Prof. Andreas Goerdt
M. Pippig, M. Riedel, U. Schönke

TU Chemnitz
Wintersemester 2003/2004

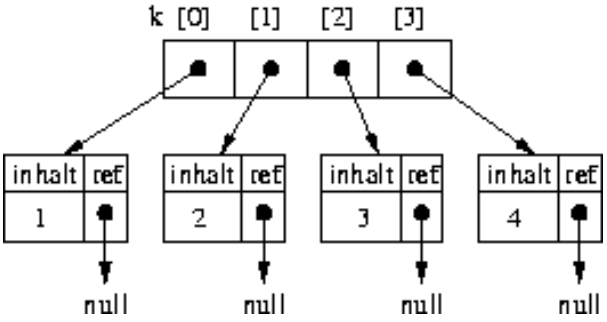
# Algorithmen und Programmierung

## 10. Übung – Lösungsvorschläge

1. Aufgabe:

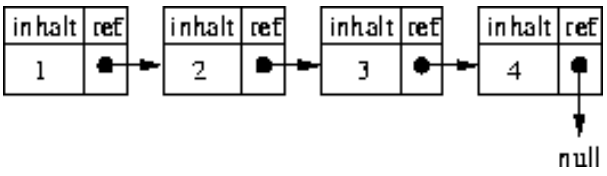
   Es wird ein Feld `k` mit 4 Elementen vom Typ `Komponente` erzeugt. In der ersten `for`-Schleife werden die 4 Komponenten erzeugt (Elemente des Felds sind wieder Referenzvariablen!). Den jeweiligen Instanzvariablen `inhalt` werden die Ziffern des Wertes der Variable `zahl` (rückwärts) zugewiesen, die Instanzvariablen `ref` bekommen alle den Null-Zeiger `null` zugewiesen.

   Datenstrukturen nach der ersten `for`-Schleife:




   In der zweiten `for`-schleife werden den Instanzvariablen `ref` der ersten drei Komponenten Referenzen auf die jeweils nächste Komponente zugewiesen. Wir erhalten eine Liste.

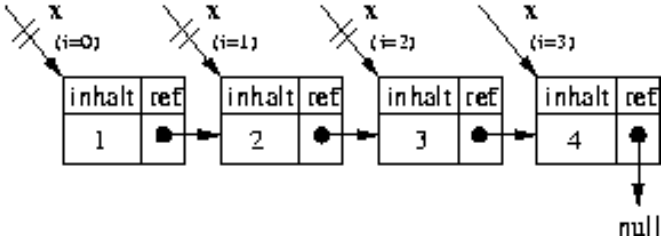
   Komponenten nach der zweiten `for`-Schleife:




   (Das Feld `k` mit den Referenzen auf die 4 Komponenten existiert natürlich noch, ist hier aber nicht mit dargestellt.)

   Nun wird eine weitere Komponente `x` angelegt. Ihr wird die erste Komponente zugewiesen, das heißt, `x` und `k[0]` sind nun identisch, sie zeigen auf die selbe Komponente. In der dritten `for`-Schleife wird nun die Instanzvariable `inhalt` von `x` ausgegeben und `x` selbst erhält den Wert seiner Instanzvariable `ref`. Nach dem ersten Schleifendurchlauf zeigt `x` also auf die zweite Komponente!

   Abarbeitung der dritten `for`-Schleife:




Das Programm liefert also folgende Ausgabe:

```
1
2
3
4
```

2. Aufgabe:

```
public static void spiegeln(int[][] matrix) {

  int i,j,hilf;

  for (i=1; i<matrix.length; i++) {
    for (j=0; j<i; j++) {
      hilf = matrix[i][j];
      matrix[i][j] = matrix[j][i];
      matrix[j][i] = hilf;
    }
  }
  return;
}
```

3. Aufgabe:

   a) Die Methode

```
public static int P(int a, int b) {

  if (b==0) {
    return a;
  }
  else {
    return P(a,b–1)+1;
  }
}
```

   berechnet die Summe von *a* und *b*.

   b) Die Methode

```
public static int Q(int a, int b) {

  if (a<b) {
    return 0;
  }
  else {
    return Q(a–b,b)+1;
  }
}
```

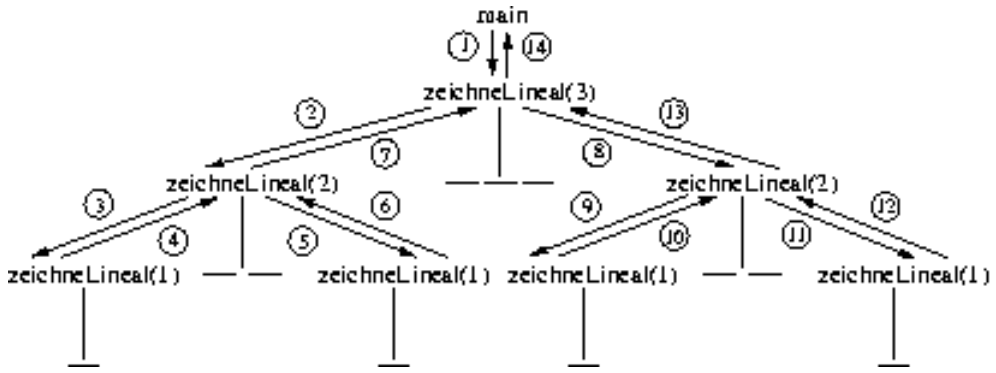
berechnet den Wert der ganzzahligen Division von *a* durch *b*.

