

Rekursion Ausprobieren

Bekannte Problem die mit Rekursionen gelöst werden

- Palindrom
- String umdrehen
- Grösster gemeinsamer Teiler
- Quicksort
- Türme von Hanoi
- Fibonacci Zahlen
- Ägyptische Multiplikation



Aufgabe:

- ⇒ Wählen Sie mehrere der obigen Aufgaben aus und implementieren Sie diese.
- ⇒ Das Ziel ist mindestens 3 unterschiedliche Probleme zum Implementieren.

Vorgehen:

- Sie finden auf dem Internet rasch fertige Lösungen.
Sie können diese benützen und zum Laufen bringen
- Doch es geht nicht nur um *copy paste* allein
→ **versuchen Sie zu verstehen, was passiert!!**

Treppe ab und Treppe auf

- Notieren Sie die Treppe

Erkennen Sie die Merkmale einer Rekursion?

- die Methoden-Signatur zu finden
- die Abbruchbedingung zu finden
- die Parameter der Methode, wo sie dienen
- Rekursiver Aufruf und die Veränderung beim Aufruf

Hilfen:

- Auf den nächsten Seiten finden Sie Hilfen zu den bekannten Problemen, von eigentlich kochfertigen Lösungen bis nur kurze Beschreibungen.

Ergebnissicherung:

- Screenshot vom Code inklusive Author / Datum
- Screenshto vom Output
- Screenshot vom Kommentar Treppe auf / Treppe ab

Beispiel Ergebnissicherung Fakultät

Rekursive Methode / unten Treppe

```
3 |= */**  
4  * Fakultaet Class  
5  * @author Peter Rutschmann  
6  * @version 10.01.2018  
7  */  
8  public class Fakultaet {  
9  
10 |  public int byLoop(int value){  
11 |    int ergebnis = 1;  
12 |    for(int i=1; i<=value;i++) {  
13 |      ergebnis = ergebnis * i;  
14 |    }  
15 |    return ergebnis;  
16 |  }  
17  
18 |  public int byRekursion(int value){  
19 |    if (value == 1) {  
20 |      return 1;  
21 |    }  
22 |    return (byRekursion( value-1 ) * value);  
23 |  }  
24 }  
25  
26 */  
27 byRekursion(4)          return 6 * 4 = 24  
28 byRekursion(3)          return 2 * 3  
29 byRekursion(2)          return 1 * 2  
30 byRekursion(1)          return 1  
31
```

Output

Berechnen der Fakultaet von: 5

Berechnung mit Loop: 120

Berechnung mit Rekursion: 120

Aufgabe Palindrom rekursiv

Das Problem des Palindroms kennen Sie bereits.
Ein Wort, das von links gelesen gleich lautet wie von rechts gelesen.

Als Vorgabe ein Codefragement:

```
..main(..){  
    ..  
    Scanner keyboard = new Scanner(System.in);  
    System.out.println("Bitte geben Sie einen Text ein:");  
    String userInput = keyboard.nextLine();  
  
    if (isPalindrom(userInput.toCharArray(), 0, userInput.length()-1)) {  
        System.out.println("Die Eingabe ist ein Palindrom");  
    } else {  
        System.out.println("Die Eingabe ist kein Palindrom");  
    }  
    keyboard.close();  
  
}  
..  
private static boolean isPalindrom(char[] data, int front, int rear) {  
    ...//hier müssen Sie selber ergänzen  
}
```

⇒ Ihre Aufgabe... das Fragment in ein Projekt übernehmen und ergänzen.

Aufgabe "String umdrehen"

Das zu erstellende Programm soll vom Benutzer auf der Konsole einen String einlesen und diesen dann in umgekehrter Zeichenfolge wieder auf der Konsole herausschreiben.

⇒ Mehr Hilfe gibt es dazu nicht 😊

Aufgabe "Grösster gemeinsamer Teiler"

rekursiv

```
static int gcd (int x, int y) {  
    int rest = x % y;  
    if (rest == 0) return y;  
    else return gcd(y, rest);  
}
```

iterativ

```
static int gcd (int x, int y) {  
    int rest = x % y;  
    while (rest != 0){  
        x = y; y = rest;  
        rest = x % y;  
    }  
    return y;  
}
```

Ihre Aufgabe... implementieren sie beide Lösungen.

