

# Algorithmen und Datenstrukturen

## Sommersemester 2022

### Woche 2

Kevin Angele, Tobias Dick, Oskar Neuhuber,  
Andrea Portscher, Monika Steidl, Laurin Wischounig

Abgabe bis 22.03.2022 23:59  
Besprechung im PS am 24.03.2022

#### Aufgabe 1 (2 Punkte): Beweis durch Widerspruch

Beweisen Sie durch Widerspruch, dass  $2^{2n} \notin \mathcal{O}(2^n)$ .

#### Aufgabe 2 (3 Punkte): Laufzeit-Komplexität von Programmen

Bestimmen Sie die Laufzeit-Komplexität der folgenden Algorithmen in Groß-O-Notation in Abhängigkeit von  $n$ .

##### Algorithmus 1:

---

```
1: Input: matrices  $a, b, c$  of size  $n \times n$ 
2: for  $i \leftarrow 0$  to  $n - 1$  do
3:   for  $j \leftarrow 0$  to  $n - 1$  do
4:     for  $k \leftarrow 0$  to  $n - 1$  do
5:        $c[i][j] \leftarrow c[i][j] + a[i][k] + b[k][j]$ 
6:     end for
7:   end for
8: end for
```

---

### Algorithmus 2:

---

```
1: sum  $\leftarrow$  0
2: for  $i \leftarrow 1$  to  $n$  do
3:   for  $j \leftarrow 1$  to 10000 do
4:      $sum \leftarrow sum + j + i$ 
5:   end for
6: end for
```

---

### Algorithmus 3:

---

```
1: Input: array  $a$  of size  $n$ 
2: while  $swapped$  do
3:    $swapped \leftarrow false$ 
4:   for  $j \leftarrow 1$  to  $n - 1$  do
5:     if  $a[j - 1] > a[j]$  then
6:        $swap(a[j - 1], a[j])$ 
7:        $swapped \leftarrow true$ 
8:     end if
9:   end for
10: end while
```

---

## Aufgabe 3 (2 Punkte): Tilde-Approximation

Bestimmen Sie die Omega-, Theta- und Tilde-Approximation der folgenden Funktionen von  $n$ .

- $10n$
- $66n^2 + 17n^3 - 120n$
- $5 + 8 \log_2 n$
- $4 \cdot 2^n + 9n^{100}$

## Aufgabe 4 (3 Punkte): Rekursion

Erstellen Sie einen rekursiven Algorithmus für die folgenden drei Funktionen. Geben Sie (Pseudo)code ab.

1.  $pow(a, b)$ , gibt  $a^b$  zurück. Die Funktion muss nur für natürliche Zahlen definiert sein.
2.  $fib(n)$ , gibt  $n$ -tes Element der Fibonacci-Folge zurück. Probieren sie eine Lösung zu finden, die eine Laufzeitkomplexität von  $\mathcal{O}(n)$  hat.
3.  $isPalindrome(str)$ , Funktion, die bestimmt ob ein String ein Palindrom ist. Ein Palindrom ist ein Wort, das rückwärts gelesen sich selbst ergibt (z.B. "bob", "reittier", "lagerregal", "a", "a").