Exercise sheet 03

Exercise 3.1

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0 \quad \Rightarrow f \le O(g)$$
$$= C \in \mathbb{R}^+ \quad \Rightarrow f = \Theta(g)$$
$$= \infty \quad \Rightarrow f \ge \Omega(g)$$

(a)

(1) $\frac{1}{5} n^3 \ge \Omega(10n^2)$ ✓

$$\lim_{n\to\infty} \frac{\frac{1}{5}n^3}{10n^2} = \lim_{n\to\infty} \frac{1}{50} n = \infty$$

(2) $n^2 + 3n = \Theta(n^2 \log(n))$ ✗

$$\lim_{n\to\infty} \frac{n^2+3n}{n^2 \cdot \log(n)} = \lim_{n\to\infty} \underbrace{\frac{1}{\log(n)}}_{\to 0} + \underbrace{\frac{3}{n\log(n)}}_{\to 0} = 0$$

(3) $5n^4 + 3n^2 + n + 8 = \Theta(n^4)$ ✓

$$\lim_{n\to\infty} \frac{5n^4+3n^2+n+8}{n^4} = \lim_{n\to\infty} 5 + \underbrace{\frac{3}{n^2}}_{\to 0} + \underbrace{\frac{1}{n^3}}_{\to 0} + \underbrace{\frac{8}{n^4}}_{\to 0} = 5 \in \mathbb{R}^+$$

(4) $3^n \ge \Omega(2^n)$ ✓

$$\lim_{n\to\infty} \frac{3^n}{2^n} = \lim_{n\to\infty} \left(\frac{3}{2}\right)^n = \infty$$

(b)

(1) $(\sin(n)+2)n = \Theta(n)$

$$-1 \le \sin(x) \le 1$$

$$\Rightarrow 1 \le \sin(n)+2 \le 3$$

$$\Rightarrow \underbrace{n \le (\sin(n)+2)\cdot n}_{n \le O((\sin(n)+2)\cdot n)} \le 3n$$

$$\Rightarrow (\sin(n)+2)\cdot n \le O(n)$$

$$\Rightarrow (\sin(n)+2)\cdot n = \Theta(n)$$

(2) $\sum_{i=1}^{n} \sum_{j=1}^{i} j = \Theta(n^3)$

$$\sum_{i=1}^{n} \sum_{j=1}^{i} \overset{n}{j} \le \sum_{i=1}^{n} \sum_{j=1}^{i} n = \sum_{i=1}^{n} i \cdot n \le \sum_{i=1}^{n} n^2 = n^3$$

$$\Rightarrow \underline{\sum \sum \le O(n^3)}$$

$$\sum_{j=1}^{i} j \geq \frac{1}{4} i^2 \qquad \sum_{i=1}^{n} i^2 \geq \frac{1}{8} n^3$$

$$\sum_{i=1}^{n} \boxed{\sum_{j=1}^{i} j} \geq \sum_{i=1}^{n} \frac{1}{4} i^2 = \frac{1}{4} \boxed{\sum_{i=1}^{n} i^2} \geq \frac{1}{4} \cdot \frac{1}{8} \cdot n^3 = \underbrace{\frac{1}{32}}_{= C} \cdot n^3$$

$$\Sigma\Sigma \geq C \cdot n^3$$

$$\Rightarrow n^3 \leq O(\Sigma\Sigma j)$$

$$\Rightarrow \Sigma\Sigma = \Theta(n^3)$$

(3)

z.z.: $\log(n^4 + n^3 + n^2) \leq O(\log(n^3 + n^2 + n))$

$C \cdot \log(n^3+n^2+n)$

$$\log(n^4 + \underbrace{n^3}_{\geq n^4} + \underbrace{n^2}_{\geq n^4}) \leq \log(3n^4) = \log(3) + 4 \cdot \log(n) \leq \log(n^3+n^2+n) + \frac{4}{3} \cdot \log(n^3+n^2+n)$$

$$= \underbrace{\frac{7}{3}}_{= C} \cdot \log(n^3+n^2+n)$$

$\log(3) \leq \log(n^3+n^2+n)$

$4 \cdot \log(n) = 4 \cdot \frac{1}{3} \cdot \log(n^3) \leq \frac{4}{3} \cdot \log(n^3+n^2+n)$

