Bisher hast du mit der **while-Schleife** gearbeitet, wie in diesem Beispiel:

```
public void ausgebenWhile()
{
   int i;
   i = 0;                              // Anfangswert für i: 0
   while (i < 10)                      // Wdh., solange i < 10
   {
      Console.println(liste[i]);
      i++;                             // i jedesmal um 1 erhöhen
   }
}
```

Schleifen, die eine Variable in jeder Wiederholung um 1 hoch zählen, sind so häufig, dass viele Programmiersprachen dafür eine spezielle Syntax eingeführt haben: die **for-Schleife**.
Bei der for-Schleife werden Angaben zur Variable in eine Zeile geschrieben, statt wie bei der while-Schleife in drei Zeilen. Die for-Schleife spart also etwas Platz im Quellcode.

Die folgende Methode hat genau die gleiche Bedeutung:

```
public void ausgebenFor()
{
   int i;                              // Anfangswert für i: 0
   for (i = 0; i < 10; i++)            // Wdh., solange i < 10
   {                                   // i jedesmal um 1 erhöhen
      Console.println(liste[i]);
   }
}
```

Allgemein werden bei der for-Schleife drei Angaben gemacht und durch Semikolon getrennt:

	1.		2.		3.	
for (	Anfangswert der Variablen	;	Bedingung, solange wiederholt wird	;	Änderung der Variablen	)

Es ist wichtig zu sehen, dass die drei Angaben zu verschiedenen Zeiten passieren:

1. Der **Anfangswert** der Variablen wird **vor der ersten Wiederholung** gesetzt.
2. Die **Bedingung** wird **vor jeder Wiederholung** geprüft.
   Wenn sie nicht mehr zutrifft, endet die Schleife.
3. Die **Änderung** der Variablen wird **am Ende jeder Wiederholung** durchgeführt.

Die Bezeichnung „for-Schleife“ kommt daher, dass man den Quellcode so lesen kann:
„Wiederhole **für** die Variable i von 0 bis 9 die nachfolgenden Schritte“.

Hinweis: Wenn man mit Arrays arbeitet, verwendet man auch bei der for-Schleife besser „liste.length“ statt einer konkreten Zahl: `for (i = 0; i < liste.length; i++)` …

Aufgabe 1 – Verständnisaufgaben

a) Was passiert hier? Erläutere die folgende Methode mit eigenen Worten.

```
public void wasMacheIch()
{
   int i;
   for (i = 0; i < liste.length; i++)
   {
      liste[i] = 20 * i + 2;
   }
}
```

b) Gib die Werte des Arrays „liste“ nach Ausführen der Methode aus Aufgabe a) an:

0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*

c) Schreibe die Methode wasMacheIch() neu, mit einer while-Schleife statt der for-Schleife.

d) Schreibe eine Methode quadrate(), die das Array „liste“ mithilfe einer for-Schleife mit den ersten 10 Quadratzahlen füllt (0, 1, 4, 9 usw.)

Aufgabe 2

Bearbeite das ausgeteilte BlueJ-Projekt (die Musterlösung des vorigen Arbeitsblatts).
Ersetze in jeder Methode die while-Schleife durch eine entsprechende for-Schleife.

Aufgabe 3

Erweitere die Klasse Zahlenliste aus Aufgabe 2 um je eine Methode für die folgenden Aufgaben.
Löse die Aufgaben jeweils mit einer **for-Schleife**.
Erweitere auch die main-Methode der Hauptklasse, um die neuen Methoden aufzurufen.

a) Den Elementen des Arrays werden die Werte 100, 200, 300 usw. zugewiesen.

b) Den Elementen des Arrays werden die Werte von -75 bis 150 zugewiesen (in 25er-Schritten)

c) Den Elementen des Arrays werden die Werte 1, 3, 6, 10, 15, 21, usw. zuwiesen

d) Den Elementen des Arrays werden die 2er-Potenzen (1, 2, 4, 8 usw.) zugewiesen.
Löse die Aufgabe ohne die Potenzfunktion (Math.pow).

e) Den Elementen des Arrays werden die Fibonacci-Zahlen zugewiesen.
Dabei ist der erste Wert 1, der zweite auch 1. Der dritte ist die Summe des ersten und des zweiten, der vierte die Summe des zweiten und dritten, usw.
Tipp: die ersten beiden Elemente des Arrays weise **vor** der Schleife zu.