4. Aufgabe:

a)

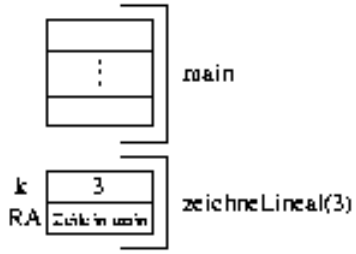
```
1   public static void zeichneLineal(int k) {
2
3     if (k==1) {
4       System.out.println("-");
5     }
6     else {
7       zeichneLineal(k–1);
8       for (int i=1; i<=k; i++) {
9         System.out.print("-");
10      }
11      System.out.println();
12      zeichneLineal(k–1);
13    }
14    return;
15  }
```

b) Aufrufbaum für $k = 3$:

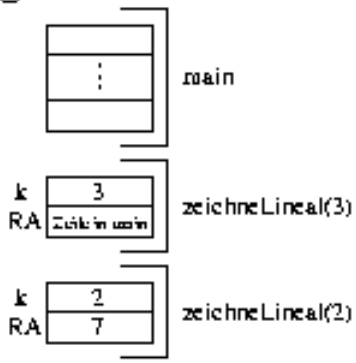



Entwicklung des Laufzeitkellers:

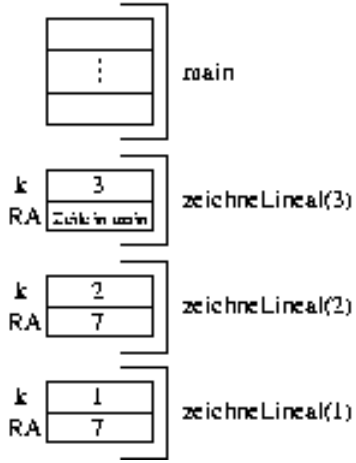
① Aufruf von zeichneLineal(3) in main



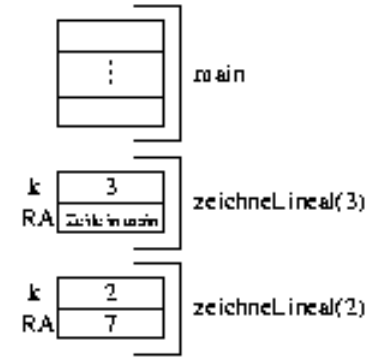

② 1. Aufruf von zeichneLineal(2)



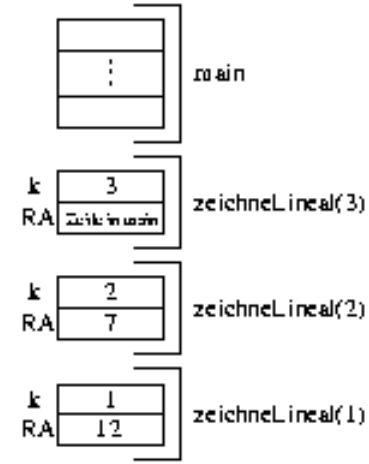

③ 1. Aufruf von zeichneLineal(1)




④ Rückkehr zu zeichneLineal(2)




⑤ 2. Aufruf von zeichneLineal(1)



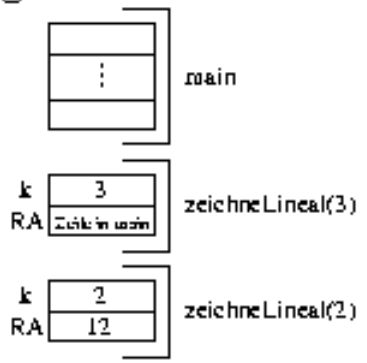

⑥ Rückkehr zu zeichneLineal(2)

⇒ wie bei ④

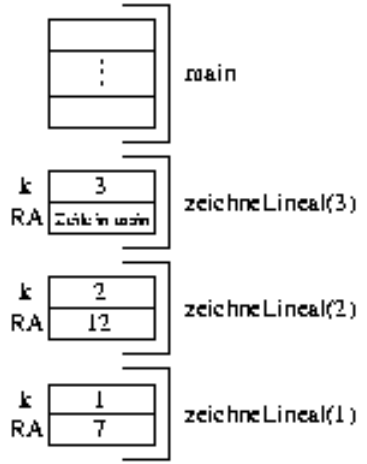
⑦ Rückkehr zu zeichneLineal(3)

⇒ wie bei ①

⑧ 2. Aufruf von zeichneLineal(2)



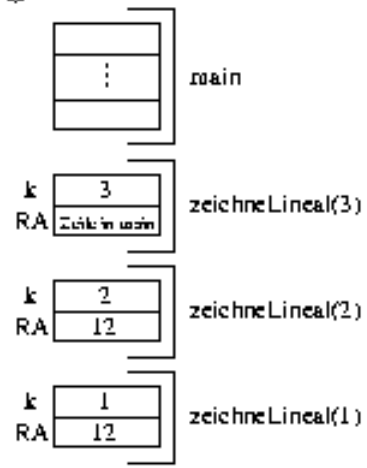

⑨ 1. Aufruf von zeichneLineal(1)




⑩ Rückkehr zu zeichneLineal(2)

⇒ wie bei ⑧

⑪ 2. Aufruf von zeichneLineal(1)




⑫ Rückkehr zu zeichneLineal(2)

⇒ wie bei ⑧

⑬ Rückkehr zu zeichneLineal(3)

⇒ wie bei ①

⑭ Rückkehr zu main