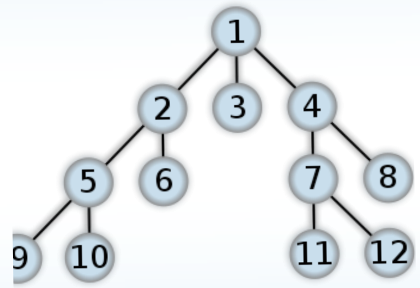


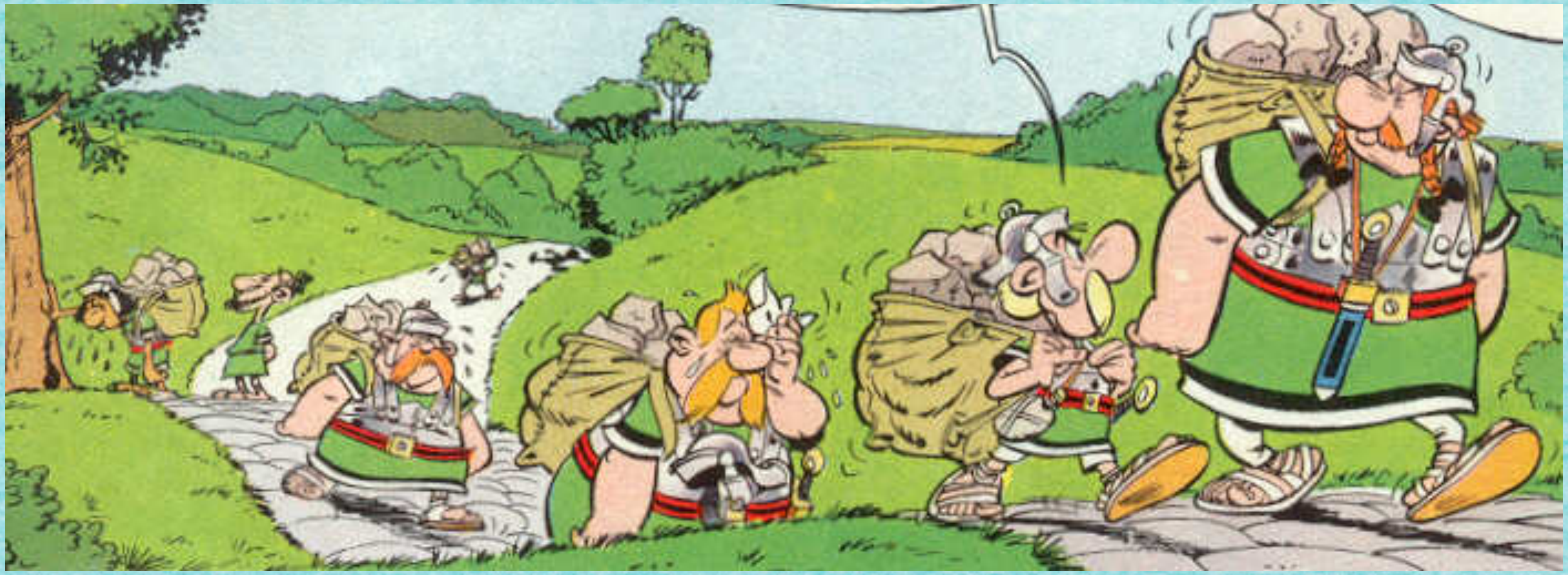
Kapitel 3.7: Wachstum von Funktionen

Algorithmen und Datenstrukturen
WS 2023/24

Prof. Dr. Sándor Fekete

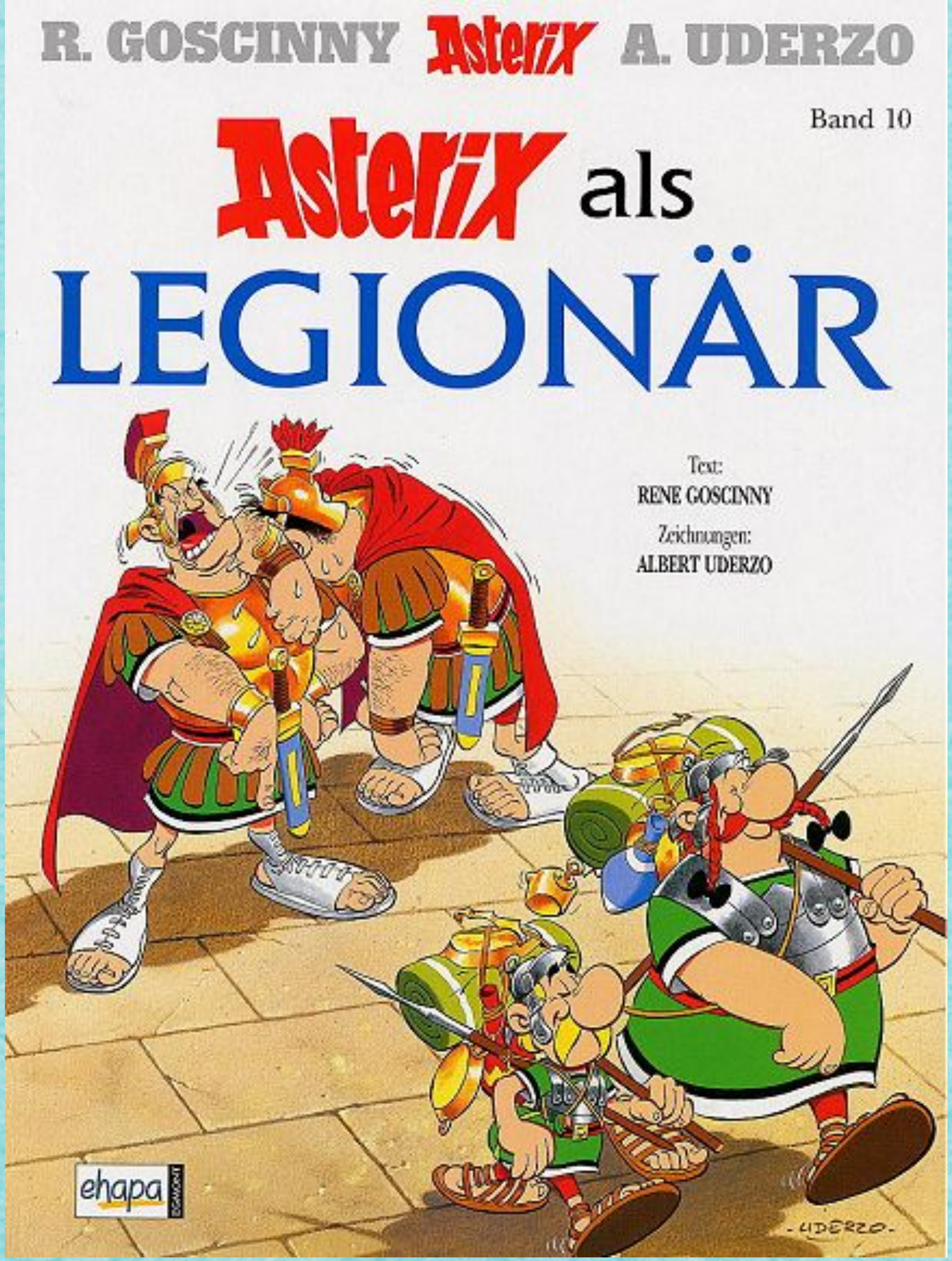
Konzentration

Zwischenbemerkung!



Grundlagentraining

Zwischenbemerkung!

