


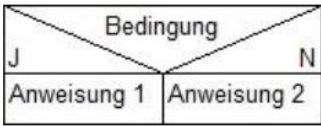
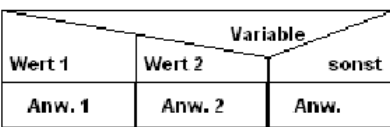
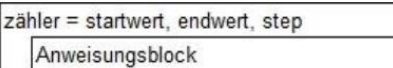
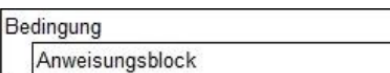
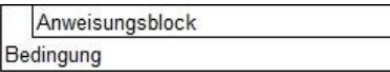
## Java ist ...

... (genau wie Pascal / Delphi) eine imperative Programmiersprache, basiert auf den bekannten Algorithmenstrukturen und besitzt daher einen ähnlichen Aufbau wie Delphi – Java-Programmierer dürften nach dieser Aussage jedoch kurz vor einem Nervenzusammenbruch stehen ☺. RECHT HABEN SIE. Pascal wurde (schon) Ende der 60er Jahre von Niklaus Wirth als „Unterrichtssprache“ entwickelt, da sich das damals vorherrschende ALGOL nicht wirklich dafür eignete. Java ist aus dieser Sicht eher jung, wurde (erst) am 23. Mai 1995 von der Firma SUN vorgestellt und versuchte die Vorteile verschiedener Sprachen wie C++, Ada, Smalltalk und Scheme zu vereinen – einer der Urgroßväter von Java war auch Pascal. Java entstand also in einer Zeit, als Computer anfangen, bezahlbar zu werden und das Internet durch das WWW seinen Durchbruch begann – wie auch z. B. die Sprachen JavaScript, Ruby und PHP (alle 1995!!!). ...

## Syntaktische Vereinbarungen

Aus den Kursen der E-Phase sollten grundlegende syntaktische Vereinbarungen bekannt sein ...

### Algorithmenstrukturen

Anweisungsblock (Sequenz)		<pre>{   Anweisung1;   Anweisung2;   Anweisung3; }</pre>
Fallunterscheidung (Verzweigung, Alternative)		<pre>if (Bedingung) {   Anweisung1; } else {   Anweisung2; }</pre>
Mehrfachauswahl		<pre>switch (Variable) {   case W1 :     Anweisung1;     break;   default :     Anweisung; }</pre>
Zählschleife		<pre>for (int i=s; i&lt;=e; i++) {   Anweisung; }</pre>
anfangsgeprüfte Schleife		<pre>while (Bedingung) {   Anweisung; }</pre>
endgeprüfte Schleife		<pre>do {   Anweisung; } while (Bedingung)</pre>

## Liste der Vergleichsoperatoren

Operator	Erläuterung	Beispiel
==	a == 3 Test, ob a den Wert 3 hat	<pre>int a = 5;  a == 4;      =&gt; false a == 5;      =&gt; true a == 15 / 3; =&gt; true 5 == a;      =&gt; true</pre>
!=	a != 3 Test, ob a NICHT den Wert 3 hat	<pre>int a = 5;  a != 4;      =&gt; true a != 5;      =&gt; false a != 15 / 3; =&gt; false 5 != a;      =&gt; false</pre>
<, >, <=, >=	kleiner als, größer als kleiner bzw. größer gleich	<pre>int a = 5;  a &lt; 4;       =&gt; false a &lt;= 5;      =&gt; true a &gt; 15 / 3;  =&gt; false 15 &lt; a;      =&gt; false</pre>