(4)

$$\sum_{i=1}^{n} \sqrt{i} = \Theta(n\sqrt{n})$$

$\leq$

$$\sum_{i=1}^{n} \sqrt{i} \leq \sum_{i=1}^{n} \sqrt{n} = n \cdot \sqrt{n} \quad \Rightarrow \sum_{i=1}^{n} \sqrt{i} \leq O(n\sqrt{n})$$

$$\sum_{i=1}^{n} \sqrt{i} \geq \sum_{i=\lceil \frac{n}{2} \rceil}^{n} \sqrt{i} = \sum_{i=\lceil \frac{n}{2} \rceil}^{n} \sqrt{\left\lceil \frac{n}{2} \right\rceil} \geq \frac{n}{2} \cdot \sqrt{\frac{n}{2}} = \underbrace{\frac{1}{2\sqrt{2}}}_{= C} \cdot n \cdot \sqrt{n}$$

$$O\left(\sum_{i=1}^{n} \sqrt{i}\right) \geq n \cdot \sqrt{n}$$

$$\Rightarrow \sum_{i=1}^{n} \sqrt{i} = \Theta(n\sqrt{n})$$

Exercise 3.2

$S = \underbrace{0100100010}_{n}$ $\qquad S = "0110" \quad k = 2$

a)

```
c ← 0
for i ← 0,…,n-1
    for j ← i,…,n-1
        x ← 0
        for l ← i,…,j do
            if S[l] = 1
                x++
        if (x == k)
            c++
```

$\Rightarrow O(n^3)$

b) "0110"

T[i]

$i=0$: 0
$i=1$: 01
$i=2$: 011, 0110
$i=3$: $\emptyset$
$i=4$: $\emptyset$

$T = \{1,1,2,0,0\}$

```
T ← int[n+1]
x ← 0
for i ← 0,…,n-1 do
    if (S[i] == 1)
        x++
    T[x]++
return T
```

$\Rightarrow O(n)$

c) Prefixtable, Suffixtable

$\underbrace{0\,1\,0\,0\,1\,\ldots\,0\,1\,1}_{n}$

$m = 2$

Spanning(m, k, S)

```
T₁ ← SUFFIXTABLE(S[0…m])
T₂ ← PREFIXTABLE(S[m+1,…,n-1])
```

$O(n)$
$O(n)$

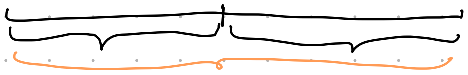
$T_1[i] \mathrel{\hat{=}}$ Anzahl bitstrings mit i einsen, die bei m enden

$T_2[i] \mathrel{\hat{=}}$ Anz. Bits. mit i einsen, die bei m beginnen

return $\sum_{i=\max(0,k-(n-m-1))}^{\min(k,m)} T_1[i] \cdot T_2[k-i]$ $O(n)$

$\Rightarrow O(n)$

$O(n \log n)$



```
Substringcount (S, k, i=0, j=n-1)
    if i=j
        if k=1 and S[i]=1 then
            return 1
        else if k=0 and S[i]=0 then
            return 1
        else
            return 0
    else
        m ← ⌊(i+j)/2⌋
        return Substringcount (S,k,i,m) + Substringcount (S,k,m+1,j) + SPANNING (m,k,S)
```

$$A(n) = 2A\left(\frac{n}{2}\right) + O(n) \leq O(n \cdot \log(n))$$

Exercise 3.4 $\qquad f_{n+1} = f_n + f_{n-1} \qquad f_0 = 0,\ f_1 = 1$

z.z. $f_n \geq \frac{1}{3} \cdot 1{,}5^n$

Base case $\quad n=1, n=2$

$n=1,\ f_1 = 1 \geq 0{,}5 = \frac{1}{3} \cdot 1{,}5^1$ ✓

$n=2,\ f_2 = 1 \geq 0{,}75 = \frac{1}{3} \cdot 1{,}5^2$ ✓

I.H.

