

**Lernziele**

- Kontrollstrukturen

**Aufgabe 1) Kontrollstrukturen 101**

Der Aufruf

```
double d = Math.random();
```

erzeugt eine Zufallszahl im Intervall [0..1[ (ohne die 1!). Somit kann man mit dem Ausdruck

```
double d = (Math.random() * (upper-lower+1)) + lower;
```

Zufallszahlen zwischen einer Obergrenze (upper) und einer Untergrenze (lower) erzeugen (jeweils inklusive). Für die Untergrenze 1 vereinfacht sich dieser Ausdruck zu

```
double d = Math.random() * upper + 1;
```

Weist man eine solche Zufallszahl einer Ganzzahl zu, kann man Lottozahlen erzeugen

```
int lottoZahl = (int) (Math.random() * 49) + 1;
```

**Teilaufgabe a) Sequenz**

Schreiben Sie ein Programm, das mit Hilfe einer Sequenz folgende Ausgabe erzeugt

```
eins  
zwei  
drei
```

**Teilaufgabe b) Schleifen(1)**

Schreiben Sie ein Programm, das mit Hilfe einer for-Schleife folgende Ausgabe erzeugt

```
1 2 3 4 5 6 7 8 9
```

Wandeln Sie das Programm anschließend um, indem Sie die for-Schleife durch eine while-Schleife ersetzen.

**Teilaufgabe c) Schleifen(2)**

Schreiben Sie ein Programm, das mit Hilfe einer do-Schleife folgende Ausgabe erzeugt

```
1  
2  
3  
4  
5
```

**Teilaufgabe d) Schleifen(3)**

Schreiben Sie ein Programm, das mit Hilfe von 2 Schleifen folgende Ausgabe erzeugt

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

**Teilaufgabe e) Entweder-Oder Entscheidung**

Schreiben Sie ein Programm, das eine Zufallszahl zwischen 1 und 10 erzeugt und anschließend mit Hilfe einer If-Anweisung eine Ausgabe der Form

4 ist eine gerade Zahl

oder

7 ist eine ungerade Zahl

erzeugt. Hinweis: Gerade Zahlen sind ohne Rest durch 2 teilbar...

**Teilaufgabe f) Fallunterscheidung (switch)**

Schreiben Sie ein Programm, das eine Zufallszahl zwischen 1 und 3 erzeugt und mit Hilfe einer switch-Anweisung abhängig von der Zahl folgende Ausgaben erzeugt

```
bei 1: ja
bei 2: nein
bei 3: vielleicht
```

Verändern Sie anschließend Ihr Programm, indem Sie die switch-Anweisung durch eine Kette von if-else-if-Anweisungen ersetzen.

**Aufgabe 2) Kontrollstrukturen 201**

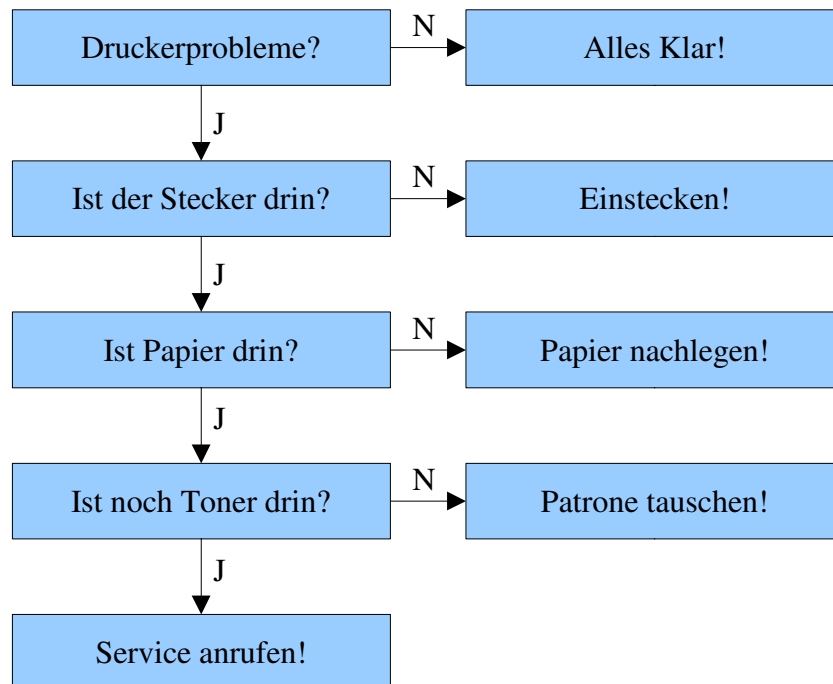
Bearbeiten Sie die folgenden Teilaufgaben um die verschiedenen Kontrollstrukturen und deren Vor- und Nachteile bzw. spezielle Eigenheiten kennenzulernen.

Hinweis: Für die Ein-/Ausgaben bei dieser Aufgabe soll wieder die Klasse CommandLine verwendet werden. Dazu muss sich die Datei CommandLine.java im selben Verzeichnis wie Ihr Programm befinden (Details zum Thema I/O in Java folgen später).