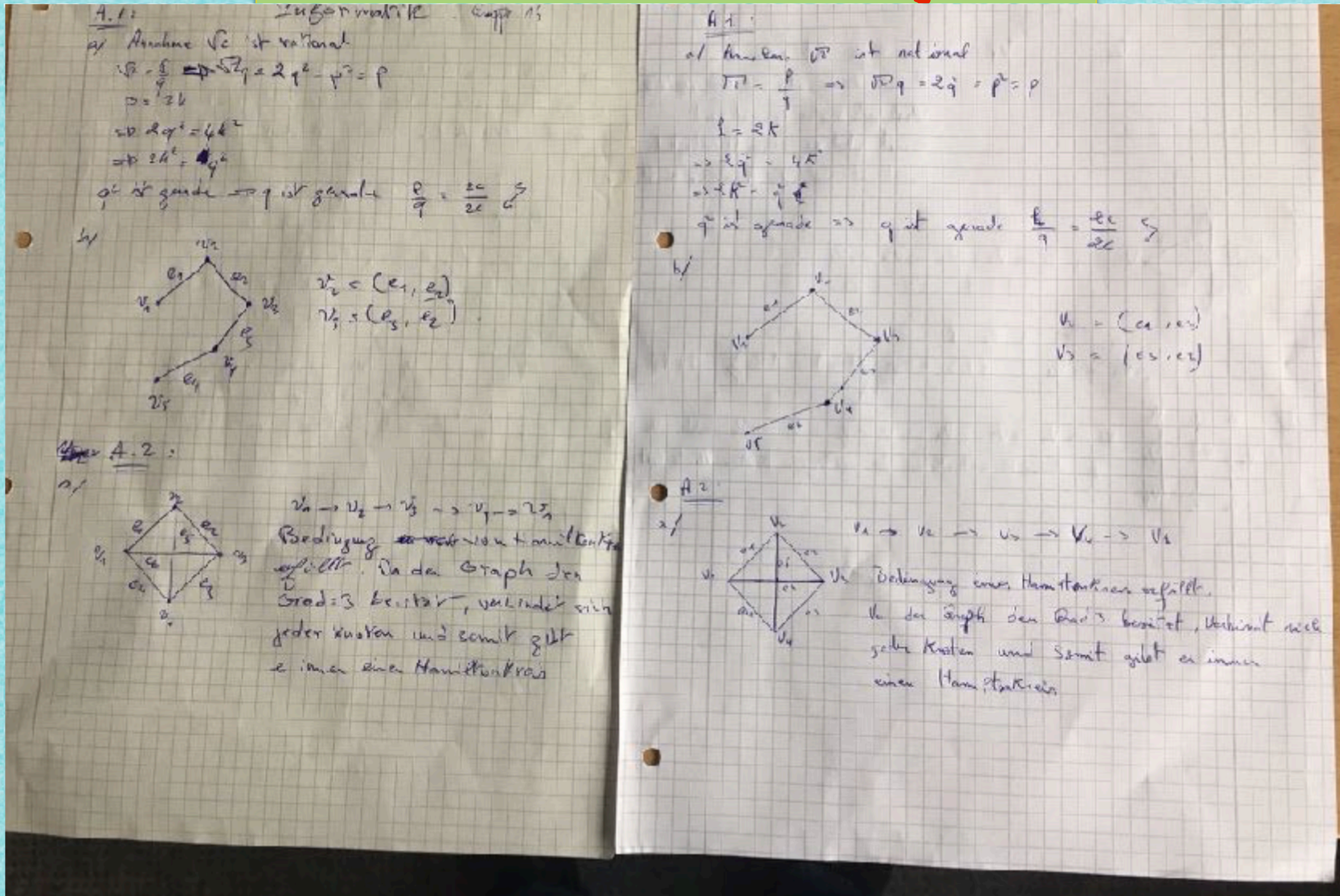


Zwischenbemerkung!

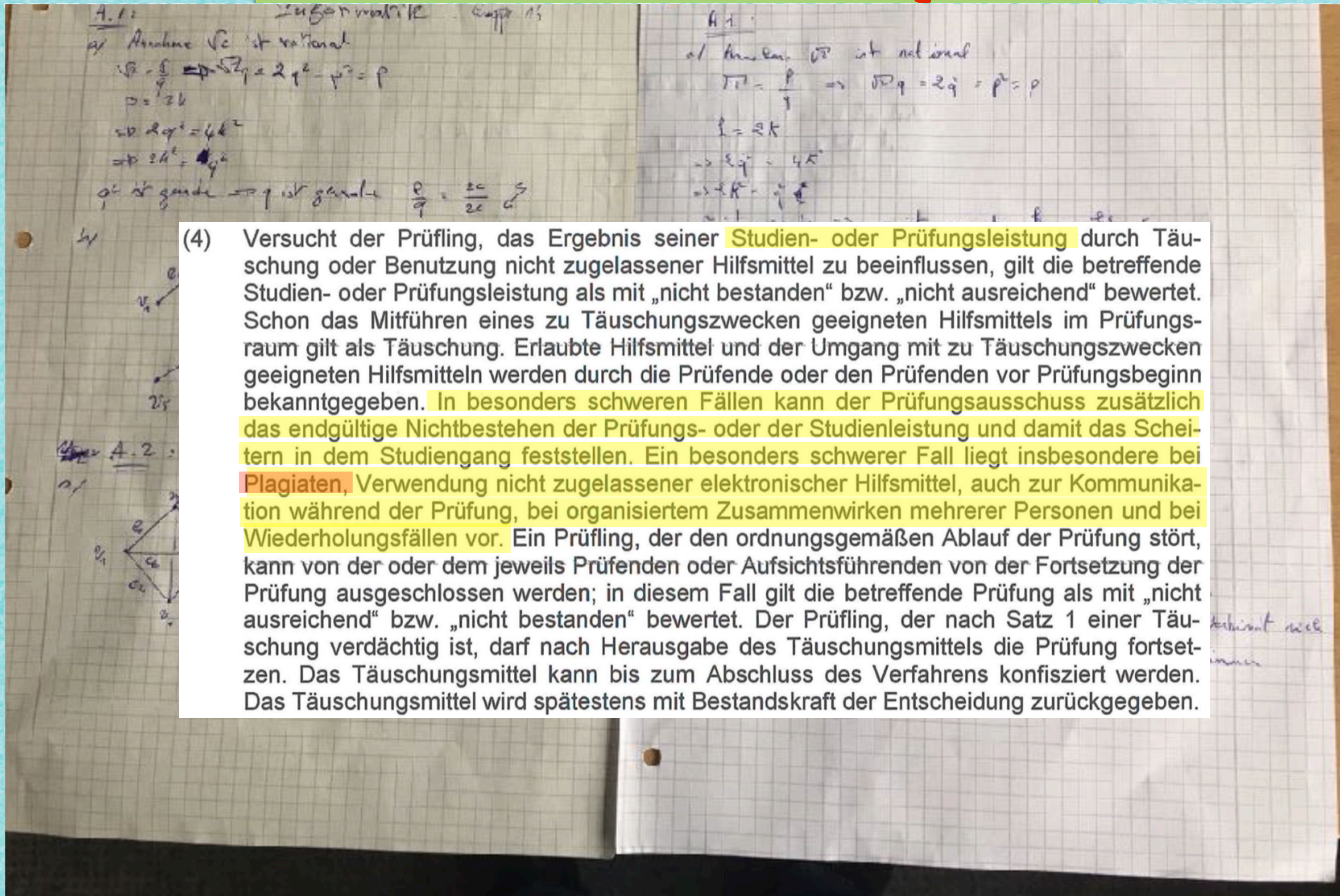
Zwischenbemerkung!

Guttenbergix, Erfinder der Buchkopierkunst

Zwischenbemerkung!