$f_k \geq \frac{1}{3} \cdot 1{,}5^k$ gilt für ein $k \geq 1$

$f_{k+1} \geq \frac{1}{3} \cdot 1{,}5^{k+1}$ für $k+1$ gilt.

I.S. $\quad k \wedge k+1 \rightarrow k+2$

$$f_{k+2} = f_{k+1} + f_k$$
$$\overset{\text{I.H.}}{\geq} \frac{1}{3} \cdot 1{,}5^{k+1} + \frac{1}{3} \cdot 1{,}5^k$$
$$= \frac{1}{3} \cdot 1{,}5^k (1{,}5 + 1)$$
$$= \frac{1}{3} \cdot 1{,}5^k \cdot 2{,}5$$
$$\geq \frac{1}{3} \cdot 1{,}5^k \cdot 1{,}5^2$$
$$= \frac{1}{3} \cdot 1{,}5^{k+2} \quad //$$

Suchalgorithmen

Lineare Suche

```java
int linearSearch(int[] A, int key) {
    for(int i=0; i<A.length; i++) {
        if(key == A[i]) return i;
    }
    return -1; // key not found
}
```

Binäre Suche

Iterativ

```java
int search(int[] A, int key) {
    int l = 0;
    int r = A.length-1;
    int m;
    while(l<=r) {
        m = (l+r)/2;
        if(key == A[m]) return m;
        if(key < A[m]) r = m-1;
        else l = m+1;
    }
    return -1; // key not found
}
```

Rekursiv

```java
int search(int[] A, int key, int l, int r) {
    int m = (l+r)/2;
    if(l==r) return -1; // key not found
    if(key == A[m]) return m;
    if(key > A[m]) {
        return search(A, key, m+1, r);
    } else {
        return search(A, key, l, m-1);
    }
}
```

$T(1) = c$

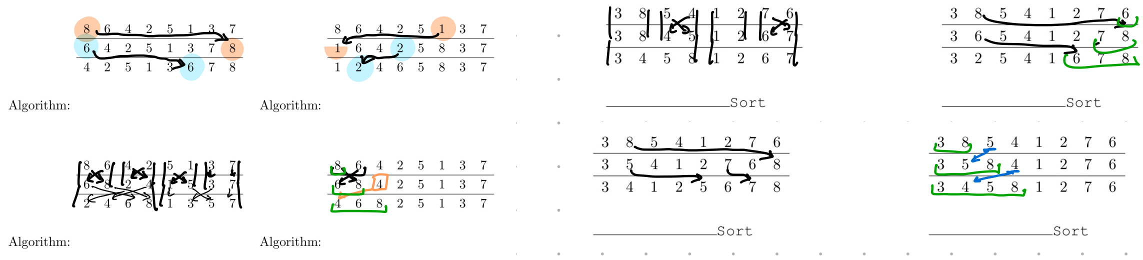
$T(n) = T(n/2) + d$

$n = 2^k \iff \log_2(n) = k$

$$T(2^k) = T(2^{k-1}) + d = T(2^{k-2}) + 2d = T(2^{k-3}) + 3d = \dots = T(1) + k \cdot d$$
$$= c + k \cdot d$$
$$= c + \log_2(n) \cdot d$$
$$\in O(\log(n))$$

Aufgabe Sortieralgorithmen (FS20 T1 e)

Insertion Sort, Selection Sort, Merge Sort, Bubble Sort



Aufgabe Invarianten (FS21 T1 b)

Let $A[0,\ldots,n-1]$ be an integer array of size n. Consider the following implementation of insertion sort:

Algorithm 1 `InsertionSort`(A)

for $i = 1 \ldots n-1$ **do**
  $B \leftarrow A[i]$
  Find the smallest index $j \in \{0,\ldots,i\}$ such that $A[i] \leq A[j]$.
  Shift the subarray $A[j,\ldots,i-1]$ by one to the right, and move the element B to position j.

Consider the following invariant $INV(i)$: After the ith iteration, $A[0,\ldots,i]$ is sorted.

