Kontrollfragen und Übungen

zu

Programmierung 1

Michael Lang

Inhaltsverzeichnis

1. Einführung

1.1. Kontrollfragen

1. Frage
Wie würden Sie den Begriff Algorithmus im Rahmen der Informatik definieren?

2. Frage
Beschreiben Sie die Bestandteile sowie die Eigenschaften von Algorithmen!

3. Frage
Nennen Sie fünf Grundelemente der Programmierung!

4. Frage
Welche Möglichkeiten stehen Ihnen zur Verfügung, um Algorithmen grafisch darzustellen?

5. Frage
Was ist ein wesentlicher Nachteil der Programmablaufpläne?

6. Frage
Beschreiben Sie die Darstellungsform „Pseudocode“!

1.2. Übungen

1. Übung
Beschreiben Sie den euklidschen Algorithmus zur Berechnung des größten gemeinsamen Teilers als Struktogramm!

2. Übung
Beschreiben Sie einen Algorithmus zur Berechnung der Fakultät! Nutzen Sie dazu den Pseudocode.

3. Übung
Beschreiben Sie im Pseudocode einen Algorithmus zur Bestimmung der Fibonacci-Folge 1 – 1 – 2 – 3 – 5 – 8 – 13 – 21!

4. Übung
Beschreiben Sie in einem Struktogramm das Sieb des Eratosthenes zur Bestimmung von Primzahlen.

2. Grundlagen von Java

2.1. Kontrollfragen

1. Frage
Beschreiben Sie die wesentlichen Eigenschaften von Java!

2. Frage
Beschreiben Sie die Funktionsweise eines Compilers!

3. Frage
Welche Aufgaben übernimmt der Linker?

4. Frage
Nennen Sie unterschiedliche Interpreter-Arten!

5. Frage
Welche Rolle spielen Compiler und Interpreter im Umfeld der Programmiersprache Java?

6. Frage
Beschreiben Sie den Prozess von der Erstellung des Quellcodes bis zur Ausführung des Programms in der Programmiersprache Java!

7. Frage
Nennen Sie drei wesentliche Java-Tools und beschreiben Sie kurz deren Aufgaben!

8. Frage
Beschreiben Sie das Paketkonzept der Programmiersprache Java!

9. Frage
Welche wesentlichen Systemvariablen kennen Sie im Java-Umfeld?

10. Frage
Beschreiben Sie die wesentlichen Unterschiede der verschiedenen Editionen im Rahmen der Java 2™ Platform!

11. Frage
Wozu werden die einzelnen Systemvariablen benötigt?

3. Datentypen

3.1. Kontrollfragen

1. Frage
Welche unterschiedlichen Datentypen kennen Sie in Java?

2. Frage
Wie lassen sich diese Datentypen klassifizieren?

3. Frage
Nennen Sie die numerischen Datentypen der Programmiersprache Java! Worin liegt der wesentliche Unterschied im Wertebereich?

4. Frage
Wie können Variablen und Konstanten in Java deklariert initialisiert werden?

5. Frage
Was ist der Unterschied zwischen einer Konstanten und einer Variablen?

6. Frage
Erläutern Sie den Begriff „Literal“ am Beispiel der numerischen Literale!

7. Frage
Was sind Escape-Sequenzen?

8. Frage
Beschreiben Sie die Konvertierungsregeln in Java bzgl. Erweiternder und einschränkender Konvertierungen für einfache Datentypen!

9. Frage
Was ist ein Array und wie ist dieses aufgebaut?

10. Frage
Welche Aufgabe erfüllt das Attribut length bei einem Array?

11. Frage
Worin unterscheidet sich die Deklaration eines Arrays von der Deklaration einer einfachen Variablen?

12. Frage
Was sind Referenzdatentypen?

4. Ausdrücke und Anweisungen

4.1. Kontrollfragen

1. Frage
Was verstehen Sie im Zusammenhang mit Programmierung unter dem Begriff „Ausdruck?

2. Frage
Woraus setzen sich Ausdrücke zusammen?

3. Frage
Zählen Sie unterschiedliche Operatoren auf! Unterscheiden Sie dabei die Operatoren nach dem Typ der Operanden!

4. Frage
Erläutern Sie den Fragezeichenoperator anhand des folgenden Codings!

```
String X = (a == b) ? "Ja" : "Nein";
```

