

## Algorithmen I Tutorium 19

**Wer?** Florian Tobias Schandinat

**Wo?** 50.34, Raum -118

**Wann?** jeden Donnerstag 15:45-17:15

## Material online

[http://github.com/schandinat/algorithmen1\\_ss11](http://github.com/schandinat/algorithmen1_ss11)

# 1. Übungsblatt

$$O(1) \not\subseteq O(0)$$

# 1. Übungsblatt

$$O(1) \not\subseteq O(0)$$

Annahme:  $\exists c \in \mathbb{R}_+ \exists n_0 \in \mathbb{N} : 1 \leq c \cdot 0 \quad \forall n > n_0$

# 1. Übungsblatt

$$O(1) \not\subseteq O(0)$$

Annahme:  $\exists c \in \mathbb{R}_+ \exists n_0 \in \mathbb{N} : 1 \leq c \cdot 0 \quad \forall n > n_0$

da  $1 \neq 0 = c \cdot 0 \quad \forall c \in \mathbb{R} \forall n \in \mathbb{N}$

$\Rightarrow O(1) \not\subseteq O(0)$

# 1. Übungsblatt

$O(1) \not\subseteq O(0)$

Annahme:  $\exists c \in \mathbb{R}_+ \exists n_0 \in \mathbb{N} : 1 \leq c \cdot 0 \quad \forall n > n_0$

da  $1 \neq 0 = c \cdot 0 \quad \forall c \in \mathbb{R} \forall n \in \mathbb{N}$

$\Rightarrow O(1) \not\subseteq O(0)$

Anschaulich:

$O(0)$  der Codeabschnitt wird **nicht** ausgeführt

$O(1)$  der Codeabschnitt wird  $k$ -Mal ausgeführt,  $k \in \mathbb{N}_+$

## Mastertheorem

$$T(n) = 1, n = 1$$

$$T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n), n > 1$$

$$\theta(n^{\log_b a}) \quad , \quad f(n) = O(n^{\log_b(a)-\epsilon}), \epsilon > 0$$

$$\theta(n^{\log_b a} \cdot \log(n)) \quad , \quad f(n) = \theta(n^{\log_b a})$$

$$\theta(f(n)) \quad , \quad f(n) = \Omega(n^{\log_b(a)+\epsilon}), \epsilon > 0$$

## binary search

```
PROCEDURE BinaereSuche(A, v, l, r)
  if  $l > r$  then
    return NIL
   $m = \lfloor \frac{l+r}{2} \rfloor$ 
  if  $v == A[m]$  then
    return m
  else if  $v > A[m]$  then
    return BinaereSuche(A, v, m + 1, r)
  else
    return BinaereSuche(A, v, l, m - 1)
  end if
```

## binary search

```
PROCEDURE BinaereSuche(A, v, l, r)
  if  $l > r$  then
    return NIL
   $m = \lfloor \frac{l+r}{2} \rfloor$ 
  if  $v == A[m]$  then
    return m
  else if  $v > A[m]$  then
    return BinaereSuche(A, v, m + 1, r)
  else
    return BinaereSuche(A, v, l, m - 1)
  end if
```

$$T(n) = 1 \cdot T\left(\frac{n}{2}\right) + 1$$



## binary search

```
PROCEDURE BinaereSuche(A, v, l, r)
  if  $l > r$  then
    return NIL
   $m = \lfloor \frac{l+r}{2} \rfloor$ 
  if  $v == A[m]$  then
    return m
  else if  $v > A[m]$  then
    return BinaereSuche(A, v, m + 1, r)
  else
    return BinaereSuche(A, v, l, m - 1)
  end if
```

$$T(n) = 1 \cdot T\left(\frac{n}{2}\right) + 1$$
$$\implies \theta(\log(n))$$

## binary tree traversal

```
PROCEDURE tree(n)
```

```
  if  $n.left \neq NIL$ 
```

```
    tree(n.left)
```

```
  if  $n.right \neq NIL$ 
```

```
    tree(n.right)
```

