Aufgabe 1.90:
Ein Geschäft gibt ihren Kunden unterschiedliche Rabatte. Schreiben ein Programm, das

- **folgende Rabattklassen vorsieht:**
  - **Klasse 0: kein Rabatt**
  - **Klasse 1: 4,5 % Rabatt**
  - **Klasse 2: 8,75% Rabatt**
- **die Rabattwahl in einer switch-Anweisung durchführt**
- **aus der Rabattklasse und dem Listenpreis den ermäßigten Preis errechnet und ausgibt.**

Das Programm soll mit zwei Argumenten für die Rabattklasse und für den Verkaufspreis aufgerufen werden.

Schleifen (Wiederholungsanweisungen)

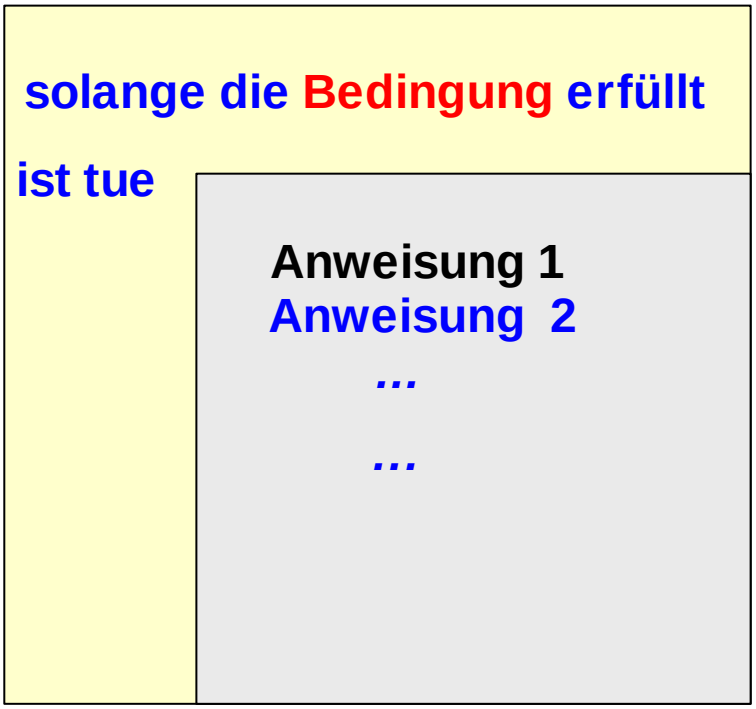
enthalten im *Schleifenkörper* die zu wiederholenden Anweisungen.

In JAVA gibt es drei Arten von Schleifen:

- **die *while*-Schleife**
- **die *for*-Schleife**
- **die *do-while*-Schleife**

while-Anweisung

```
while (Bedingung)
 {   Anweisung1
     Anweisung2
          .....
 }
```



Wie bei if und if-else gilt:

- **Die Bedingung muss in runde Klammern eingeschlossen werden.**

Die while-Schleife heißt auch abweisende Schleife:
Ist die Bedingung schon beim ersten Auswerten falsch, so wird der Schleifenkörper überhaupt nicht ausgeführt.

Beispiel:

```
import java.util.Random ;

public class wuerfel3{
    public static void main (String args[ ]) {
      int zaehler,w ;                          // für die Anzahl der Würfe
      zaehler=1;                               //  1. Wurf
      Random zzahl=new Random();
      w=(zzahl.nextInt(6));                    // Zufallszahl zwischen -5 und 5
      if (w<0)  w=-w;                          // 0<=w<=5
      w=w+1;                                   // 1<=w<=6
      System.out.print(  w +"  " );
      While (w!=6) {                           // ggf. weitere Würfe
          w=(zzahl.nextInt(6));
          if (w<0)  w=-w;
          w=w+1;
          System.out.print(  w +"  " );
          zaehler=zaehler+1;
      }
      System.out.println("");
      System.out.println(zaehler+"  Wuerfe bis zur ersten 6  ");
   }
}
```

Einrücken des Schleifenkörper macht den Quelltext leichter lesbar !!