5. Frage
Definieren Sie den Begriff „Anweisung“ im Sinne der Programmierung!

6. Frage
Welche Möglichkeiten bietet Ihnen die Programmiersprache Java, um Verzweigungen zu realisieren?

7. Frage
Erläutern Sie den Begriff „dangling else“!

8. Frage
Welche Bedeutung messen Sie der Break-Anweisung im Zusammenhang mit der Switch-Anweisung bei?

9. Frage
Nennen Sie die unterschiedlichen Schleifenarten in Java und beschreiben Sie deren Verhalten!

10. Frage
Worin besteht der wesentliche Unterschied zwischen einer kopf- und einer fußgesteuerten Schleife?

11. Frage
Beschreiben Sie den Schleifenkopf einer For-Schleife!

12. Frage
Was bewirken die Break- und die Continue-Anweisung innerhalb einer Schleife?

4.2. Übungen

1. Übung

Implementieren Sie Ihr Struktogramm aus Übung 1 im Kapitel 1.2. zur Berechnung des größten gemeinsamen Teilers nach Euklid!

2. Übung

Implementieren Sie Ihre Pseudocode-Lösung aus Übung 2 im Kapitel 1.2. zur Berechnung der Fakultät einer Zahl!

3. Übung

Implementieren Sie Ihr Struktogramm aus Übung 4 im Kapitel 1.2. zur Bestimmung der Primzahlen nach dem Sieb des Eratosthenes, wobei der Anwender eine beliebige Zahl eingeben soll. Verwenden Sie dazu folgende Eingabemöglichkeit:

```
import javax.swing.JOptionPane;
...
String s = JOptionPane.showInputDialog("Geben Sie eine Zahl ein:");
int zahl = Integer.parseInt(s);
```

4. Übung

Formulieren Sie eine Anweisung, die zur folgenden semantisch äquivalent ist

```
return a==b ? false : true;
```

==> return !(a==b)

ohne Verwendung des Wortes “if” und unter Verwendung logischer Operatoren.

5. Übung

Gegeben sind folgende Variablen mit ihren Werten:

```
int a = 10;
int b = 5;
boolean z = false;
```

Füllen Sie folgende Wahrheitstabelle aus:

Ausdruck	Wahrheitswert
!z	true
a < 20	true
a == 2*b	true
a%b != 0	false
(a > b) && z	false
(a > b) || z	true
!(a < b) ^ !z	false
(a < b) || ((a%3 < b) && !z)	true

6. Übung
Folgende Ausgabe soll erzeugt werden: 1 3 5 7 9
Lösen Sie diese Aufgabe mithilfe einer
   a) While-Schleife
   b) Do-while-Schleife
   c) For-Schleife und des Modulo-Operators
   d) For-Schleife ohne den Modulo-Operator

Lösen Sie die Teilaufgaben zunächst mit einem Struktogramm, bevor Sie mit der Implementierung beginnen.

7. Übung
Folgende Ausgabe soll erzeugt werden: 1 2 4 7 11 16 22 29 37
Lösen Sie diese Aufgabe mithilfe einer
   a) Do-while-Schleife
   b) For-Schleife

8. Übung
Entwerfen Sie eine For-Schleife, die die Buchstaben A bis Z auf dem Bildschirm ausgibt.

