Rekursion

Gilbert Beyer und Annabelle Klarl

Zentralübung zur Vorlesung Einführung in die Informatik

`http://www.pst.ifi.lmu.de/Lehre/wise-11-12/infoeinf`

WS11/12

Inhalte der heutigen Vorlesung:

- Rekursionsbegriff
- Auswertung rekursiver Methodenaufrufe
- Terminierung rekursiver Methoden
- Rekursion vs. Iteration
- Formen der Rekursion
- Rekursiver Sortieralgorithmus: Quicksort

Aufgabe 1)

Schreiben Sie eine Methode, die jeweils für zwei Integerzahlen a und n die Potenz a^n berechnet.

Beispiel

$5^4 = 5 * 5 * 5 * 5$

$5^4 = 5 * 5 * 5^2$

$5^4 = 5 * 5^3$

Präzisierung: $5^n = 5 * \ldots * 5$ (n-mal) $= 5 * 5^{n-1}$

Per Definition: $5^0 = 1$

Induktive Definition:

$a^n = 1$ falls $n = 0$,

$a^n = a * a^{n-1}$ falls $n > 0$

Potenzfunktion realisiert durch rekursive Methode

```
public static int potrek(int a, int n) {
  if (n == 0)
    return 1;
  else
    return a * potrek(a, n - 1);
}
```

Beispiel 5^3

1. Aufruf a = 5, n = 3	potrek(5, 3);
2. Aufruf a = 5, n = 2	5 * potrek(5, 2);
3. Aufruf a = 5, n = 1	5 * (5 * potrek(5, 1));
4. Aufruf a = 5, n = 0	5 * (5 * (5 * potrek(5, 0)));
Rückgabewert = 1	5 * (5 * (5 * 1));
Rückgabewert = 5	5 * (5 * 5);
Rückgabewert = 25	5 * 25;
Rückgabewert = 125	125

Potenzfunktion realisiert durch iterative Methode

```
public static int potiterativ(int a, int n) {
  int akk = 1;
  while (n > 0) {
    akk = a * akk;
    n = n - 1;
  }
  return akk;
}
```

Beispiel 5^3

Aufruf a = 5, n = 3	potiterativ(5, 3);	
1. Anweisung:	akk = 1 ,	n = 3
1. Schleifendurchlauf	akk = 5*1 = 5,	n = 2
2. Schleifendurchlauf	akk = 5*5 = 25,	n = 1
3. Schleifendurchlauf	akk = 5*25 = 125,	n = 0 Schleifenabbruch!

Potenzfunktion realisiert durch endrekursive Methode

```
public static int potiterativ2(int a, int n, int akk) {
  if (n == 0) return akk;
  else return potiterativ2(a, n - 1, a * akk);
}

//Aufruf z.B. für 5³:  potiterativ2(5, 3, 1);
```

Beispiel 5^3

Aufruf a = 5, n = 3, akk = 1	potiterativ2(5, 3, 1);	
1. Aufruf:	akk = 5*1 = 5,	n = 2
2. Aufruf	akk = 5*5 = 25,	n = 1
3. Aufruf	akk = 5*25 = 125,	n = 0

Aufgabe 2)

Gewinnchance beim Lotto:

Wieviele Möglichkeiten gibt es, 6 Zahlen aus gegebenen 49 Zahlen auszuwählen?

Entwickeln Sie einen rekursiven Algorithmus für die Lösung des Problems.

Berechnung klassisch

6 aus 49, in der Statistik übliche Lösung:

Aus der Statistik ist bekannt, dass es $\binom{n}{k}$ Teilmengen mit k Elementen aus einer Menge mit n Elementen gibt:

auswahl (k,n) = $\binom{n}{k}$ für $1 \leq k \leq n$

Berechnung mit Binomialkoeffizient:

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

auswahl (6, 49) = $\binom{49}{6} = \frac{49!}{6!(49-6)!} = \frac{(49 * 48 * 47 \ldots * 1)}{(6*5* \ldots *1) * (43 *42 * \ldots *1)}$

Rekursive Lösungsidee

a) Jede Auswahl ”6 aus 48“ ist auch eine gültige
Auswahl für ”6 aus 49“
(das sind alle Auswahlen, in denen 49 nicht vorkommt)

b) Jede Auswahl ”5 aus 48“ zusammen mit der
<u>festen sechsten Zahl 49</u> ist eine Auswahl für ”6 aus 49“
(das sind alle Auswahlen, in denen 49 vorkommt).

a) + b) liefert alle Auswahlen für ”6 aus 49“

Also: auswahl(6, 49) = auswahl(6, 48) + auswahl(5, 48)

Rekursive Lösungsidee

Also: auswahl(6, 49) = auswahl(6, 48) + auswahl(5, 48)

Allgemein: auswahl(k, n) = auswahl(k, n-1) + auswahl(k-1, n-1)

Im Übrigen gelten ohne Beweis:

auswahl(1, n) = n sowie auswahl(n, n) = 1

Induktive Definition:

auswahl(k, n) = n falls k = 1

auswahl(k, n) = 1 falls k = n

auswahl(k, n) = auswahl(k, n-1) + auswahl(k-1, n-1) sonst

6 aus 49 rekursiv

```
public static int auswahl(int k, int n) {
  if (k == 1) return n;
  else if (k == n) return 1;
  else return auswahl(k,n-1) + auswahl(k-1,n-1);
}
```