## Liste wichtiger logischer Operatoren

Operator	Erläuterung	Beispiel
&&	e = a && b e ist true, wenn a und b true sind, ansonsten ist e false (UND).	<pre>int a = 5; int b = 3;  (a &gt; 4) &amp;&amp; (b == 3)    =&gt; true (a &lt;= 5) &amp;&amp; (b != 3)   =&gt; false</pre>
	e = a    b e ist true, wenn a oder b true sind, ansonsten ist e false (OR).	<pre>int a = 5; int b = 3;  (a &gt; 4)    (b == 3)    =&gt; true (a &lt;= 5)    (b != 3)   =&gt; true</pre>
^	e = a ^ b e ist true, wenn entweder a oder b true ist, aber nicht beide gemeinsam (XOR).	<pre>int a = 5; int b = 3;  (a &gt; 4) ^ (b == 3)     =&gt; false (a &lt;= 5) ^ (b != 3)    =&gt; true</pre>
!	e = !a e ist true, wenn a nicht true ist (NOT).	<pre>int a = 5;  !(a &gt; 4)               =&gt; false !(a &lt; 5)               =&gt; true</pre>

## Wichtige Datentypen

byte, int, double (float)

char, String (Objekt)

[]

## Styleguide

---

Seit längerer Zeit gibt es Programmierrichtlinien, die genormte Quelltexte hervorbringen sollen, so dass diese auch von anderen Entwicklern nachvollzogen werden können. Unter „Programmierrichtlinien“ fällt nun Verschiedenes: Namensgebung von Variablen, Methoden und Klassen, Einrückung des Programmcodes, Dokumentation und mehr.

Der *Bezeichner* dient zur Identifizierung einer Variablen (somit auch einer Konstante), Funktion oder Klasse und muss innerhalb eines Geltungsbereichs eindeutig sein. In verschiedenen Geltungsbereichen können dagegen die gleichen Bezeichner verwendet werden:

- Namen dürfen nur aus bestimmten Zeichen gebildet werden. Buchstaben und Ziffern sind erlaubt, und auf das Unterstreichungszeichen »\_« sollte verzichtet werden. Wenn möglich, sollten aber Zeichenketten in Großbuchstaben aufgelöst werden. So sollte `MyABCEXAM` zu `MyAbcExam` werden, die Alternative `MyABC_Exam` ist nicht so gut.
- Setzt sich ein Bezeichnername aus mehreren Wörtern zusammen, so werden diese zusammengeschrieben und die jeweiligen Anfangsbuchstaben groß gesetzt. (`kaugummiGenerator`, `anzahlKaugummis`).
- Von Sonderzeichen (ä, ö, ü, ß, \$, ...) sollte Abstand genommen werden, auch wenn Java diese Zeichen zulässt, da sie im Unicode-Zeichensatz definiert sind.
- Es dürfen keine Bezeichner gewählt werden, die sich nur in der Groß- beziehungsweise Kleinschreibweise unterscheiden. Java achtet auf Groß- und Kleinschreibung, so dass jedoch `xyz` und `XYZ` für eine Variable und eine Konstante verwendet werden können. Wir sollten also nicht die Methode `makeMyDay()` und irgendwo die Boolean-Konstante `MakeMyDay` einsetzen, da dies zu Verwirrungen führt, obwohl es formal zulässig wäre.
- Unaussprechbare Namen dürfen nicht vergeben werden. Ein aussagekräftiger langer Name ist besser als ein kurzer, dessen Aufbau sich nicht erkennen lässt. So ist `resetPrinter` ein besserer Name als `rstprt`.
- Namen dürfen keine Abkürzungen enthalten, die nicht allgemein anerkannt sind. So ist der Bezeichner `groupID` sicherlich besser als `gopID`. Bei Abkürzungen ist darauf zu achten, dass diese nicht missverstanden werden können. Heißt `termProcess()` vielleicht »terminiere den Prozess«, oder besteht die Verbindung zu einem »Terminal-Prozess«?
- Bezeichner von Variablen und Methoden beginnen mit kleinen Buchstaben (`int myAbc`; `float downTheRiverside`; `boolean haveDreiFragezeichen`).
- Bezeichner von Klassen beginnen mit großen Buchstaben (`class OneTwoTree`).
- Bezeichner müssen »sprechend«, also selbsterklärend gewählt werden.
- Ausnahme bilden Schleifenzähler. Diese werden oft einbuchstabig gewählt, wie `i`, `k`.
- Konstanten schreiben wir vollständig groß mit Unterstrichen an den Wortgrenzen. Damit vermeiden wir von vornherein das Problem, auf eine Konstante versehentlich schreibend zugreifen zu wollen (Beispiel: `_MEHRWERTSTEUER_`).