**Teilaufgabe a) Entscheidung (if-else)**

Versuchen Sie, das folgende Ablaufschema durch verschachtelte if-else Anweisungen zu implementieren. Dazu geben Sie zunächst die entsprechende Frage aus und verzweigen dann entsprechend der eingegebenen Antwort wie im Codebeispiel vorgegeben.

Beachten Sie, daß der Stringvergleich mit Hilfe der Methode `equals()` durchgeführt werden muss, da der Operator `==` bei Objekten nicht auf Gleichheit sondern auf Identität prüft (zum Unterschied zwischen Gleichheit und Identität später mehr).



Beachten Sie, daß das Ablaufdiagramm nicht der Norm entspricht. Es handelt sich vielmehr um die Art von informellem Diagramm, die man im Rahmen der Anforderungsanalyse vom Kunden bekommen könnte.

Zusatzaufgaben für zuhause:

- Zeichnen Sie dieses Diagramm als Flußdiagramm.
- Zeichnen Sie dieses Diagramm als Nassi-Shneidermann Diagramm.
- Erweitern Sie das Diagramm um weitere Schritte, z.B. Rückfrage/Wiederholung nach Durchführung einer Korrekturmaßnahme oder Zählung der Reparaturversuche.

### Teilaufgabe b) Fallunterscheidung

Implementieren Sie die folgende Fallunterscheidung nach Eingabe einer Zahl als verschachteltes if-else sowie mit Hilfe einer switch-Anweisung.

Ausgabe:

"null" falls number = 0  
 "negativ" falls number = -1  
 "positiv" falls number = +1

Welche Variante ist eleganter? Welche würden Sie vorziehen, um viele Fälle zu prüfen? Wie lauten die Einschränkungen für diese Variante? Was bedeutet dies, wenn Sie statt

auf 0 / -1 / +1 auf 0 / <0 / >0 testen wollen? Testen Sie Ihr Programm dazu mit den Werten -2, -1, 0, 1, 2 (Was passiert bei Eingabe von +1? Warum?).

---

### Teilaufgabe c) Schleifen

In Java gibt es drei Arten von Schleifen:

- while () { }
- do {} while ()
- for (;) { }

Diese sollen in dieser Teilaufgabe benutzt werden, um verschiedene Varianten eines Countdown Zählers zu implementieren. Dazu soll jeweils ein separate Methode erstellt werden, die aus `main()` mit dem Startwert aufgerufen wird (vgl. Beispieldatei). Die Ausgabe der Funktion soll folgendermaßen aussehen (mit Startwert 3)

3

2

1

Kuckuck!

Testen Sie ihre Implementierung mit den Startwerten 3,1,0 und -1. Machen Sie sich anhand des Resultats nochmals den Unterschied zwischen kopf- und fußgesteuerten Schleifen klar. Dieser besteht darin, daß

---

### Aufgabe 3) Zusatzaufgaben

#### Teilaufgabe a) Lottozahlen

Schreiben Sie ein Programm, das 6 Lottozahlen erzeugt, in einem Array speichert und anschließend ausgibt. Verwenden Sie 2 verschiedene Methoden für Erzeugung und Ausgabe. Die `main`-Methode soll also z.B. so aussehen:

```
public static void main(String[] args) {
    int[] lottozahlen = lottozahlenErzeugen();
    lottozahlenAusgeben( lottozahlen);
}
```

#### Teilaufgabe b) Statistik

Schreiben Sie ein Programm, das 10000 Zufallszahlen zwischen 0 und 9 erzeugt. Dabei soll in einem Array gezählt werden, wie oft jede Zahl vorkommt. Geben Sie am Programmende die relative Häufigkeit für jede Zahl aus (Zählerstand / Anzahl Durchläufe).

**Teilaufgabe c) Zahlenraten**

Implementieren Sie ein Zahlenratespiel. Dabei soll der Benutzer sich eine Zahl zwischen 0 und 100 ausdenken. Anschließend soll das Programm diese Zahl raten und den Benutzer fragen, ob seine Zahl größer, kleiner oder gleich der geratenen Zahl ist (Eingabe ">", "<" oder "="), bis es die Zahl gefunden hat.

Offenbar braucht man hierfür eine Schleife für die vielen Versuche und eine Fallunterscheidung für die Auswertung der Benutzereingabe. Erstellen Sie zunächst ein Flußdiagramm, Nassi-Shneidermann Diagramm oder Pseudocode für ihren Algorithmus.

Welche Ratestrategie ist wohl am schnellsten (kommt mit den wenigsten Versuchen aus)? Bedenken Sie dazu, wie groß der Suchraum ist (wieviele Zahlen er enthält) und überlegen Sie sich, wie Sie ihn am schnellsten verkleinern (d.h. Zahlen ausschließen können). Hinweis: Jede Frage liefert den Informationsgehalt von einem Bit...

---