VL05: Überblick der Themen

Montag, 10. Mai 2021 09:13

- Einführung
  - Fünf Beispiele
- Rekursionsbasis und -vorschrift
- Rekursiv programmieren
- Analyse rekursiver Programme
- Formen der Rekursion
  - Direkte vs. indirekte Rekursion
  - Divide and Conquer (Teile und Herrsche)
  - Backtracking
- Iterative Lösung oder Rekursion?
- Klassisches Beispiel: Türme von Hanoi

Zu welchen Themen haben Sie Fragen?

Einstiegsfragen

Montag, 10. Mai 2021 09:13

- Was ist Rekursion?
- Was zeichnet einen rekursiven Algorithmus bzw. eine rekursive Methode aus?
- Was beschreibt die Rekursionsbasis?
- Was beschreibt der Rekursionsvorschrift?

Verhalten im Speicher: Umkehren einer Zeichenkette

Donnerstag, 6. Mai 2021 15:09

Java-Programm:

```
public static String reverse(String s)
{
    String erg;

    if (s.length()>1)
    {
        char c = s.charAt(0);
        String rest = s.substring(1);
        erg = reverse(rest) + c;
    }
    else
        erg = s;

    return erg;
}

public static void main(String[] args)
{
    System.out.println(reverse("abc"));
}
```

	Stack
args	…
Rückgabewert	
Parameter s	"abc"
erg	
c	
rest	

Verhalten im Speicher: Umkehren einer Zeichenkette

Donnerstag, 6. Mai 2021 15:09

Java-Programm:

```
public static String reverse(String s)
{
    String erg;

    if (s.length()>1)
    {
        char c = s.charAt(0);
        String rest = s.substring(1);
        erg = reverse(rest) + c;
    }
    else
        erg = s;

    return erg;
}

public static void main(String[] args)
{
    System.out.println(reverse("abc"));
}
```

	Stack
args	...
Rückgabewert	
Parameter s	"abc"
erg	
c	'a'
rest	

Verhalten im Speicher: Umkehren einer Zeichenkette

Donnerstag, 6. Mai 2021 15:09

Java-Programm:

```
public static String reverse(String s)
{
    String erg;

    if (s.length()>1)
    {
        char c = s.charAt(0);
        String rest = s.substring(1);
        erg = reverse(rest) + c;
    }
    else
        erg = s;

    return erg;
}

public static void main(String[] args)
{
    System.out.println(reverse("abc"));
}
```

	Stack
args	...
Rückgabewert	
Parameter s	"abc"
erg	
c	'a'
rest	"bc"

Verhalten im Speicher: Umkehren einer Zeichenkette

Donnerstag, 6. Mai 2021 15:09

Java-Programm:

```
public static String reverse(String s)
{
    String erg;

    if (s.length()>1)
    {
        char c = s.charAt(0);
        String rest = s.substring(1);
        erg = reverse(rest) + c;
    }
    else
        erg = s;

    return erg;
}

public static void main(String[] args)
{
    System.out.println(reverse("abc"));
}
```

	Stack
args	...
Rückgabewert	
Parameter s	"abc"
erg	
c	'a'
rest	"bc"
Rückgabewert	
Parameter s	"bc"
erg	
c	
rest	

Verhalten im Speicher: Umkehren einer Zeichenkette

Donnerstag, 6. Mai 2021 15:09

Java-Programm:

```
public static String reverse(String s)
{
    String erg;

    if (s.length()>1)
    {
        char c = s.charAt(0);
        String rest = s.substring(1);
        erg = reverse(rest) + c;
    }
    else
        erg = s;

    return erg;
}

public static void main(String[] args)
{
    System.out.println(reverse("abc"));
}
```

	Stack
args	...
Rückgabewert	
Parameter s	"abc"
erg	
c	'a'
rest	"bc"
Rückgabewert	
Parameter s	"bc"
erg	
c	'b'
rest	

Verhalten im Speicher: Umkehren einer Zeichenkette

Donnerstag, 6. Mai 2021 15:09

Java-Programm:

```
public static String reverse(String s)
{
    String erg;

    if (s.length()>1)
    {
        char c = s.charAt(0);
        String rest = s.substring(1);
        erg = reverse(rest) + c;
    }
    else
        erg = s;

    return erg;
}

public static void main(String[] args)
{
    System.out.println(reverse("abc"));
}
```

	Stack
args	...
Rückgabewert	
Parameter s	"abc"
erg	
c	'a'
rest	"bc"
Rückgabewert	
Parameter s	"bc"
erg	
c	'b'
rest	"c"