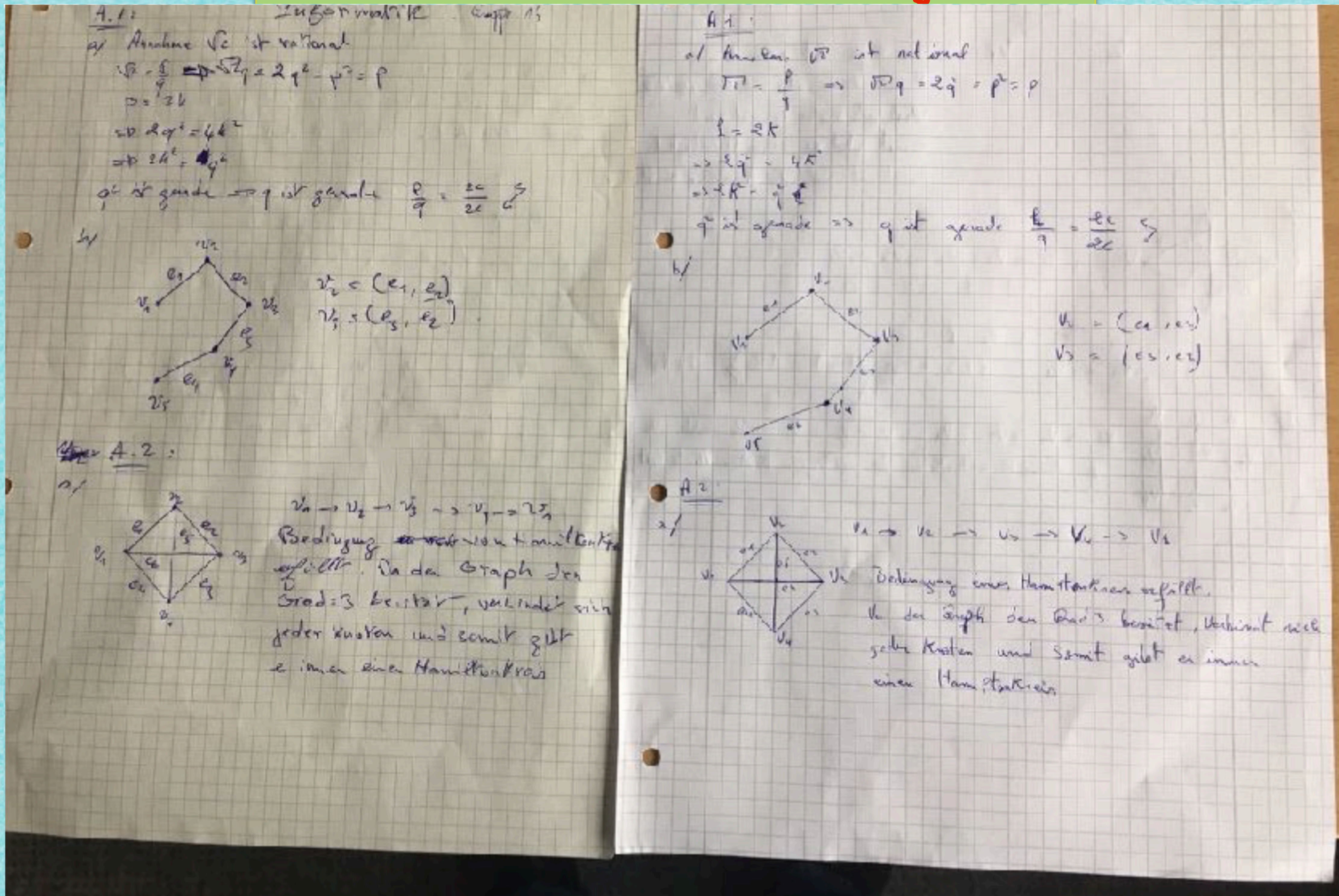


Zwischenbemerkung!