Speziell für Java und objektorientierte Sprachen gibt es noch ein paar weitere Faustregeln:

- Methodennamen beginnen mit Verben, Klassennamen sind Substantive und Schnittstellennamen bevorzugt Adjektive.
- Es sollte zudem keine Funktion geben, die den Namen einer Klasse trägt.

## Übungsaufgaben

### Aufgabe 1

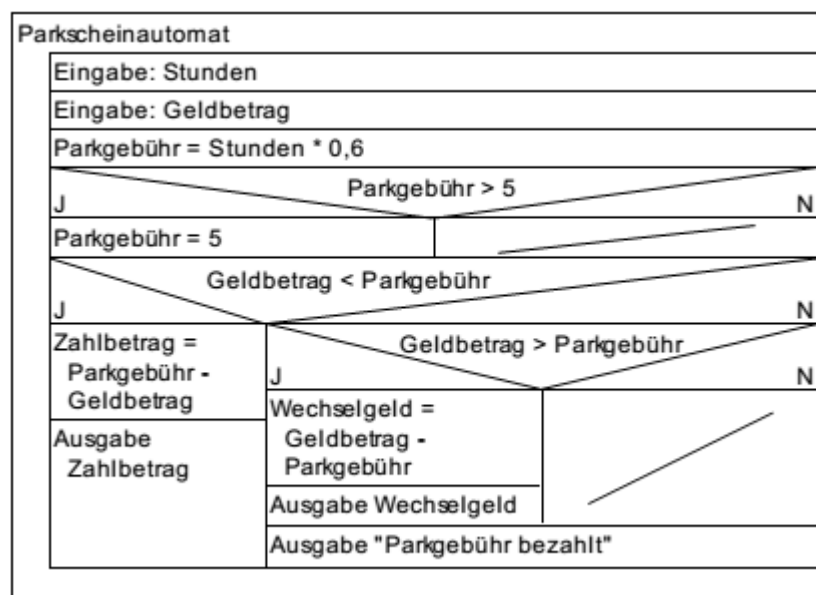
Der Tarif „Glücklich Simsen“ des Anbieters H3PO bietet im Monat 25 Frei-SMS. Für jede weitere SMS fallen Kosten in Höhe von 9,2 Cent an. Geben Sie einen Algorithmus (Struktogramm) an, welcher für eine eingegebene Anzahl an SMS die anfallenden Kosten berechnet.

- Implementieren Sie diesen Algorithmus in Java.
- Implementieren Sie auch ein GUI-Programm.

### Aufgabe 2

Auf einem kostenpflichtigen Parkplatz gilt der Tarif 0,6 € pro angefangener Stunde, maximal 5 € pro Tag. Beahlt wird beim Verlassen des Parkplatzes. Die Funktionsweise des Parkscheinautomaten wird durch nebenstehendes Struktogramm beschrieben.

Simulieren Sie den Parkscheinautomat in einem Java-Programm (Konsole und GUI).



### Aufgabe 3

Ermittle mit Papier und Bleistift, was das folgende Programm ausgibt.

```
public static void main(String[] args) {
    int y;
    int x = 1;
    int total = 0;

    while (x <= 5) {
        y = x * x;
        System.out.println(y); total = total + y;
        x = x + 1;
    }

    System.out.println("Total ist: " + total);
}
```

## Aufgabe 4

```
01 public class wasTuIch01 {
02     public static void main(String[] args) {
03         int z, r;
04
05         z = Input.readInt();
06
07         for (int i = 1; i <= (z / 2); i++) {
08             r = z % i;
09             if (r == 0) {
10                 System.out.println(i);
11             }
12         }
13     }
14 }
```