Verhalten im Speicher: Umkehren einer Zeichenkette

Donnerstag, 6. Mai 2021 15:09

Java-Programm:

```
public static String reverse(String s)
{
    String erg;

    if (s.length()>1)
    {
        char c = s.charAt(0);
        String rest = s.substring(1);
        erg = reverse(rest) + c;
    }
    else
        erg = s;

    return erg;
}

public static void main(String[] args)
{
    System.out.println(reverse("abc"));
}
```

	Stack
args	…
Rückgabewert	
Parameter s	"abc"
erg	
c	'a'
rest	"bc"
Rückgabewert	
Parameter s	"bc"
erg	
c	'b'
rest	"c"
Rückgabewert	
Parameter s	"c"
erg	

Verhalten im Speicher: Umkehren einer Zeichenkette

Donnerstag, 6. Mai 2021 15:09

Java-Programm:

```
public static String reverse(String s)
{
    String erg;

    if (s.length()>1)
    {
        char c = s.charAt(0);
        String rest = s.substring(1);
        erg = reverse(rest) + c;
    }
    else
        erg = s;

    return erg;
}

public static void main(String[] args)
{
    System.out.println(reverse("abc"));
}
```

	Stack
args	...
Rückgabewert	
Parameter s	"abc"
erg	
c	'a'
rest	"bc"
Rückgabewert	
Parameter s	"bc"
erg	
c	'b'
rest	"c"
Rückgabewert	
Parameter s	"c"
erg	"c"

Verhalten im Speicher: Umkehren einer Zeichenkette

Donnerstag, 6. Mai 2021 15:09

Java-Programm:

```
public static String reverse(String s)
{
    String erg;

    if (s.length()>1)
    {
        char c = s.charAt(0);
        String rest = s.substring(1);
        erg = reverse(rest) + c;
    }
    else
        erg = s;

    return erg;
}

public static void main(String[] args)
{
    System.out.println(reverse("abc"));
}
```

	Stack
args	...
Rückgabewert	
Parameter s	"abc"
erg	
c	'a'
rest	"bc"
Rückgabewert	
Parameter s	"bc"
erg	
c	'b'
rest	"c"
Rückgabewert	"c"
Parameter s	"c"
erg	"c"

Verhalten im Speicher: Umkehren einer Zeichenkette

Donnerstag, 6. Mai 2021 15:09

Java-Programm:

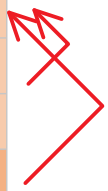
```
public static String reverse(String s)
{
    String erg;

    if (s.length()>1)
    {
        char c = s.charAt(0);
        String rest = s.substring(1);
        erg = reverse(rest) + c;
    }
    else
        erg = s;

    return erg;
}

public static void main(String[] args)
{
    System.out.println(reverse("abc"));
}
```

	Stack
args	…
Rückgabewert	
Parameter s	"abc"
erg	
c	'a'
rest	"bc"
Rückgabewert	
Parameter s	"bc"
erg	"cb"
c	'b'
rest	"c"
Rückgabewert	"c"



Verhalten im Speicher: Umkehren einer Zeichenkette

Donnerstag, 6. Mai 2021 15:09

Java-Programm:

```
public static String reverse(String s)
{
    String erg;

    if (s.length()>1)
    {
        char c = s.charAt(0);
        String rest = s.substring(1);
        erg = reverse(rest) + c;
    }
    else
        erg = s;

    return erg;
}

public static void main(String[] args)
{
    System.out.println(reverse("abc"));
}
```

	Stack
args	...
Rückgabewert	
Parameter s	"abc"
erg	
c	'a'
rest	"bc"
Rückgabewert	**"cb"**
Parameter s	"bc"
erg	"cb"
c	'b'
rest	"c"

Verhalten im Speicher: Umkehren einer Zeichenkette

Donnerstag, 6. Mai 2021 15:09

Java-Programm:

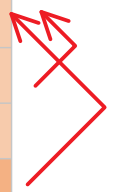
```
public static String reverse(String s)
{
    String erg;

    if (s.length()>1)
    {
        char c = s.charAt(0);
        String rest = s.substring(1);
        erg = reverse(rest) + c;
    }
    else
        erg = s;

    return erg;
}

public static void main(String[] args)
{
    System.out.println(reverse("abc"));
}
```

	Stack
args	...
Rückgabewert	
Parameter s	"abc"
erg	"cba"
c	'a'
rest	"bc"
Rückgabewert	"cb"



Verhalten im Speicher: Umkehren einer Zeichenkette

Donnerstag, 6. Mai 2021 15:09

Java-Programm:

```
public static String reverse(String s)
{
    String erg;

    if (s.length()>1)
    {
        char c = s.charAt(0);
        String rest = s.substring(1);
        erg = reverse(rest) + c;
    }
    else
        erg = s;

    return erg;
}

public static void main(String[] args)
{
    System.out.println(reverse("abc"));
}
```

