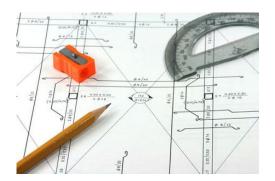
## Begriff "Algorithmus"

Ein **Algorithmus** in der Informatik (oder auch in der Mathematik) ist ein **Verfahren**, mit dem man eine Berechnung durchführen oder eine bestimmte Aufgaben lösen kann. In der Informatik gibt es viele typische Problemstellungen, zum Beispiel:

- Berechnung der Wurzel einer der Zahl mithilfe der Grundrechenarten (das ist ein sogenanntes "numerisches" Verfahren)
- Was ist der kürzeste Weg zwischen zwei beliebigen Orten?
- Sortiere eine Liste von Adressen nach Nachnamen.

Es gibt verschiedene Möglichkeiten, Algorithmen darzustellen:

- umgangsprachlich
- als UML-Aktivitätsdiagramm
- als **Quellcode** in einer Programmiersprache



## **Beispiel**

Für die folgende Aufgabe soll ein **Algorithmus** formuliert werden: Ein Array soll linear mit Werten gefüllt werden, angefangen bei –75, in 25er-Schritten.

## Umgangsprache Aktivitätsdiagramm Setze eine Variable "a" auf -75. Wiederhole für alle Elemente des Arrays: Setze das aktuelle Element auf den Wert Variable a = -75 der Variablen Erhöhe die Variable um 15. **Programmcode** Setze aktuelles Element auf a [für jedes public void fuelleLinear() Element] int i, a; Erhöhe a um 25 a = -75;for (i=0; i<liste.length; i++)</pre> liste[i] = a;a = a + 25;}

## **Aufgabe**

Die folgenden Algorithmen sind als Java-Quellcode dargestellt. Fülle jeweils die Tabelle mit den Werten des Arrays aus. Stelle den Algorithmus dann jeweils 1. in <u>Umgangssprache</u>, 2. als <u>Aktivitätsdiagramm</u> dar.

a) Lineare Füllung in 20er-Schritten

```
public void linear20()
{
   int i;
   for (i = 0; i < liste.length; i++)
   {
      liste[i] = 20 * i;
   }
}

0   1  2  3  4  5  6  7  8  9</pre>
```

b) Füllung mit den "Dreieckszahlen"

```
public void dreieck()
   int i, wert;
   wert = 0;
   for (i = 0; i < liste.length; i++)
      wert = wert + i;
      liste[i] = wert;
}
                                                        7
0
                                        5
                                                6
                                                                8
                                                                        9
        1
                2
                        3
                                4
```

c) Füllen mit der "Fibonacci-Folge"

```
public void fibonacci()
   int i;
   liste[0] = 1;
   liste[1] = 1;
   for (i = 2; i < 10; i++)
      liste[i] = liste[i-1] + liste[i-2];
}
0
                                                        7
                2
                        3
                                        5
                                                6
                                                                8
                                                                        9
        1
                                4
```