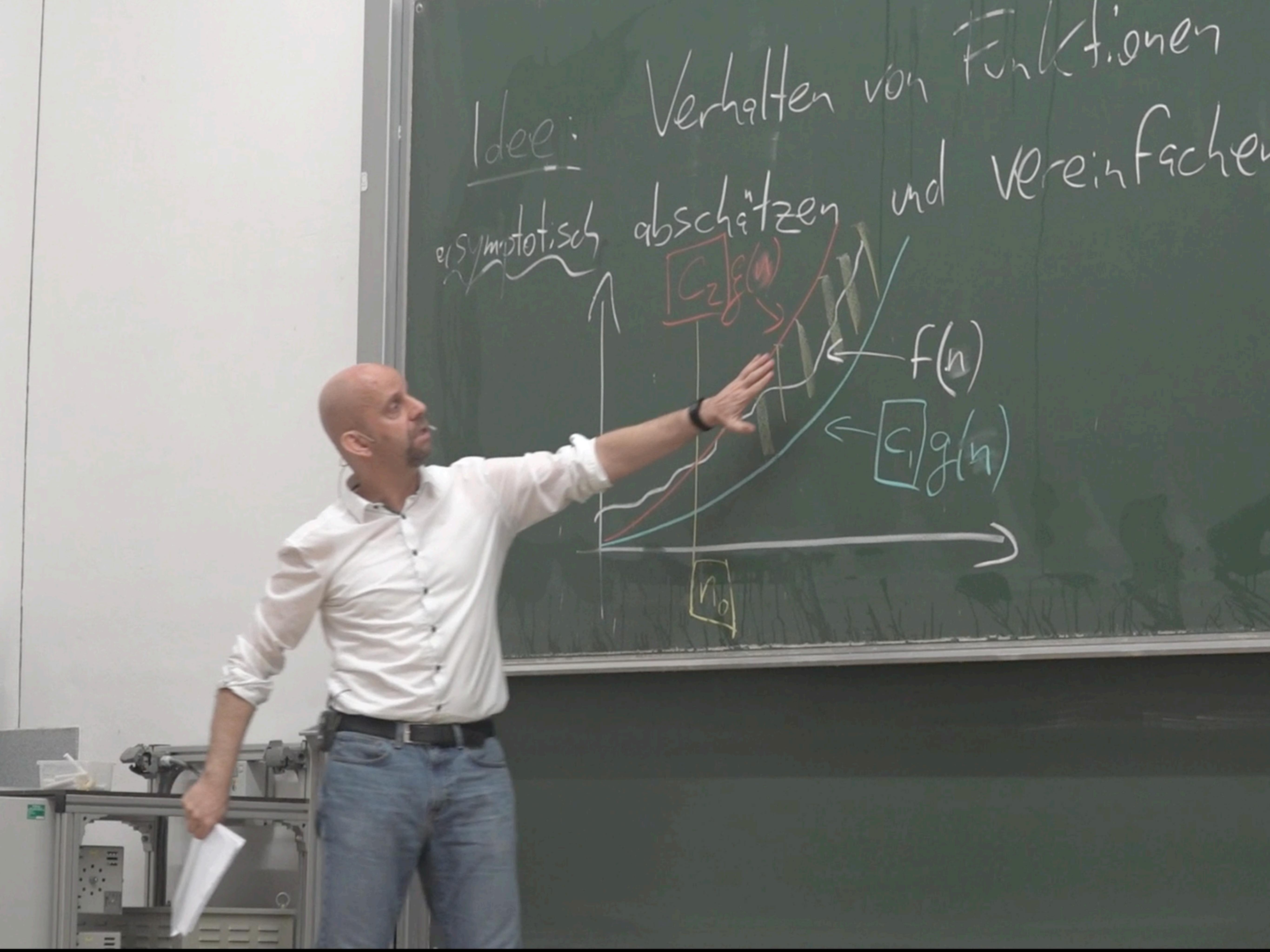


(4) Versucht der Prüfling, das Ergebnis seiner Studien- oder Prüfungsleistung durch Täuschung oder Benutzung nicht zugelassener Hilfsmittel zu beeinflussen, gilt die betreffende Studien- oder Prüfungsleistung als mit „nicht bestanden“ bzw. „nicht ausreichend“ bewertet. Schon das Mitführen eines zu Täuschungszwecken geeigneten Hilfsmittels im Prüfungsraum gilt als Täuschung. Erlaubte Hilfsmittel und der Umgang mit zu Täuschungszwecken geeigneten Hilfsmitteln werden durch die Prüfende oder den Prüfenden vor Prüfungsbeginn bekanntgegeben. In besonders schweren Fällen kann der Prüfungsausschuss zusätzlich das endgültige Nichtbestehen der Prüfungs- oder der Studienleistung und damit das Scheitern in dem Studiengang feststellen. Ein besonders schwerer Fall liegt insbesondere bei Plagiaten, Verwendung nicht zugelassener elektronischer Hilfsmittel, auch zur Kommunikation während der Prüfung, bei organisiertem Zusammenwirken mehrerer Personen und bei Wiederholungsfällen vor. Ein Prüfling, der den ordnungsgemäßen Ablauf der Prüfung stört, kann von der oder dem jeweils Prüfenden oder Aufsichtsführenden von der Fortsetzung der Prüfung ausgeschlossen werden; in diesem Fall gilt die betreffende Prüfung als mit „nicht ausreichend“ bzw. „nicht bestanden“ bewertet. Der Prüfling, der nach Satz 1 einer Täuschung verdächtig ist, darf nach Herausgabe des Täuschungsmittels die Prüfung fortsetzen. Das Täuschungsmittel kann bis zum Abschluss des Verfahrens konfisziert werden. Das Täuschungsmittel wird spätestens mit Bestandskraft der Entscheidung zurückgegeben.

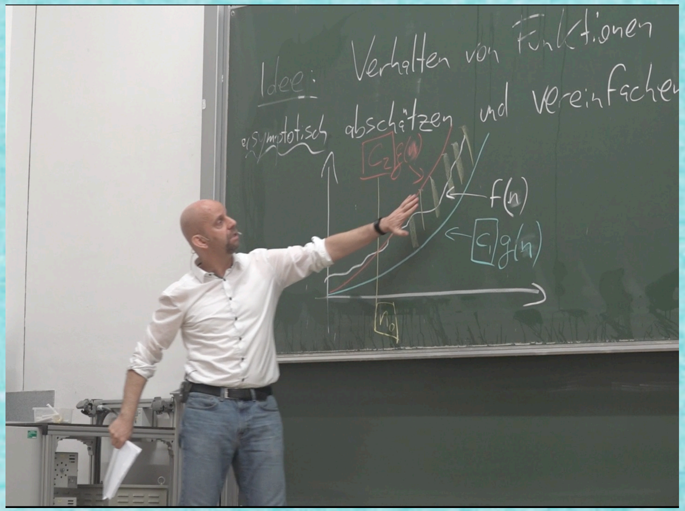
Zwischenbemerkung!