	Stack
args	...
Rückgabewert	"cba"
Parameter s	"abc"
erg	"cba"
c	'a'
rest	"bc"

Verhalten im Speicher: Umkehren einer Zeichenkette

Donnerstag, 6. Mai 2021 15:09

Java-Programm:

```
public static String reverse(String s)
{
    String erg;

    if (s.length()>1)
    {
        char c = s.charAt(0);
        String rest = s.substring(1);
        erg = reverse(rest) + c;
    }
    else
        erg = s;

    return erg;
}

public static void main(String[] args)
{
    System.out.println(reverse("abc"));
}
```

	Stack
args	…
Rückgabewert	"cba"

Aufgabe: Rekursive Funktion

Donnerstag, 6. Mai 2021 15:09

Rekursive Definition:

$$f(n) = \begin{cases} 1 & \text{für } n \leq 1 \\ f(n-1) + n & \text{für } n > 1 \end{cases}$$

Aufgabe 1:

a) Berechnen Sie f(4).
f(4) = f(3) + 4 = f(2) + 3 + 4 = f(1) + 2 + 3 +4
= 1 + 2 + 3 + 4 = 10

b) Welche mathematische Funktion berechnet f(n)? Geben Sie eine Formel an.
$f(n) =$ **1 + 2 + 3 + … + n =** $\sum_{i=1}^{n} i = \frac{n\,(n+1)}{2}$

Aufgabe 2:

Programmieren Sie die Funktion f rekursiv in Java

```
public static int fRek(int n)
{
   return (n > 1) ?  n + fRek(n-1) : 1;

   // oder
   int erg = 0;
   if (n > 1)
      erg = fRek(n-1)+n;
   else
      erg = n;
    return erg;
}
```

Aufgabe 3:

Programmieren Sie die Funktion f iterativ in Java

```
public static int fIter(int n)
{
    int erg = 0;
    for (int i = 1; i <= n; i++)
       erg += i;
    return erg;
}
```

Aufgabe: Größter gemeinsamer Teiler (ggT)

Donnerstag, 6. Mai 2021 15:09

Rekursive Definition:

$$ggT(m,n) = \begin{cases} m & für\ n = 0 \\ ggT(n, m\ mod\ n) & für\ n > 0 \end{cases}$$

Aufgabe 4:

a) Berechnen Sie ggT(60, 42).
ggT(60, 42) = ggT(42, 60 mod 42) =
ggT(42, 18) = ggt(18, 42 mod 18) =
ggT(18, 6) = ggT(6, 18 mod 6) =
ggT(6, 0) = 6

b) Berechnen Sie ggT(25, 16).
ggT(25, 16) = ggT(16, 25 mod 16) =
ggT(16, 9) = ggT(9, 16 mod 9) = ggT(9, 7) =
ggT(7, 9 mod 7) = ggT(7, 2) =
ggT(2, 7 mod 2) = ggT(2, 1) =
ggT(1, 2 mod 1) = ggT(1, 0) = 1

Aufgabe 5:

Programmieren Sie die ggT-Funktion rekursiv in Java

```
public static int ggTRek(int m, int n)
{
    return (n>0) ? ggTRek(n, m%n) : m;
}
```

Aufgabe 6:

Programmieren Sie die ggT-Funktion iterativ in Java

```
public static int ggTiter(int m, int n)
{
    while(n > 0)
    {
       int h = m % n;
       m = n;
       n = h;
    }
    return m;
}
```

Beispiel: Rekursive Bestimmung der Länge einer Liste

Freitag, 7. Mai 2021 18:41

Rekursionsbasis: Länge einer leeren Liste ist 0
Rekursionsvorschrift: Länge einer nicht-leeren Liste ist
Länge der Restliste ohne das erste Element + 1

```
public class Liste
{
  protected Link anfang;

  //Iterative Lösung

  public int bestimmeAnzahlIter()
  {
     int zaehler=0;
     Link a = anfang;
     while (a != null) {
        zaehler++;
        a = a.naechster;
     }
     return zaehler;
  }

  // Rekursive Lösung

  public int bestimmeAnzahl()
  {
        return bestimmeAnzahlRek(anfang);
  }

  private int bestimmeAnzahlRek(Link zeiger)
  {
        if (zeiger != null)
              return bestimmeAnzahlRek(zeiger.naechster) + 1;
        else
              return 0;
  }
}
```

Aufgabe: Analyse rekursiver Programme

Montag, 10. Mai 2021 09:19

Aufgabe 7:

Bestimmen Sie jeweils die Anzahl der Aufrufe von `tuwas()` für n = 3 und n = 4.

