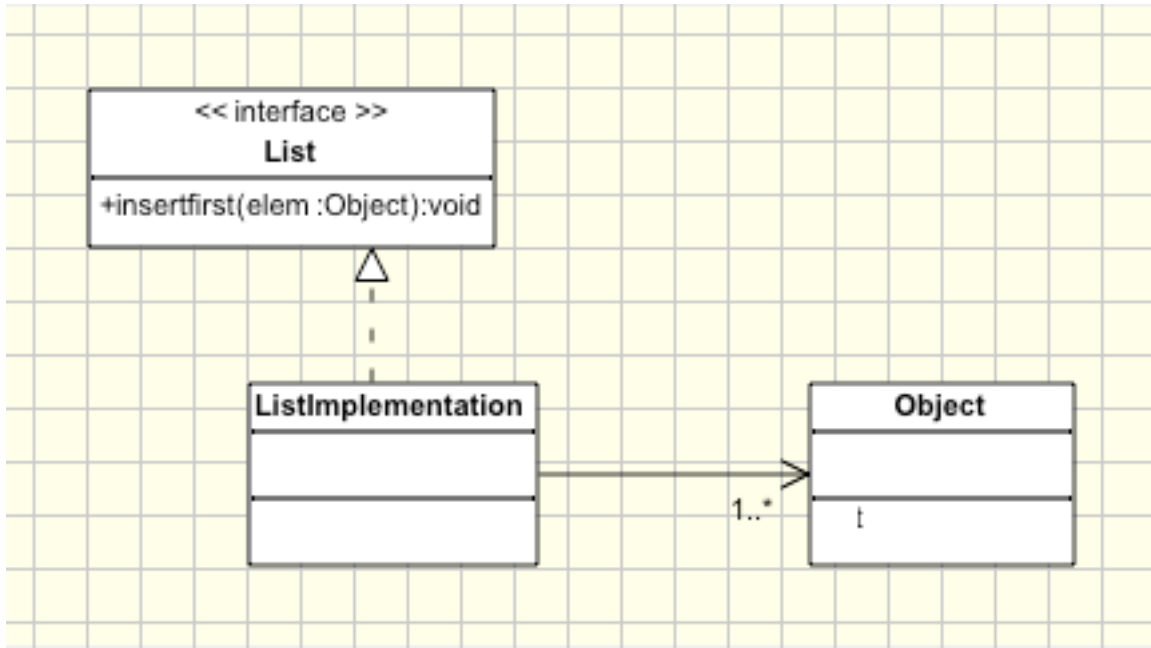
Für die Übung

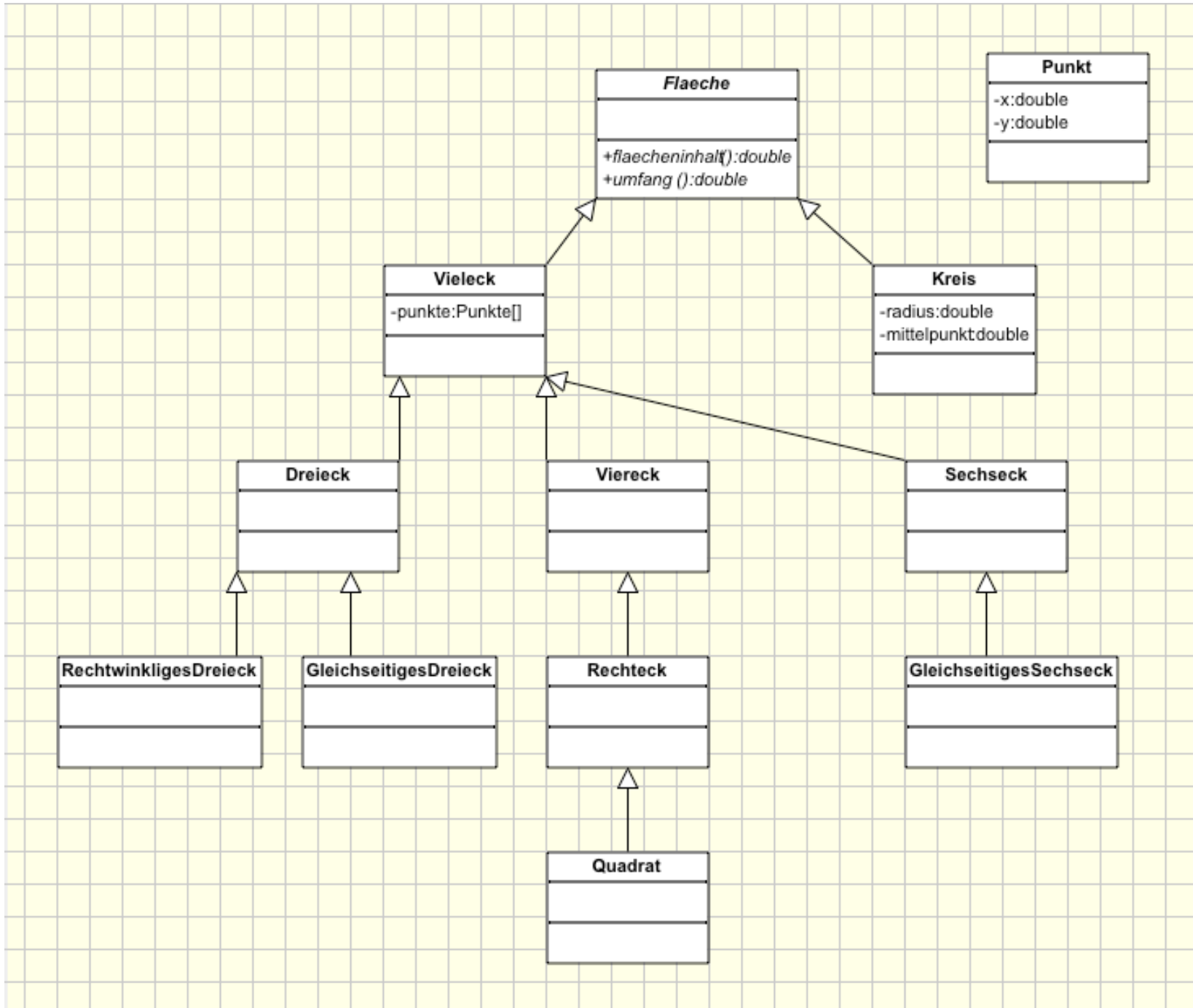
- nach oben weiterreichen ist ausreichend
- try/catch macht auch nur wirklich Sinn, wenn man den Fehler “behandeln” kann, z.B. wenn eine Datei nicht gefunden wird und man dann beim Benutzer nach einem neuen Namen fragen kann

Ein bisschen UML

- UML - Unified Modelling Language
- “Sprache” zur Objekt-Modellierung
- über ein Dutzend Diagramm-Typen
- Hier: Klassendiagramme