9. Übung
Welche mathematischen Operationen werden durch die Methoden meth1 und meth2 beschrieben und welche Werte haben zahl1 und zahl2 am Ende der Berechnung?

```
public class TestAlgorithmen {

     public static void main(String[] args) {
          int zahl1 = meth1(-5);
          int zahl2 = meth2(2, 6);

          System.out.println("Zahl 1:\t" + zahl1 +
                             "\nZahl 2:\t" + zahl2);
     }

     static int meth1( int a ) {    ==> bildet das Quadrat von a (a * a)
          if( a<0 ) a=-a;
          int rc=0;
          for( int i=0; i<a; i++ )
               rc += a;
          return rc;
          }

     static int meth2( int a, int b ) { ==> potenzieren von a mit exponent b (a hoch b)
//   Wenn b<0 wird eine Fehlermeldung erzeugt -> 2. Semester
//   Die Fehlermeldung führt zum Abbruch der Methode
          if( b<0 )
               throw new RuntimeException( "unerlaubter Wert"
          );
          if( b==0 ) return 1;
          int rc=a;
          for( int i=1; i<b; i++ )
               rc *= a;
```

Und deswegen wollen wir sprechende Namen für Methoden und Variablen benutzen :-)

```
            return rc;
            }
}
```

10. Übung
Setzen Sie Ihre Lösung zur Bestimmung der Fibonacci-Folge aus Übung 3 im Kapitel 1.2 in ein Programm um, wobei die Zahlenfolge dauerhaft in einem Array gespeichert werden soll!

11. Übung
Beschreiben Sie einen iterativen Algorithmus zur Berechnung der Wurzel einer Zahl nach dem Heron-Verfahren (babylonisches Wurzelziehen) als Struktogramm! Dabei gilt folgende Berechnungsvorschrift:
a = Zahl, deren Quadratwurzel bestimmt werden soll
x_n = Ergebnis des letzten Iterationsschrittes, wobei x_0 dem Wert a entspricht

$$x_{n+1} = \frac{x_n + \frac{a}{x_n}}{2}$$

Die Berechnung soll abbrechen, sobald die Differenz zwischen x_{n+1} und x_n weniger als ± 0,000001 beträgt.

Beispiel (http://de.wikipedia.org/wiki/Heron-Verfahren, Stand: 08.01.2009)

$a = 9$ und $x_0 = 9$

$$x_1 = \frac{9 + \frac{9}{9}}{2} = \frac{10}{2} = 5$$

$$x_2 = \frac{5 + \frac{9}{5}}{2} = \frac{\frac{34}{5}}{2} = \frac{34}{10} = 3{,}4$$

$$x_3 = \frac{\frac{34}{10} + \frac{9}{\frac{34}{10}}}{2} = \frac{\frac{34}{10} + \frac{90}{34}}{2} = \frac{257}{85} \approx 3{,}0235$$

$$x_4 = \frac{\frac{257}{85} + \frac{9}{\frac{257}{85}}}{2} = \frac{\frac{257}{85} + \frac{765}{257}}{2} = \frac{65537}{21845} \approx 3{,}00009.$$

Implementieren Sie Ihre Lösung.

12. Übung
Beschreiben Sie in einem Struktogramm einen Algorithmus zur Berechnung des Pascal'schen Dreiecks!

```
                1
              1   1
            1   2   1
          1   3   3   1
        1   4   6   4   1
      1   5   10  10  5   1
    1   6   15  20  15  6   1
              usw.
```

Implementieren Sie Ihre Lösung!

5. Objektorientierung

5.1. Kontrollfragen

1. Frage
Grenzen Sie die Begriffe Klasse und Objekt voneinander ab!

2. Frage
Was beschreiben Attribute und Methoden?

3. Frage
Wie werden Klassen in der UML dargestellt?

4. Frage
Wie werden in Java Objekte erzeugt? Welche Rolle spielen dabei die Konstruktoren einer Klasse?

5. Frage
Erläutern Sie den Begriff des Konstruktors!

6. Frage
Beschreiben Sie das Konzept der Kapselung in der objektorientierten Programmierung!

7. Frage
Welche unterschiedlichen Sichtbarkeitsmodifier sind Ihnen bekannt? Beschreiben Sie die Sichtbarkeit jedes Modifiers!

8. Frage
Beschreiben Sie den Zugriff auf Instanzattribute und –methoden! Welche Rolle spielen dabei die Sichtbarkeitsmodifier?

