

# Rekursion Übungen 1

---



## (A1) Potenzberechnung

Implementiere eine rekursive Methode `Potenz(a, n)`, die bei Eingabe einer Dezimalzahl `a` und einer natürlichen Zahl `n` als Ergebnis die Potenz  $a^n$  zurückgibt.

*Beispiel:* Der Aufruf `Potenz(2.5, 3)` gibt den Wert 15,625 zurück.

### Hinweis 1

Beginne mit einem Methodengerüst, welches eine Verzweigung enthält, die den Basisfall und den Rekursionsfall unterscheidet.

Was ist der Basisfall - also wann weißt du sofort, was die Potenz ist, ohne zu überlegen (unabhängig von der Basis, betrachte den Exponenten).

### Hinweis 2: Pseudocode

```
potenz(double basis, int exponent):  
    wenn exponent ist gleich 0:  
        return 1  
    sonst  
        return basis * potenz(basis, exponent-1)
```

### Hinweis 3: Methodengerüst mit Basisfall

```
public double potenz(double b, int e)  
{  
    if(e==0) {  
        return 1;  
    } else {  
        // Rekursionsfall ??  
    }  
}
```

Was muss im Rekursionsfall berechnet werden? Welchen Aufrufparameter muss man beim rekursiven Aufruf der Methode verändern, damit der Basisfall irgendwann erreicht wird?

---



## (A2) Verzinsung

Implementiere eine rekursive Methode `Guthaben(g, z, n)`, die bei Eingabe eines Guthabens `g` in Euro, eines Zinssatzes `z` in Prozent und einer Laufzeit `a` in Jahren als Ergebnis das verzinste Guthaben nach Ende der Laufzeit zurückgibt.

*Beispiel:* Der Aufruf `Guthaben(1000, 1, 2)` gibt den Betrag 1020,10 (€) zurück.

### Hinweis

Das funktioniert genau wie Aufgabe 1, wenn man sich klar macht, dass das Guthaben nach `a` Jahren berechnet werden kann als:

$$G(b, z, a) = b * (1 + (z/100))^a$$

---



## (A3) Fibonacci-Zahlen

Implementiere eine rekursive Methode `Fibonacci(n)`, die bei Eingabe einer natürlichen Zahl `n` als Ergebnis die `n`-te Fibonacci-Zahl zurückgibt. Die erste und zweite Fibonacci-Zahl ist jeweils 1. Die weiteren Fibonacci-Zahlen berechnen sich als Summe der beiden Vorgängerzahlen. Die ersten zehn Fibonacci-Zahlen lauten: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55.

*Beispiel:* Der Aufruf `Fibonacci(11)` gibt die Zahl 89 zurück.

### Hinweis

Das ist ein Beispiel, in dem sich die Funktion im Rekursionsfall mehr als einmal selbst aufruft. Es muss ja `fibonacci(n-1)` und `fibonacci(n-2)` bekannt sein, um `fibonacci(n)` auszurechnen.

Fange an wie immer: Fallunterscheidung, Basisfall. Überlege dann, was im Rekursionsfall geschehen muss.

---



## (A4) Palindrom

Palindrome sind Wörter wie OTTO oder RELIEFPFEILER, die vorwärts wie rückwärts gelesen gleich sind. Implementiere eine rekursive Methode `palindrom(text, l, r)`, die bei Eingabe eines Strings `text` sowie einer linken Feldgrenze `l` und einer rechten Feldgrenze `r` überprüft, ob `text[l..r]` ein

Palindrom ist. Das Ergebnis der Methode soll ein Wahrheitswert sein.

Beispiele:

- Der Aufruf `palindrom("OTTO", 0, 3)` gibt `true` zurück.
- `palindrom("ORTO", 0, 3)` gibt `false` zurück.

From:  
<https://www.info-bw.de/> -

Permanent link:  
<https://www.info-bw.de/faecher:informatik:oberstufe:algorithmen:rekursion:uebungen01:start?rev=1683623170>

Last update: **09.05.2023 09:06**

