Kontrollstrukturen und Schleifen

Du kennst bisher die Kontrollstruktur *if*, die überprüft, ob eine bestimmte Bedingung erfüllt ist und Anweisungen ermöglicht, die in diesem Fall ausgeführt werden sollen. Darüber hinaus lässt sich mit der Ergänzung *else* angeben, was getan werden soll, wenn die Bedingung nicht erfüllt wird. Das sah so aus:

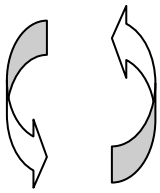
```
if (Bedingung) {
                Anweisung(en)
        }
else {
                Anweisung(en)
        }
```

Die Anweisungen werden nur einmal ausgeführt. Oft kommt es jedoch vor, dass Anweisungen mehrmals ausgeführt werden sollen. Wenn die Anzahl der Durchläufe nicht von vornherein feststeht, kann man sie an die Einhaltung einer Bedingung knüpfen. Der Code müsste sich an dem Muster „Führe eine oder mehrere Anweisungen aus, solange folgende Bedingung gilt...“ In Java sieht das wie folgt aus

```
do {
        Anweisung(en);
}  while (Bedingung);
```

Im Beispiel

Java-Code	**Bedeutung**
`do {`	Führe folgende Anweisung aus:
`  x = x+1;`	Erhöhe x bei jedem Durchlauf um 1
`  System.out.println(x);`	und gib den Wert von x auf der Konsole aus
`} while (x<10);`	Solange x kleiner als 10 ist



Da der Code mehrfach durchlaufen wird, spricht man auch von einer Schleife. Mit der do-while-Schleife werden die Anweisungen mindestens einmal ausgeführt, da erst nach dem ersten Durchlauf geprüft wird, ob eine bestimmte Bedingung gilt. Will man den Durchlauf durch die Schleife von einer Bedingung abhängig machen, beginnt man mit der Bedingung

```
while (Bedingung)
{
        Anweisung(en);
}
```

Hier ist ein do nicht nötig, seine Funktion wird von den Klammern übernommen. Am Beginn der Schleife wird die Erfüllung der Bedingung abgefragt. Wenn sie nicht gilt, wird die Schleife gar nicht ausgeführt. Gilt sie, wird die Schleife so lange durchlaufen, bis die Bedingung nicht mehr erfüllt ist

Im Beispiel

Java-Code	**Bedeutung**
while (x<10)	Solange x kleiner als 10 ist
{	führe folgende Anweisung aus:
x = x+1;	Erhöhe x bei jedem Durchlauf um 1
System.out.println(x);	und gib den Wert von x auf der Konsole aus
}	

Achtung: Bei do-while wird while mit einem Semikolon abgeschlossen, weil die Anweisung beendet ist. Bei while wird kein Semikolon gesetzt, da die Anweisung erst auf die Bedingung folgt.

Aufgabe 1) Welche Ausgabe erwartest du?

```
public class variablenwerte {
        int a;
        int b;
public variablenwerte() {
        a=0;
        b=3;
        }

        public void wasWirdAusgegeben() {
                System.out.print(„Der Wert von a ist: “ + a);
                while (a < 10) {
                System.out.print(a);
                a=a+b;
                 }
}
```

Aufgabe 2) Was würde deiner Meinung nach passieren, wenn a den Startwert 15 hätte?

Aufgabe 3) Was würde deiner Meinung nach passieren, wenn die Anweisung in der Schleife a=a-b; lauten würde?

Aufgabe 4) Dein Freund ist ganz verrückt darauf, Schuhe zu kaufen. Er nimmt sich aber vor, niemals mehr als 49 Paare zu besitzen. Entwirf eine Klasse „Shopping“ mit einer Methode kaufeSchuhe() , die solange Schuhe kauft, bis die Anzahl der eigenen Schuhpaare 49 erreicht hat. Da sich dein Freund über jeden Kauf freut, sollte jeder Kauf auf der Konsole ausgegeben werden. In einem ersten Durchlauf kannst du die Zahl der Paare, die er schon hat, auf 20 festlegen.
Wenn alles funktioniert, wandle das Programm so ab, dass der Nutzer die Startzahl der eigenen Paare angeben kann. Nutze für die Anpassung dein Wissen über Variable (die Abbruchbedingung kann auch Variablen enthalten).