## binary tree traversal

PROCEDURE tree(*n*)

if *n.left*  $\neq$  *NIL*

    tree(*n.left*)

if *n.right*  $\neq$  *NIL*

    tree(*n.right*)

$$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + 1$$

## binary tree traversal

PROCEDURE tree(*n*)

if *n.left*  $\neq$  *NIL*

    tree(*n.left*)

if *n.right*  $\neq$  *NIL*

    tree(*n.right*)

$$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + 1$$
$$\implies \theta(n)$$

$$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + n$$

$$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + n$$
$$\implies \theta(n \cdot \log(n))$$

$$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + n$$
$$\implies \theta(n \cdot \log(n))$$

$$T(n) = 4 \cdot T\left(\frac{n}{2}\right) + n^3$$

$$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + n$$
$$\implies \theta(n \cdot \log(n))$$

$$T(n) = 4 \cdot T\left(\frac{n}{2}\right) + n^3$$
$$\implies \theta(n^3)$$



# Amortisierte Analyse

$U$  Universalmenge

$K$  Schlüsselmenge

$h$  Hashfunktion

$$h : U \rightarrow K$$

$U$  Universalmenge

$K$  Schlüsselmenge

$h$  Hashfunktion

$$h : U \rightarrow K$$

**Hinweis:** Bei  $|K| < |U|$  **immer** Kollisionen möglich

- Verkettung
- Offene Adressierung
  - linear
  - quadratisch
  - doppeltes Hashing

$$U = \{i \in \mathbb{N}_0 \mid i \leq 20\}, K = \{i \in \mathbb{N}_0 \mid i \leq 6\}, h(k) = k \bmod 7$$

Skizziere die folgenden Operationen mit Verkettung; offener Adressierung mit linearer Konfliktauflösung

- INSERT(2)
- INSERT(3)
- INSERT(4)

$$U = \{i \in \mathbb{N}_0 \mid i \leq 20\}, K = \{i \in \mathbb{N}_0 \mid i \leq 6\}, h(k) = k \bmod 7$$

Skizziere die folgenden Operationen mit Verkettung; offener Adressierung mit linearer Konfliktauflösung

- INSERT(2)
- INSERT(3)
- INSERT(4)
- INSERT(10)

$$U = \{i \in \mathbb{N}_0 \mid i \leq 20\}, K = \{i \in \mathbb{N}_0 \mid i \leq 6\}, h(k) = k \bmod 7$$

Skizziere die folgenden Operationen mit Verkettung; offener Adressierung mit linearer Konfliktauflösung

- INSERT(2)
- INSERT(3)
- INSERT(4)
- INSERT(10)
- INSERT(11)

$$U = \{i \in \mathbb{N}_0 \mid i \leq 20\}, K = \{i \in \mathbb{N}_0 \mid i \leq 6\}, h(k) = k \bmod 7$$

Skizziere die folgenden Operationen mit Verkettung; offener Adressierung mit linearer Konfliktauflösung

- INSERT(2)
- INSERT(3)
- INSERT(4)
- INSERT(10)
- INSERT(11)
- DELETE(4)



$$U = \{i \in \mathbb{N}_0 \mid i \leq 20\}, K = \{i \in \mathbb{N}_0 \mid i \leq 6\}, h(k) = k \bmod 7$$

Skizziere die folgenden Operationen mit Verkettung; offener Adressierung mit linearer Konfliktauflösung

- INSERT(2)
- INSERT(3)
- INSERT(4)
- INSERT(10)
- INSERT(11)
- DELETE(4)
- SEARCH(2)

$$U = \{i \in \mathbb{N}_0 \mid i \leq 20\}, K = \{i \in \mathbb{N}_0 \mid i \leq 6\}, h(k) = k \bmod 7$$

Skizziere die folgenden Operationen mit Verkettung; offener Adressierung mit linearer Konfliktauflösung

- INSERT(2)
- INSERT(3)
- INSERT(4)
- INSERT(10)
- INSERT(11)
- DELETE(4)
- SEARCH(2)
- SEARCH(5)

$$U = \{i \in \mathbb{N}_0 \mid i \leq 20\}, K = \{i \in \mathbb{N}_0 \mid i \leq 6\}, h(k) = k \bmod 7$$

Skizziere die folgenden Operationen mit Verkettung; offener Adressierung mit linearer Konfliktauflösung

- INSERT(2)
- INSERT(3)
- INSERT(4)
- INSERT(10)
- INSERT(11)
- DELETE(4)
- SEARCH(2)
- SEARCH(5)
- SEARCH(11)

**Vielen Dank für die  
Aufmerksamkeit!**