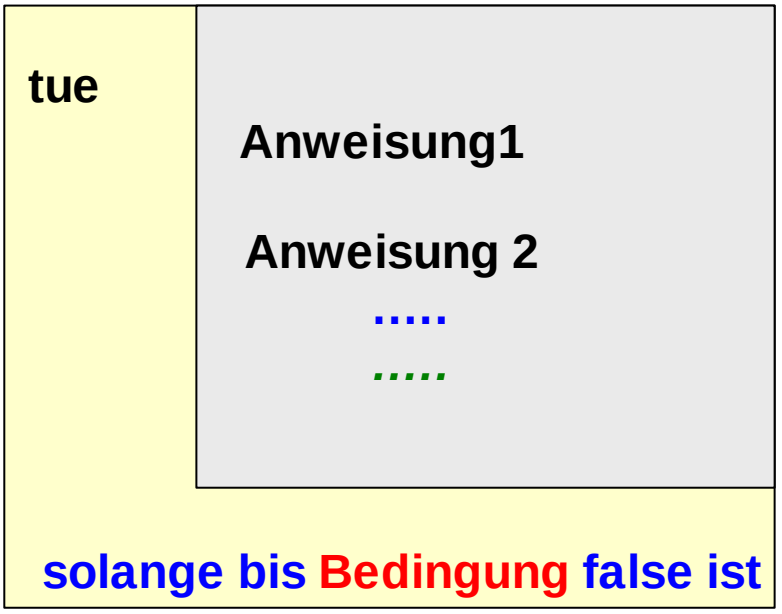
Ein häufiger Programmierfehler sind sogenannte Endlosschleifen, deren Bedingung nie den Wert false annimmt.

do-while-Anweisung

Diese Anweisung ist eng verwandt mit der while-Anweisung. Der Unterschied besteht lediglich im Zeitpunkt der Prüfung der Bedingung. Bei der do-while-Anweisung steht die zu prüfende Bedingung nach dem Schleifenkörper.

JAVA-Syntax:

```
do
   Anweisungsblock
while (Bedingung);
```



Die Bedingung wird erst nach dem Ausführen des Schleifenkörpers geprüft. Der Schleifenkörper wird also unabhängig von der Bedingung im Gegensatz zur while-Schleife wenigstens einmal ausgeführt.

Merke: Am Ende der do-while-Schleife nach while(Bedingung) steht ein Semikolon!

for-Anweisung

Die for-Schleife ist dadurch gekennzeichnet, dass die Zahl der Schleifendurchläufe von vornherein feststehen muss. Die *Bedingung* steht vor dem Schleifenkörper zusammen mit der *Initialisierung* und der *Aktualisierung*.
Java-Syntax:

```
for (Initialisierung; Bedingung; Aktualisierung)
    Anweisungsblock     //Schleifenkörper
```

zähle die Laufvariable ….von …. bis …. mit der Schrittweite ….

und tue bei

jedem Schritt

```
Anweisung 1
Anweisung 2
.....
.....
```

Die Initialisierung wird nur beim ersten Durchlaufen der Schleife ausgeführt. Der Anfangswert der Laufvariablen wird hier festgelegt.

Wie oben kontrolliert eine Bedingung, ob der Schleifen-körper ausgeführt wird.

Die Aktualisierung legt fest, wie sich die Laufvariable nach jedem Schleifendurchlauf ändert.

```
import java.util.Random ;

public class wuerfel4
{
 public static void main (String args[])
  {
    int w,i ;
    Random zzahl=new Random();
     for (i=1; i<=10; i++)
     {
      w=(zzahl.nextInt(6));          // -5<=w<=5
      if (w<0)
          w=-w;                      //  0<=w<=5
      w=w+1;                         //  1<=w<=6
      System.out.println( i + "-ter Wurf :  " +w);
     }
    System.out.println(" Es wurde  " + i +" mal gewürfelt ");
  }
}
```

- Die **Zählvariable** i erhält beim ersten Erreichen der Schleife den Wert 1.
- Die **Bedingung** lautet im Beispiel i <=10 .
- Mit der **Aktualisierung** i++ wird der Wert von i nach jedem Schleifendurchlauf inkrementiert.

Oftmals wird die Laufvariable wie im Beispiel erst mit der for-Schleife deklariert.

```
for ( int i=1; i<=10; i++)
     {
      w=zzahl.nextInt(6));
      System.out.println(" Zufallszahl :   "+w);
     }
```

Dadurch wird der Gültigkeitsbereich der Variablen auf die Schleife beschränkt.

Merke: Das Trennzeichen im Schleifenkopf ist ein Semikolon ; und kein Komma , !

break-Anweisung

Die while, do-while- und for-Schleifen können mittels der break-Anweisung an jeder Stelle innerhalb der Schleife beendet werden.

Kommt eine break-Anweisung im Schleifenkörper zur Ausführung, so wird die Schleife unmittelbar verlassen und die nächste Anweisung nach der Schleife wird ausgeführt. (vgl. auch die Verwendung von break in der switch-Anweisung).

```
for ( int i=1; i<=10; i++)
      {
       w=(zzahl.nextInt(6));
       if w<0
          w=-w;
       w=w+1;
       System.out.println(" Augenzahl :  " +w);
       If (w==6)       // Abbruch wenn eine 6 erscheint
          break        // nur eine Anweisung, daher können die
                       // Klammern entfallen
     }
```