9. Frage
Unterscheiden Sie die Begriffe „call by value“ und „call by reference“!

10. Frage
Was versteht man unter dem Begriff „überladen von Methoden“?

11. Frage
Worin unterscheiden sich Klassenattribute und –methoden von Instanzattributen und –methoden?

12. Frage
Wie können Klassenmethoden auf den Instanzenkontext zugreifen? Beschreiben Sie zwei Lösungsansätze!

13. Frage
Erläutern Sie das Prinzip der Garbage Collection in Java!

14. Frage
Was sind Destruktoren und wie werden Sie aufgerufen?

15. Frage
Nennen Sie unterschiedliche Klassenarten in Java!

16. Frage
Erläutern Sie die Begriffe Assoziation, Aggregation und Komposition! Wie werden die unterschiedlichen Beziehungen in der UML dargestellt?

5.2. Übungen

1. Übung
Gegeben ist folgendes UML-Diagramm einer Klasse Haus:

Haus
- tueren: int - fenster: int - etagen: int - flaeche: double - strasse: String - hausnummer: String - plz: int - ort: String - flachdach: boolean
+ Haus() + Getter-/Setter-Methoden

Bei den Setter-Methoden soll die Plausibilität der übergebenen Werte überprüft werden (z.B. keine leeren Strings, keine negativen Postleitzahlen, keine negative Anzahl an Türen, Fenstern oder Etagen, eine minimale Wohnfläche größer 100 und kleiner 500, usw.).
Schreiben Sie zusätzlich eine Testklasse HausTest, in der Sie 3 Hausobjekte erzeugen und in einem Array speichern. Geben Sie alle Hauseigenschaften der im Array gespeicherten Hausobjekte innerhalb einer For-Schleife aus.

2. Übung
Erweitern Sie die Klasse Haus um weitere Konstruktoren. Es soll zu jedem Attribut ein eigener Konstruktor zur Verfügung gestellt werden. Außerdem soll ein Konstruktor zur Verfügung stehen, der alle Attribute entgegen nimmt.

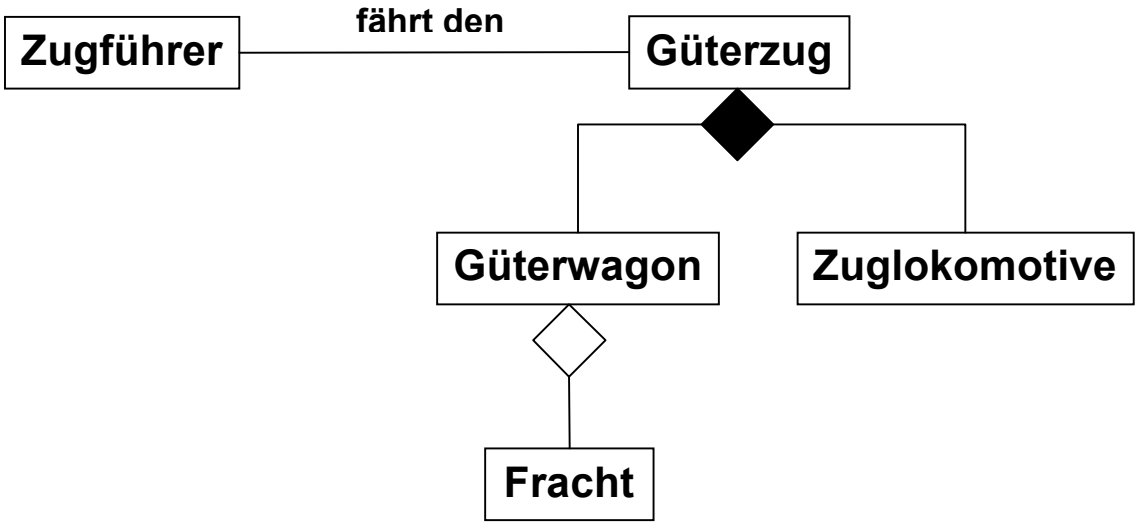
3. Übung
Erweitern Sie die Klasse Haus aus Übung 1 um ein statisches Attribut objCnt vom Typ Integer mit dem Initialwert 0 und die statische Methode getObjCnt() vom Typ Integer. Ändern Sie Ihren Konstruktor so ab, dass objCnt um 1 erhöht wird, sobald ein neues Objekt erzeugt wird.
Geben Sie in der Klasse HausTest nach der Ausgabe der Eigenschaften der Hausobjekte die Anzahl der erzeugten Objekte aus.