Konzentration


Idee: Verhalten von Funktionen
und vereinfache
asymptotisch abschätzen
$c_2 g(n)$
$f(n)$
$c_1 g(n)$
n_0

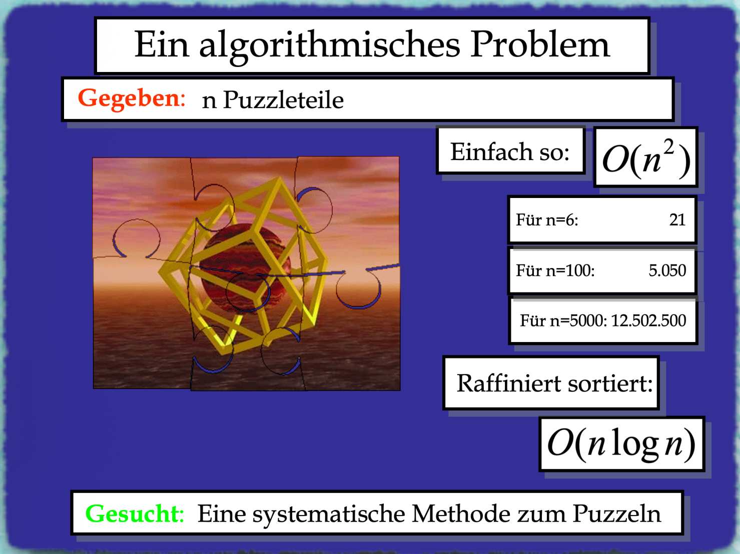
Wofür ist das gut?

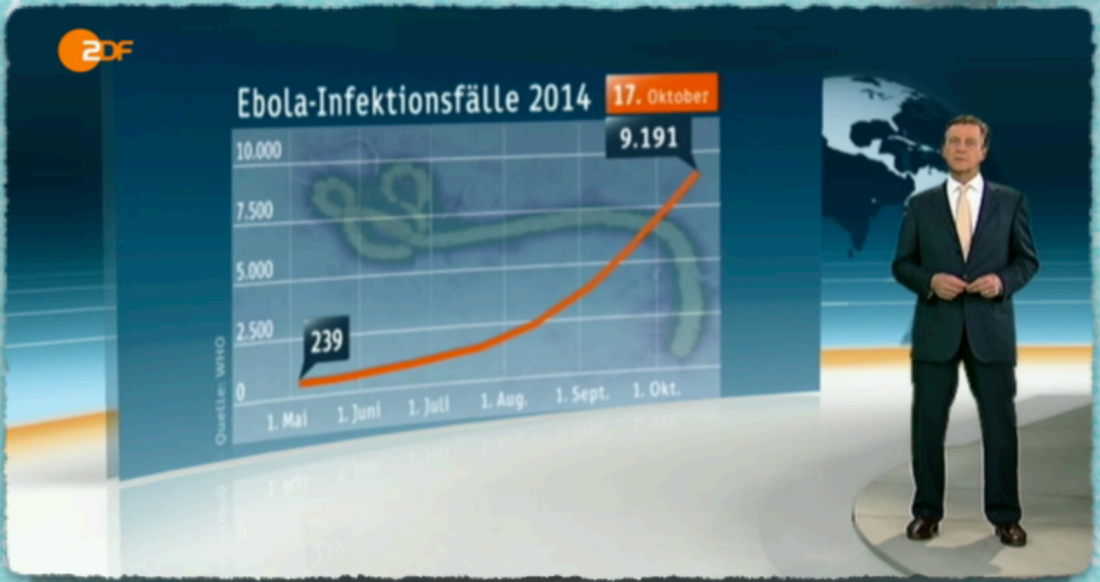


$$\log_2\lfloor(2n + 4m + n(\lfloor\log_2 n\rfloor + 1) + 2m(\lfloor\log_2 n\rfloor + 1))\rfloor + 1$$