	proz1	proz2	proz3
n=3	**3**	**7**	**3**
n=4	**4**	**15**	**7**

Erläuterung auf der nächsten Folie!

```
public static void proz1(int n)
{
   tuwas();

   if (n > 1)
      proz1(n - 1);
}
```

```
public static void proz2(int n)
{
   tuwas();

   if (n > 1)
   {
      proz2(n-1);
      proz2(n-1);
   }
}
```

```
public static void proz3(int n)
{
   tuwas();

   if (n > 1)
   {
      proz3(n/2);
      proz3(n/2);
   }
}
```

Aufgabe: Analyse rekursiver Programme

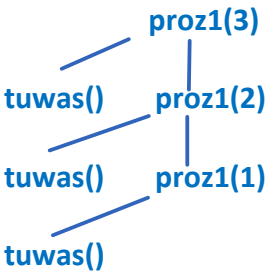
Montag, 10. Mai 2021 09:19

Aufgabe 7:

Bestimmen Sie jeweils die Anzahl der Aufrufe von `tuwas()`
für n = 3 und n = 4.

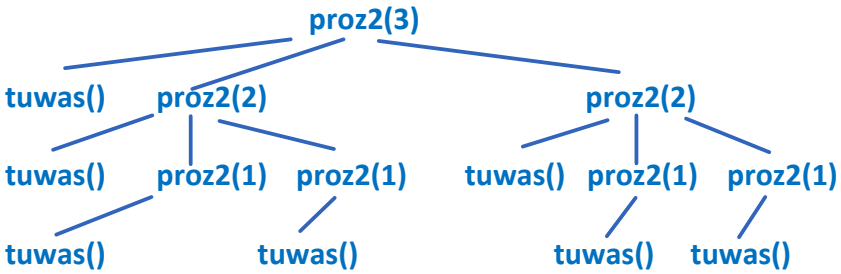
Betrachtung des Aufruf"baums":

proz1(3) ruft einmal tuwas() auf und führt einen rekursiven Aufruf für n-1 durch, usw. proz1(1) erzeugt keinen weiteren rekursiven Aufruf.



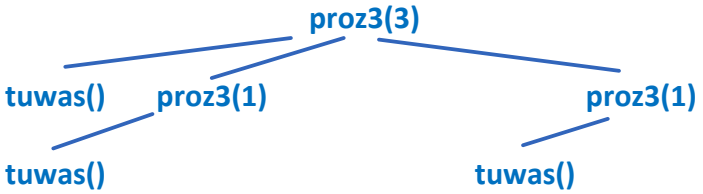
Betrachtung des Aufrufbaums:

proz2(3) ruft einmal tuwas() auf und führt zwei rekursive Aufrufe für n-1 durch, usw. proz2(1) erzeugt keinen weiteren rekursiven Aufruf.



Betrachtung des Aufrufbaums:

proz3(3) ruft einmal tuwas() auf und führt zwei rekursive Aufrufe für n/2 durch, usw. proz3(1) erzeugt keinen weiteren rekursiven Aufruf.



Beispiel: Backtracking: Das N-Damenproblem

Freitag, 7. Mai 2021 18:05

Problemstellung:

Platziere n Damen so auf einem nxn-"Schachfeld", so dass sie sich gegenseitig nicht schlagen

Eine Dame schlägt andere Figuren

- in derselben Reihe
- in derselben Spalte
- in den beiden Diagonalen

Animation:

https://upload.wikimedia.org/wikipedia/commons/1/1f/Eight-queens-animation.gif

Java-Code:

```
private static boolean queens(int[] qs, int col)
{
   if (col == qs.length)
      return true;

   // finde die Erstbeste passende Zeile (row)
   for (int row = 0; row < qs.length; row++)
   {
      if (isSafe(qs, row, col))
      {
         qs[col] = row;
         // wenn auch der Rest gesetzt werden kann,
         // sind wir fertig
         if (queens(qs, col+1)) return true;
      }
   }
   return false;
}
```

Quelle: http://www.gm.fh-koeln.de/ehses/paradigmen/beispiele/Queens.java

Beispiel: Türme von Hanoi

Freitag, 7. Mai 2021 18:30

Algorithmus in Pseudocode:

```
transportiereTurm(n, turm1, turm2, turm3)
 // Transportiere n Scheiben von turm1 nach turm3
 // mittels turm2
 if n >= 1 then
   // Rekursiver Aufruf
   if n > 1 then
     transportiereTurm(n-1, turm1, turm3, turm2)
   end if
   trageScheibe(turm1, turm3)

   // Rekursiver Aufruf
   if n > 1 then
     transportiereTurm(n-1, turm2, turm1, turm3)
   end if
 end if
```

```
trageScheibe(turm1, turm2)
 // Trage eine Scheibe von turm1 nach turm2
 nimmObersteScheibe(turm1)
 legeScheibeAuf(turm2)
```

Animation:

ILIAS: BSP05-Rekursion/BSP05-Hanoi