4. Übung
Erweitern Sie Ihre Klasse Haus um die finalize()-Methode. Dekrementieren Sie objCnt um 1, wenn ein Objekt gelöscht wird.
Löschen Sie zum Test am Ende Ihrer Klasse HausTest ein Hausobjekt und geben Sie die Anzahl der vorhandenen Hausobjekte erneut aus.

5. Übung

Gegeben ist das folgende Diagramm, dass die Assoziationen, Aggregationen und Kompositionen zwischen den Klassen abbildet.

a) Nennen Sie die Klassen und die Beziehungsarten, die zwischen den Klassen vorherrschen.
b) Wie können die Beziehungen sinnvoll realisiert werden? Erweitern Sie die nachstehenden Klassendiagramme um entsprechende Attribute oder Methoden!
c) Implementieren Sie die einzelnen Klassen und berücksichtigen Sie dabei die Beziehungen.
d) Implementieren Sie eine Testklasse, um einen Güterzug zu erstellen, ihm einen Fahrer zuzuordnen und die Güterwagons mit Fracht zu beladen. Geben Sie anschließend über die jeweiligen Getter-Methoden die Informationen zum Fahrer, dem Güterzug und der geladenen Fracht aus.



Zugführer
- zug : Güterzug - name: String - vorname: String - id: int - gebDat: Date
+ Zugführer(name: string, vorname: String, id: int, gebDat: Date) + Getter-/Setter-Methoden

Güterzug
- fahrer : Zugführer - lok : Zuglokomotive - wagons : Güterwagons[] - nummer: int - laenge: double - anzWagons: int
+ Güterzug(log: Zuglokomotive, wagons: Güterwagon[], nummer: int, anzWagons: int) + Güterzug(nummer: int, anzWagons: int) + Getter-/Setter-Methoden

Zuglokomotive
- güterzug : Güterzug - nummer: int - typ: String - antrieb: String - achsen: byte
+ Zuglokomotive(nummer: int, typ:string, antrieb: String, achsen: byte) + Getter-/Setter-Methoden

Güterwagon
- güterzug : Güterzug - inhalt : Fracht[] - maxLast: double - typ: String - achsen: byte
+ Güterwagon(maxLast: double, typ:string, achsen: byte) + Getter-/Setter-Methoden

Fracht
- wagon : Güterwagon - bezeichnung: String - menge: double - verpackung: String
+ Fracht(bezeichnung: String, menge: double, verpackung: String) + Getter-/Setter-Methoden

6. Vererbung

6.1. Kontrollfragen

1. Frage
Beschreiben Sie das objektorientierte Konzept der Vererbung!

2. Frage
Erläutern Sie die Begriffe Superklasse, Subklasse und Vererbungshierarchie!

3. Frage
Beschreiben Sie die Darstellung der Vererbung in der UML mithilfe eines kleinen Beispiels!

4. Frage
Was ist die Besonderheit in der Vererbung bei Java und wie ist die Vererbung syntaktisch in Java realisiert?

5. Frage
Beschreiben Sie die wesentlichen Eigenschaften der Klasse Object und ihre besondere Rolle in der Vererbungshierarchie in Java?