$$\Theta(m \log n)$$

Einfach so:	$O(n^2)$
Für n=6:	21
Für n=100:	5.050
Für n=5000:	12.502.500



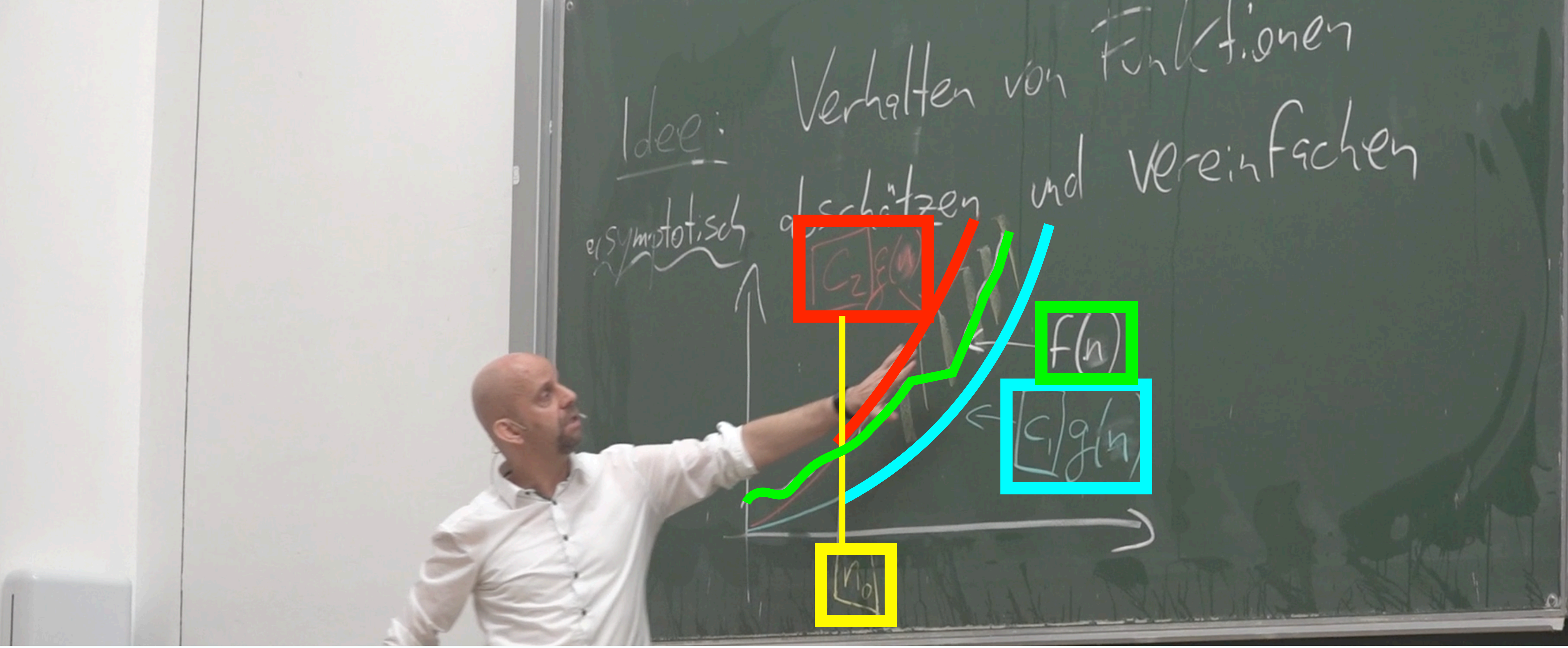


$$\Theta(n^2)$$

$$(2n)^2 = 4 \cdot n^2$$

$$\Theta(2^n)$$

$$2^{2n} = 2^n \cdot 2^n$$



DEFINITION 3.9 (Θ-Notation)

Seien $f, g : \mathbb{N} \to \mathbb{R}$ Funktionen.

Dann gilt

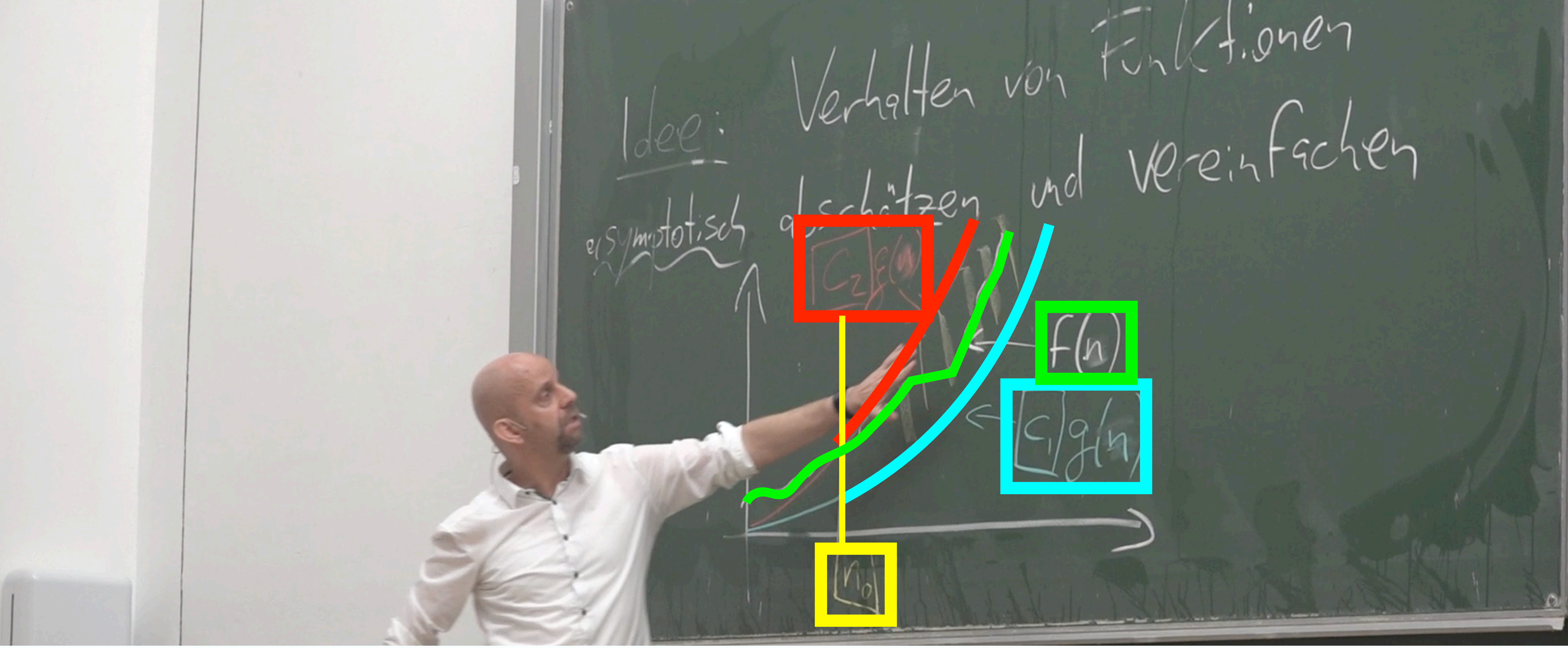
$f \in \Theta(g) :\Leftrightarrow$ Es gibt positive Konstanten c_1, c_2, n_0 mit