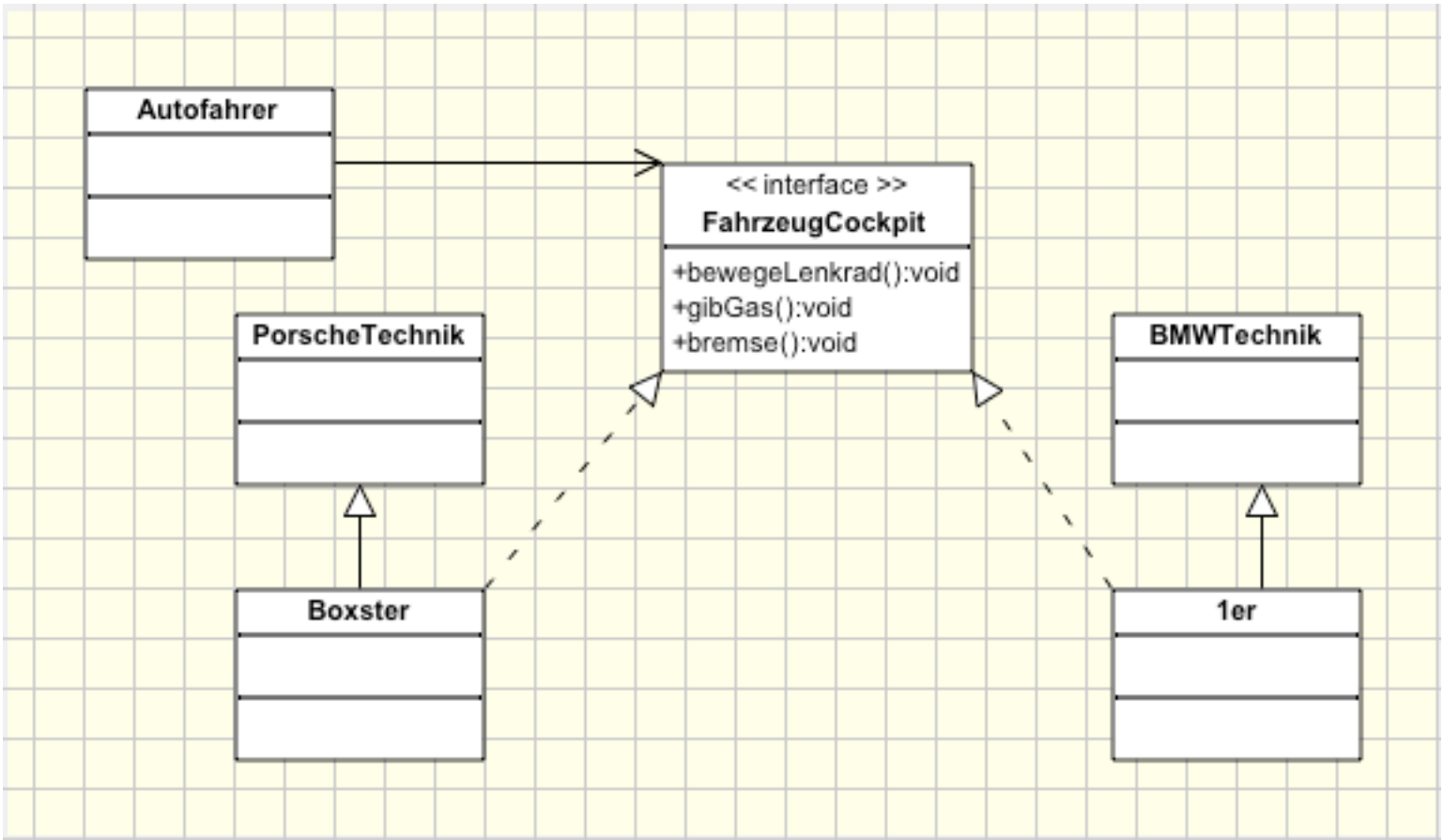
Beispiel


<< interface >>
List
+insertfirst(elem :Object):void
ListImplementation
Object
1..*
t


Flaeche
+flaecheninhalt():double
+umfang ():double
Punkt
-x:double
-y:double
Vieleck
-punkte:Punkte[]
Kreis
-radius:double
-mittelpunkt:double
Dreieck
Viereck
Sechseck
RechtwinkligesDreieck
GleichseitigesDreieck
Rechteck
GleichseitigesSechseck
Quadrat

3: Vorlesung

Interfaces


Autofahrer
<< interface >>
FahrzeugCockpit
+bewegeLenkrad():void
+gibGas():void
+bremse():void
PorscheTechnik
BMWTechnik
Boxster
1er

Interface Beispiel

```
FahrzeugCockpit fc = new Boxster();

public void umsteigen() {
    fc = new 1er();
}

public void fahre() {
    for (int i = 0; i < 100; i++)
        fc.gibGas();
}
```

Wann Interface?

- Interfaces immer dann, wenn ich die Methoden beschreiben will, die ein Objekt haben soll, ohne Funktionalität zu vererben.
- (Abstrakte) Superklassen, wenn ich auch Funktionalität / Attribute vererben will.