Aufgabe "Quicksort"

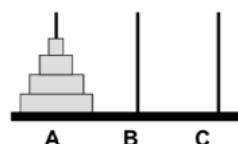
Siehe:

<https://www.baeldung.com/java-quicksort>

Aufgabe "Türme von Hanoi"

Bei den Türmen von Hanoi geht es darum den Turm auf einem anderen Stab umzuschichten. Dabei darf jeweils

- Gegeben sind **3 Pfosten** mit **n Scheiben** (unterschiedlicher Größe)
- **Grundstellung:** alle Scheiben nach Größe geordnet auf Pfosten A
- **Ziel:** Lege alle n Scheiben von A nach C
- **Restriktion 1:** jeweils nur eine Scheibe darf bewegt werden
- **Restriktion 2:** niemals darf eine größere auf einer kleineren Scheibe liegen
- **Lösungsstrategie:**
Problem allg. für Turm der **Höhe n** lösen; folgende Fälle sind zu unterscheiden
 - **n = 0:** gar nichts machen
 - **n > 0:**
 - (1) Turm der **Höhe n-1** von **A nach B** bewegen (mittels C)
 - (2) **Scheibe von A nach C legen**
 - (3) Turm der **Höhe n - 1** von **B nach C** bewegen (mittels A)



Aufgabe " Fibonacci-Reihe "

Aus der Mathematik kennen Sie vielleicht die Fibonacci-Reihe, welche folgendermassen aussieht:

1, 1, 2, 3, 5, 8, 13, ...

Wikipedia:

Die **Fibonacci-Folge** ist die unendliche Folge von [natürlichen Zahlen](#), die (ursprünglich) mit zweimal der Zahl 1 beginnt oder (häufig, in moderner Schreibweise) zusätzlich mit einer führenden Zahl 0 versehen ist.^[1] Im Anschluss ergibt jeweils die Summe zweier aufeinanderfolgender Zahlen die unmittelbar danach folgende Zahl:

0, 1, 1, 2, 3, 5, 8, 13, ...
(optional) 0+1 1+1 1+2 2+3 3+5 5+8

Lösung mit For-Loop

```
value = 10;
System.out.println("Berechnen der ersten " + value + " Fibonacci-Zahlen:");
int n0 = 0;
int n1 = 1;
for (int i=1; i<=value; i++)
{
    int result = n0 + n1;
    System.out.print(n0+n1 + " ");
    n1=n0;
    n0=result;
}
```

Ausgabe auf die Konsole

```
Berechnen der ersten 10 Fibonacci-Zahlen:
1 1 2 3 5 8 13 21 34 55
```

Ihre Aufgabe... ändern Sie die Implementierung, so dass es eine Rekursion ist.

Tipps:

Die Rekursive **Methode** in der Klasse Fibonacci könnte diese **Signatur** haben.

```
public int calculate(int value)
```

Aufruf der Methode:

```
System.out.println("Berechnen der ersten " + value + " Fibonacci-Zahlen:");
for (int i=1; i<=value; i++)
{
    System.out.print(myFibonacci.calculate(i) + " ");
```

Aufgabe "Ägyptische Multiplikation"

(Quelle: 24.10.2019 <http://www.inf.fu-berlin.de/lehre/SS99/ALP2/material/folienalp2-2.pdf>)

2 Einfache imperativen Algorithmen in Java

Am Beispiel von $2807 \cdot 7$ im Papyrus Rhind, 16. Jh. v. Chr.

◆ Das Ägyptische Multiplizieren

Gegeben zwei natürliche Zahlen $a, b > 0$

Auch bekannt als „Methode der russischen Bauern“

Halbiere b , verdopple a mit dem Ergebnis (a', b') solange, bis $b' = 1$

Summiere alle a' , für die b' ungerade

Ergebnis: $a \times b$

$a=17 \quad b=11$

17	11
34	5
68	2
136	1

$$a \times b = 136+34+17 = 187$$

Wenn ihr nur duplizieren und halbieren könntet, so könntet ihr das übrige ohne das Eins mal Eins multipliciren.
Christian von Wolff, 1679- 1754
(Philosoph und Mathematiker,
Univ. Halle)

8	9
16	4
32	2
64	1

hs / fub - alp2-2 1