6. Frage
Beschreiben Sie das Konzept des Überschreibens von Methoden!

7. Frage
Was bewirken die Modifier abstract und final in Bezug auf Klassen?

8. Frage
Welche Auswirkungen haben die Modifier abstract und final bei Methoden?

9. Frage
Wozu wird der Ausdruck super in der Vererbung benötigt?

10. Frage
Was stellt der Ausdruck this dar?

11. Frage
Beschreiben Sie das Konzept des narrowing und widening Cast!

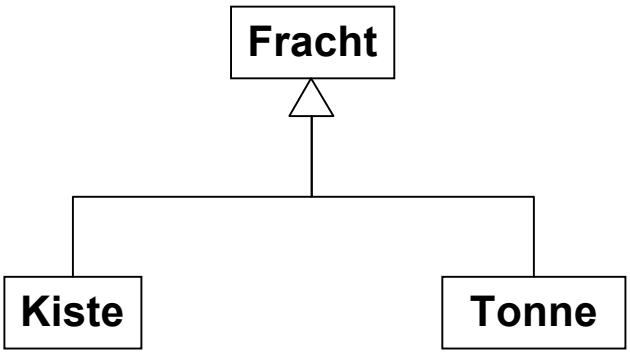
12. Frage
Beschreiben Sie das Konzept des Polymorphismus in der objektorientierten Programmierung anhand eines einfachen Beispiels!

13. Frage
Welche Möglichkeiten haben Sie, um in Java Objekte zu kopieren? Worin besteht der Unterschied in den beiden Verfahren?

6.2. Übungen

1. Übung

Das nachstehende Vererbungsdiagramm erweitert das Modell aus der Übung 5 aus dem Übungsteil 5.2.



Fracht
bezeichnung: String # volumen: double # menge: double # verpackung: String
+ Fracht(bezeichnung: String, menge: double, verpackung: String) + Getter-/Setter-Methoden

Kiste
- breite: int - hoehe: int - tiefe: int
+ Kiste(bezeichnung: String, menge: double, breite: int, hoehe: int, tiefe: int) + Getter-/Setter-Methoden

Tonne
- hoehe: int - radius: int
+ Tonne(bezeichnung: String, menge: double, hoehe: int, radius: int) + Getter-/Setter-Methoden

a) Erweitern Sie die Klasse Fracht um das Attribut Volumen und die entsprechende Getter-Methode für dieses Attribut.

b) Implementieren Sie anschließend die beiden Subklassen Kiste und Tonne. Beim Aufruf der Konstruktoren der Sub-Klassen soll zunächst der Konstruktor der Super-Klasse gerufen werden, um die allgemeinen Attribute zu setzen. Die Bezeichnung der Verpackung soll dabei als konstanter Wert „Kiste“ oder „Tonne“ gesetzt werden. Berechnen Sie innerhalb des Konstruktors das Volumen der Fracht.

c) In den Subklassen sollen lediglich die Getter- und Setter-Methoden für die subklassenspezifischen Attribute zur Verfügung gestellt werden.
d) Erweitern Sie Ihre Testklasse aus der Übung 5. aus dem Übungsteil 5.2., indem Sie unterschiedliche Objekte der Klassen Kiste und Tonne erzeugen und als Fracht in die Güterwagons laden. Nutzen Sie dazu das Konzept des Polymorphismus. Geben Sie sich die Informationen zu Ihren Frachtobjekten über eine polymorphe Implementierung aus.

7. Interfaces

7.1. Kontrollfragen

1. Frage

7.2. Übungen

1. Übung
Neue Zahlenfolge berechnen:
1 4 2 5 3 6 4 7 5

2. Übung
Noch eine Zahlenfolge
5 10 15 20 17 22 27 32 29 34 39 44 41 46 51 56 53

3. Übung
Komposition: Auto (?) besteht aus Motor (PS, Hubraum, Zylinder), Reifen (Breite, Durchmesser, Hersteller, Sommer) und Karosserie (Farbe, Metallic, Material)
Wie muss der Konstruktor bei Kompositionen in der Kompositionsklasse aussehen?
Implementierung der Komposition

4. Übung
Vererbung:
Fahrzeug (Antrieb, Plätze, abstrakte Methode fahren) vererbt an Auto (PS, Farbe, Marke, Geschwindigkeit) und Schiff (Typ, Geschwindigkeit); Vererbung implementieren, Testklasse mit einem Fahrzeug-Array mit 4 Elementen -> Speichern von je 2 Schiffen und 2 Autos; Polymorpher Aufruf der Methode fahren