Datenstrukturen

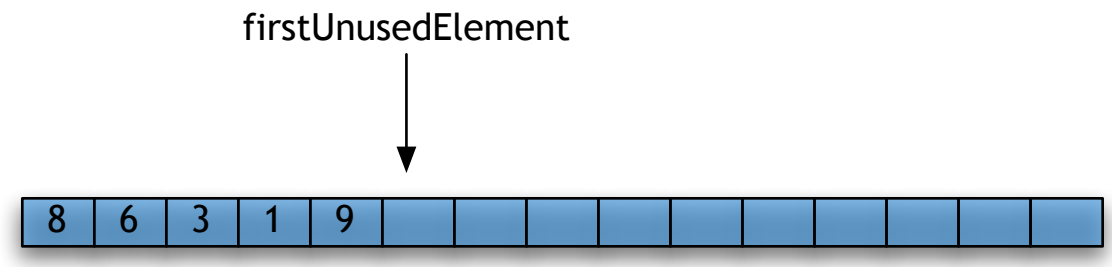
- Grundsätzlich benutzt jedes vernünftige Programm Variablen.
- Wenn man z.B. eine variable Anzahl von ints braucht, bei der ständig vorne Elemente angehängt werden, bietet sich eine Liste an.

Liste Allgemein

- Eine Liste ist ein Speicher für eine variable Anzahl von Elementen und besitzt eine interne Ordnung, d.h. die Reihenfolge bleibt erhalten.

Listenimplementierungen

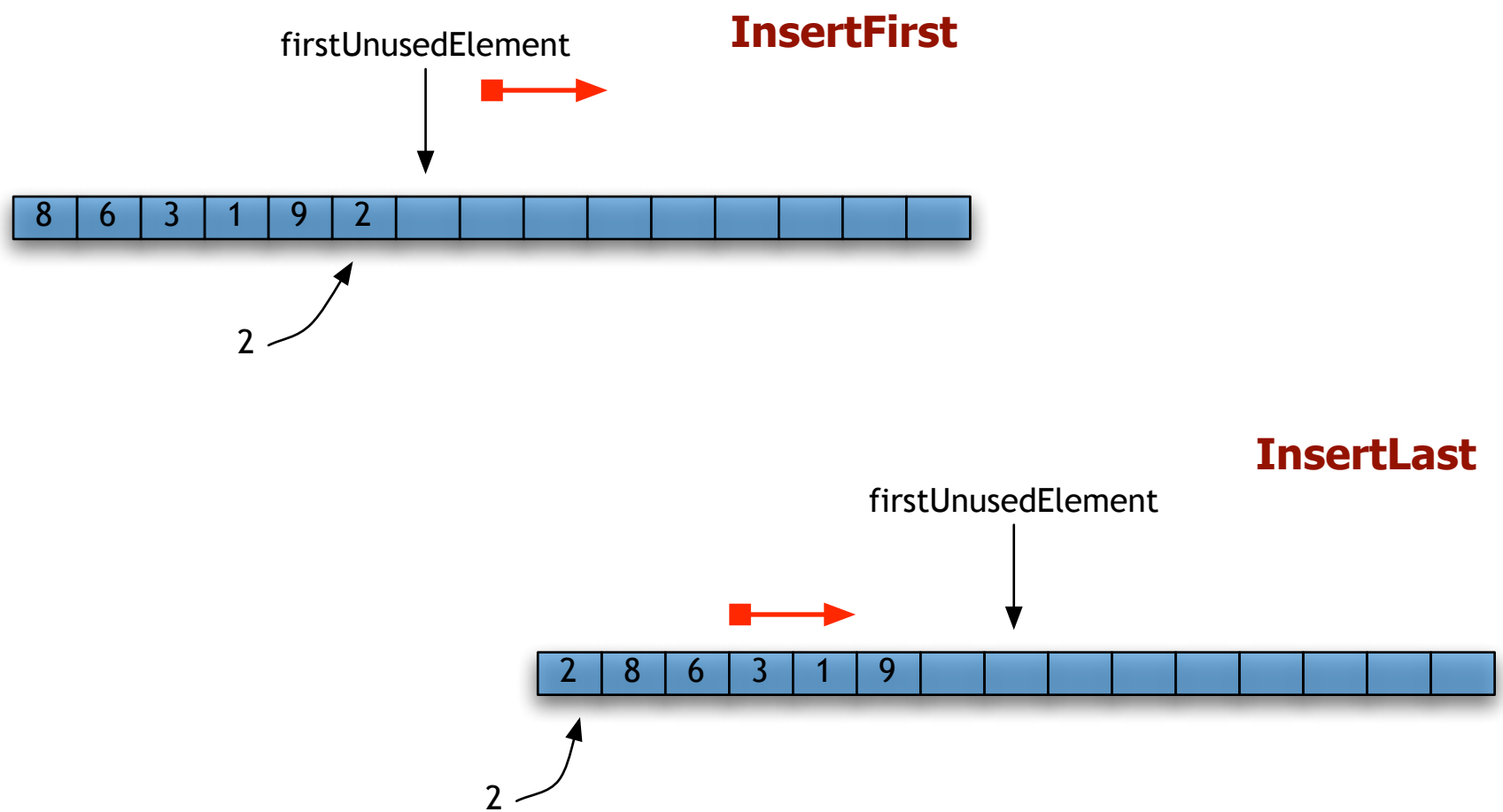
- mit einem Object[] Feld
- verkettete Objekte
  - einfach verkettet
  - doppelt verkettet