- Welches Problem bearbeitet der dargestellte Algorithmus?
- Welche Aufgabe erfüllen die vereinbarten (drei) Variablen?
- Welche Kontrollstrukturen enthält dieser Algorithmus?
- Welche Aktion wird in den Zeilen 05 bzw. 10 ausgeführt?
- Welche Zahlen werden für den Startwert 24 beim Durchlauf des Algorithmus (der Reihenfolge nach) ausgegeben?
- Erstellen Sie ein Struktogramm für diesen Algorithmus.

## Aufgabe 5

Ein Programm soll helfen, zu entscheiden, ob eine Person über-, unter- oder normalgewichtig ist. Als Eingabedaten werden Geschlecht, Alter, Gewicht und Größe verlangt.

Als normalgewichtig bezeichnet man Personen, deren Gewicht in Kilogramm gleich der Körpergröße über 100 cm ist. Entsprechend ergeben sich die Unter- bzw. Übergewichtigen.

Als Toleranzgrenzen sind bei Frauen +4% bis -7% und bei Männern +5% bis -5% Abweichung von den Sollwerten zugelassen. Alle Aussagen treffen erst für Personen zu, deren Wachstum beendet ist, d. h. die älter als 17 Jahre und größer als 1,20 m sind. Bei anderen Personen können keine Angaben gemacht werden.

In einem geeigneten GUI-Programm sollen die Daten eingegeben werden und eine entsprechende Antwort ausgegeben werden.

## Aufgabe 6

Ein Programm soll dem Anwender die Möglichkeit geben, eine beliebige vom Anwender einzugebende Anzahl vierstelliger Zufallszahlen...

- ... auszugeben,
- ... deren Mittelwert anzugeben,
- ... deren Minimum und Maximum anzugeben.

Implementiere ein Java-Programm, welches diese Aufgabe bearbeitet.

## Aufgabe 7 (HERON - Verfahren)

Heron von Alexandria (genannt Mechanicus) war ein Mathematiker und Ingenieur, der vermutlich im 1. Jahrhundert lebte. Bekannt sind vor allem seine Ausführungen zu automatischen, teilweise sogar schon programmierbaren Geräten und der Ausnutzung von Wasser, Luft und Hitze als treibende Kraft. Hier sind insbesondere die Erfindung der Aeolipile, auch Heronsball genannt, und der Heronsbrunnen zu nennen.



Außerdem entwickelte er einen Algorithmus, mit dem man für eine beliebige Zahl die Quadratwurzel auf eine beliebige Genauigkeit berechnen kann ... das Heron-Verfahren:

- (1) Wähle einen Startwert  $x$  zur Bestimmung der Wurzel aus  $w$ .
- (2) Wähle  $\left(x + \frac{w}{x}\right) : 2$  als nächsten Näherungswert.
- (3) Wiederhole (2) mit den jeweils neuen Näherungswerten bis die gewünschte Genauigkeit erreicht ist.

Die Grundidee für dieses Verfahren basiert auf der Frage, wie ein Rechteck in ein flächengleiches Quadrat umgewandelt werden kann ... dafür wird die Seitenlänge des Quadrats benötigt.

Implementiere ein Java-GUI-Programm, welches für eine einzugebende Zahl und Genauigkeit die Quadratwurzel berechnet.

## Zusatzaufgabe (Schnittpunkt):

Zwei Geraden  $g$  und  $h$  werden durch die Gleichungen  $g: y = ax + b$  und  $h: y = cx + d$  beschrieben. Schreiben Sie ein Programm, welches mit Hilfe der eingegebenen Koeffizienten  $a$ ,  $b$ ,  $c$  und  $d$  den Schnittpunkt der beiden Geraden berechnen kann. Dabei sollen möglichst alle Lagebeziehungen zweier Geraden zueinander berücksichtigt werden. Erstellen Sie vorher ein Struktogramm.