For each of the following claims, state whether it is true or false. You get 1P for a correct answer, -1P for a wrong answer and 0P for a missing answer. In total, you get at least 0 points.

Claim	**true**	**false**
$INV(i)$ holds after the ith iteration of the `for`-loop.	☒	☐
$INV(i)$ can be used to prove the correctness of `InsertionSort`.	☒	☐
$INV(1)$ already holds before the first loop iteration is executed.	☐	☒
At the start of the ith loop iteration, $A[0,\ldots,i-1]$ is sorted. Further, for the smallest index $j \in \{0,\ldots,i\}$ that satisfies $A[i] \leq A[j]$, the following holds: All elements in $A[0,\ldots,j-1]$ are less than $A[i]$ and all elements in $A[j,\ldots,i-1]$ are greater than or equal to $A[i]$. Thus, shifting $A[j,\ldots,i-1]$ by one to the right and moving B to position j yields a sorted subarray $A[0,\ldots,i]$.	☒	☐
After the $(n-1)$th loop iteration $INV(n-1)$ holds and per definition this implies that $A[0,\ldots,n-1]$ is sorted.	☒	☐

Invarianten

i) zeige $INV(1)$

ii) zeige $INV(i) \rightarrow INV(i+1)$

iii) zeige $INV(n) \rightarrow$ Korrektheit

Mehr zu Invarianten:

- Aufgaben Woche 4 HS21, HS22
- FS20 T3

Master Theorem

Master theorem. The following theorem is very useful for running-time analysis of divide-and-conquer algorithms.

Theorem 1 (master theorem). *Let $a, C > 0$ and $b \geq 0$ be constants and $T : \mathbb{N} \to \mathbb{R}^+$ a function such that for all even $n \in \mathbb{N}$,*

$$T(n) \leq aT(n/2) + Cn^b. \quad (1)$$

Then for all $n = 2^k$, $k \in \mathbb{N}$,

- *If $b > \log_2 a$, $T(n) \leq O(n^b)$.*
- *If $b = \log_2 a$, $T(n) \leq O(n^{\log_2 a} \cdot \log n)$.*[1]
- *If $b < \log_2 a$, $T(n) \leq O(n^{\log_2 a})$.*

If the function T is increasing, then the condition $n = 2^k$ can be dropped. If (1) holds with "=", then we may replace O with Θ in the conclusion.

This generalizes some results that you have already seen in this course. For example, the (worst-case) running time of Karatsuba algorithm satisfies $T(n) \leq 3T(n/2) + 100n$, so $a = 3$ and $b = 1 < \log_2 3$, hence $T(n) \leq O(n^{\log_2 3})$. Another example is binary search: its running time satisfies $T(n) \leq T(n/2) + 100$, so $a = 1$ and $b = 0 = \log_2 1$, hence $T(n) \leq O(\log n)$.

1) $T(n) \leq \underbrace{4}_{a} \cdot T(\frac{n}{2}) + \underbrace{100}_{c} n^{b=1}$ $\quad T(n) \leq O(n^2)$

$\log_2(a) = 2$

2) $T(n) = T(n/2) + \frac{3}{2}n$ $\quad T(n) \leq O(n)$

$a = 1 \quad b = 1 \quad c = 3/2$

$\log_2 a = 0$

3) $T(n) = 4 \cdot T(n/2) + \frac{7}{2}n^2$ $\quad T(n) \leq O(n^2 \cdot \log n)$

$a = 4 \quad b = 2 \quad c = \frac{7}{2}$

$\log_2 a = 2$

```
func g(N):
if(N>1) {
    for(i=0 to 2) {
        g(N/2)
        g(N/2)
        for(j=0 to N) {
            f()
        }
    }
} else {
    f()
}
```

$T(n) = 4 \cdot T(\frac{n}{2}) + 2n$

$T(1) = 1$

$a = 4, \log_2 a = 2$

$b = 1$

$c = 2$

$T(n) \leq O(n^2)$

Peer Grading Ex. 3.1

(in allen Gruppen)