mit einem Object Feld

```
public class VectorList implements List {
     private Object[] elements;
     private int firstUnusedElement;
     private static final int bucketSize = 20;

     VectorList() {
          elements = new Object[bucketSize];
          firstUnusedElement = 0;
     }

     private void growVector(){
          Object[] newElements = new Object[elements.length + bucketSize];
          System.arraycopy(elements, 0,  newElements, 0, elements.length);
          elements = newElements;
     }

     public String toString(){
          StringBuffer sb = new StringBuffer();
          for(int i = 0; i < elements.length; i++)
               sb.append(String.format("%s:", elements[i]));
          return(sb.toString());
     }

/...

}
```

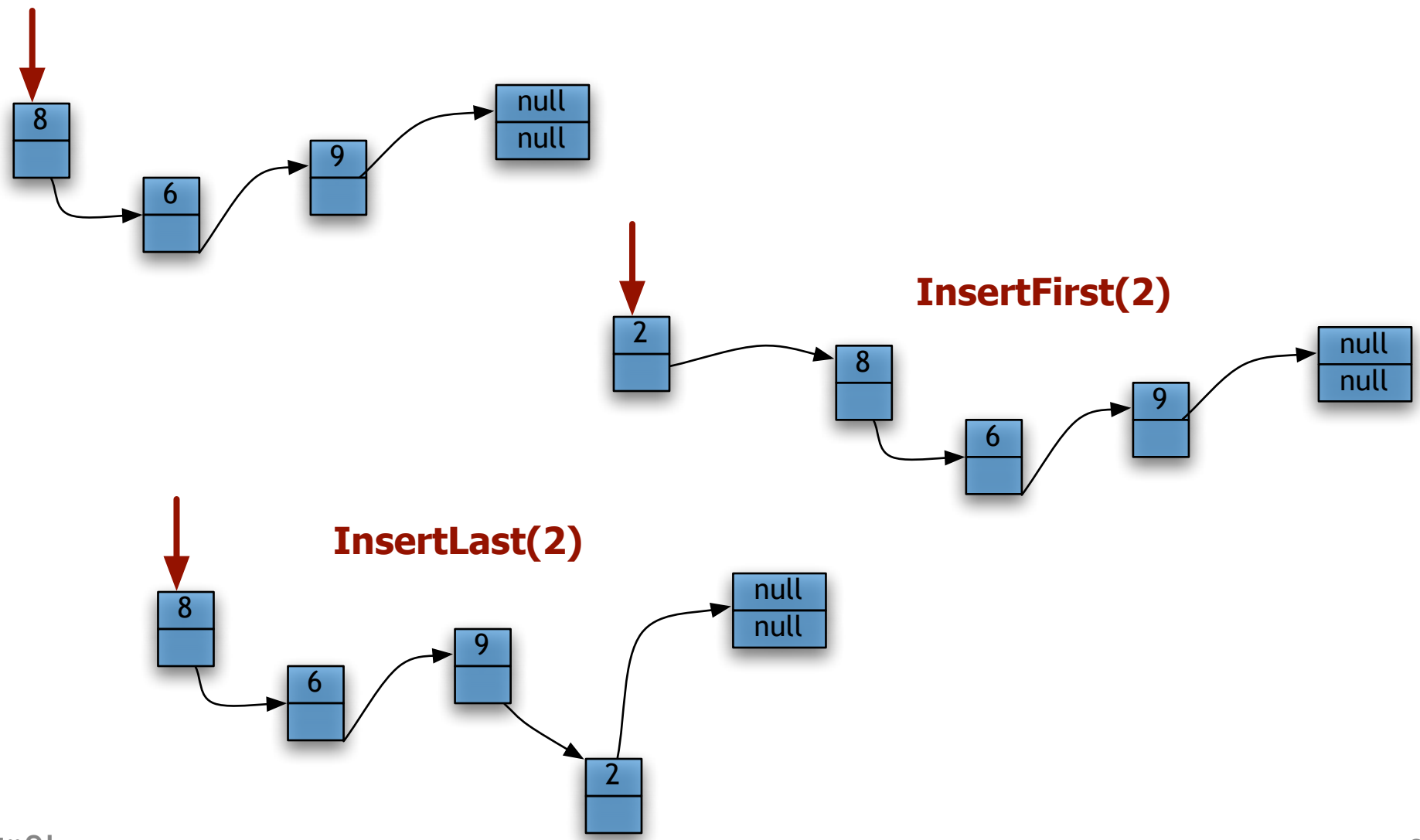
Listenoperation



Vor / Nachteile

- Datenstruktur ist übersichtlich
- ich könnte auch auf Elemente per Index zugreifen
- braucht viel Hauptspeicher
- kopier-Operationen teuer

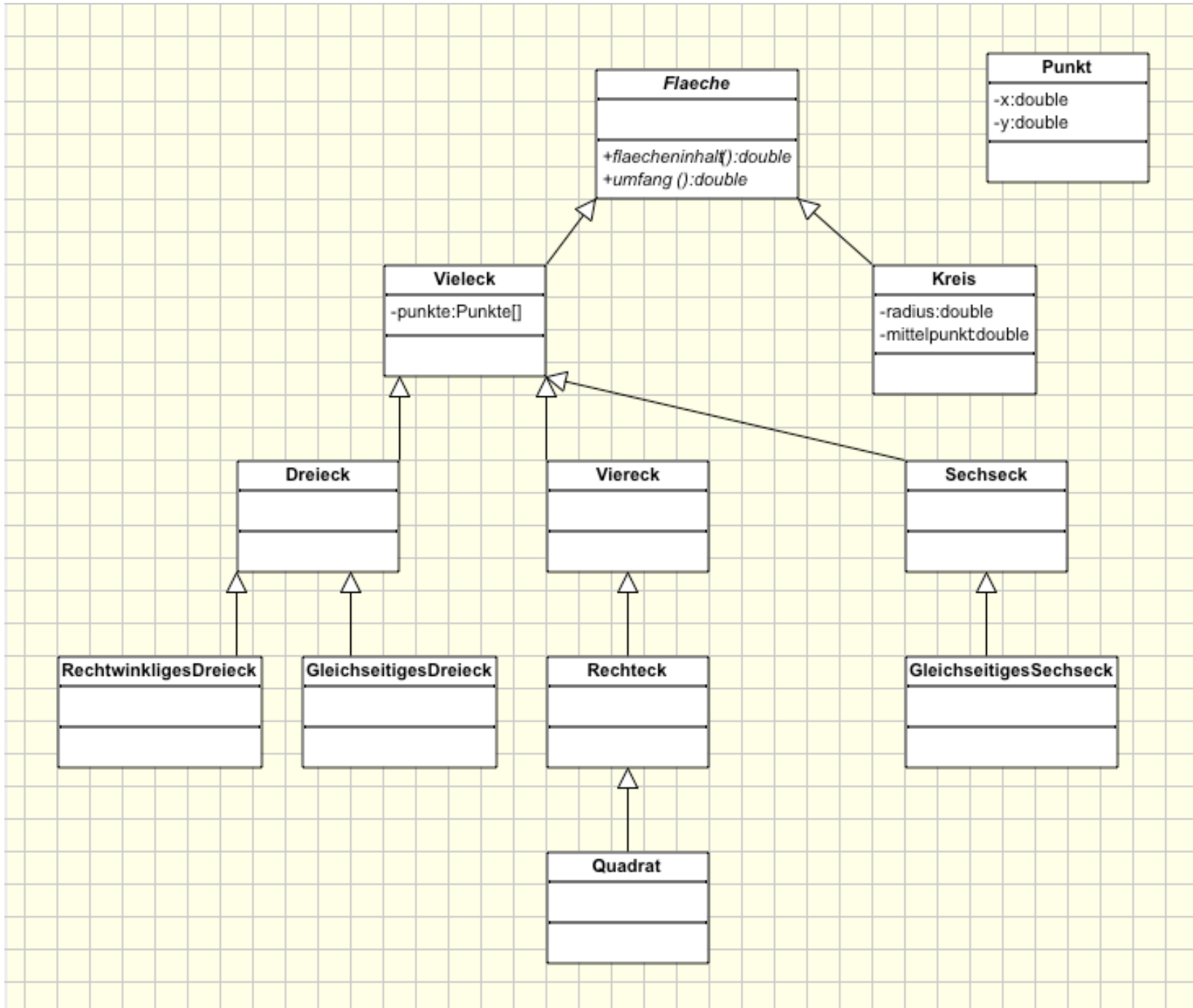
Verkettete Liste



4: Übungsblatt 4/A3

Geometrie-Aufgabe

- zunächst Vererbungs-Hierarchie aufbauen
- dann die Klassen-Dateien anlegen
- Attribute festlegen, und Konstruktoren
- Interface-Methoden implementieren


Flaeche
+flaecheninhalt():double
+umfang ():double
Punkt
-x:double
-y:double
Vieleck
-punkte:Punkte[]
Kreis
-radius:double
-mittelpunkt:double
Dreieck
Viereck
Sechseck
RechtwinkligesDreieck
GleichseitigesDreieck
Rechteck
GleichseitigesSechseck
Quadrat

5: Übungsblatt 7

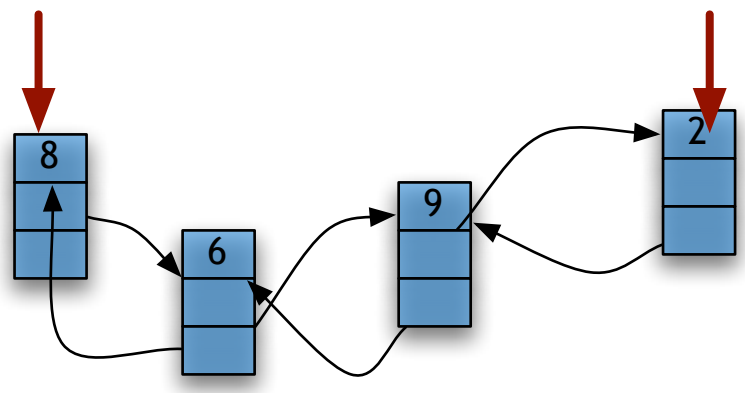
Hinweise A1

- Hanoi-Algorithmus der Vorlesung implementieren.
- Unter Verwendung der Hanoi.class
- mit javap kann man das Interface anzeigen
- Exception einfach nach oben weiterleiten

Hinweise A2

- Fibonacci mit Laufzeitmessung
- zeit speichern -> laufen lassen -> differenzzeit bilden

Hinweis A3



Hinweise A4

- Funktion erstmal in Formel umwandeln
- dann vollst. Induktion durchführen
- guter wikipedia.de Artikel vorhanden

Hinweise Blatt8

- soweit selbsterklärend
- B-Baum Operationen auf Papier
- Fibonacci-Zahlen
- bei Fragen: über Liste mailen

Ende