$$0 \le c_1 g(n) \le f(n) \le c_2 g(n) \quad \text{für alle } n \ge n_0$$

Man sagt: f wächst asymptotisch in derselben Größenordnung wie g.

Quiz!

s.fekete@tu-bs.de



DEFINITION 3.9 (Θ-Notation)

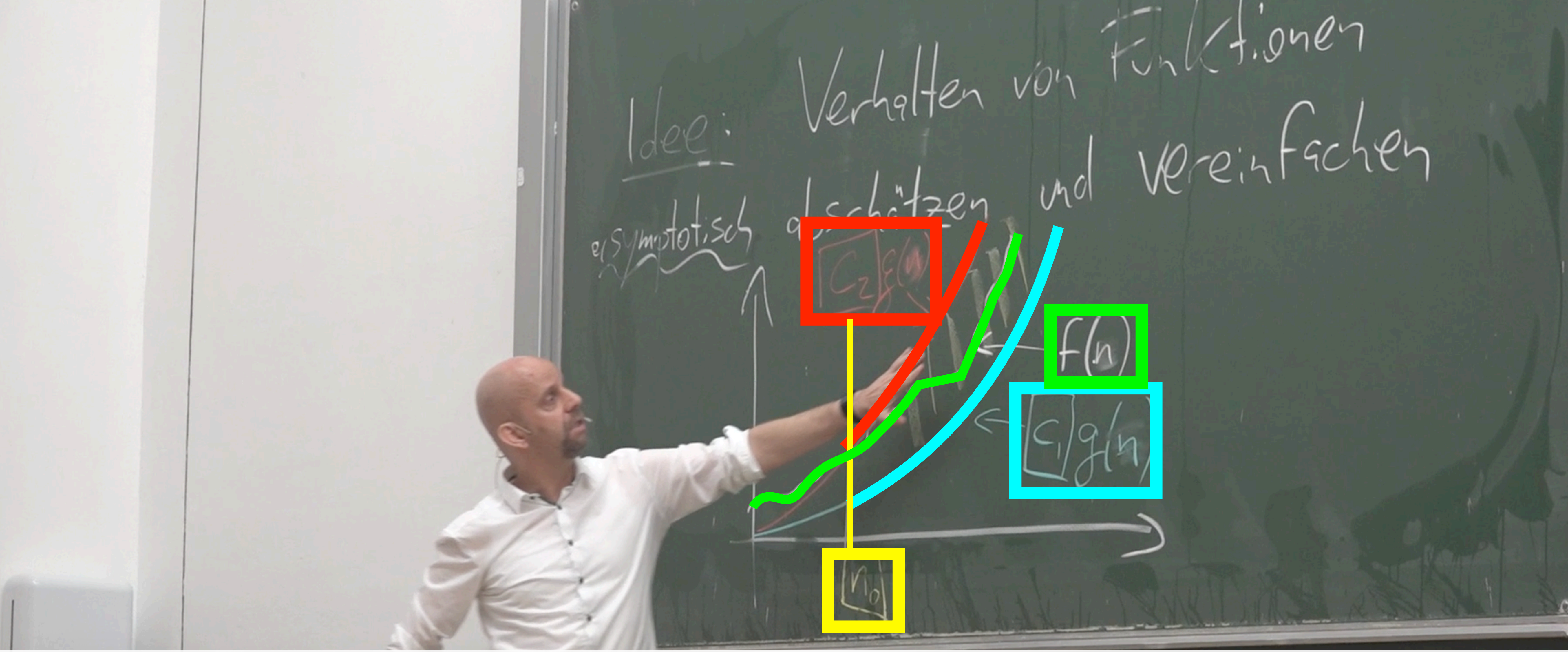
Seien $f, g : \mathbb{N} \to \mathbb{R}$ Funktionen.

Dann gilt

$f \in \Theta(g) :\Leftrightarrow$ Es gibt positive Konstanten $\boxed{c_1}$ $\boxed{c_2}$ $\boxed{n_0}$ mit

$0 \le \boxed{c_1 g(n)} \le \boxed{f(n)} \le \boxed{c_2 g(n)}$ für alle $\boxed{n \ge n_0}$

Man sagt: f wächst asymptotisch in derselben Größenordnung wie g.



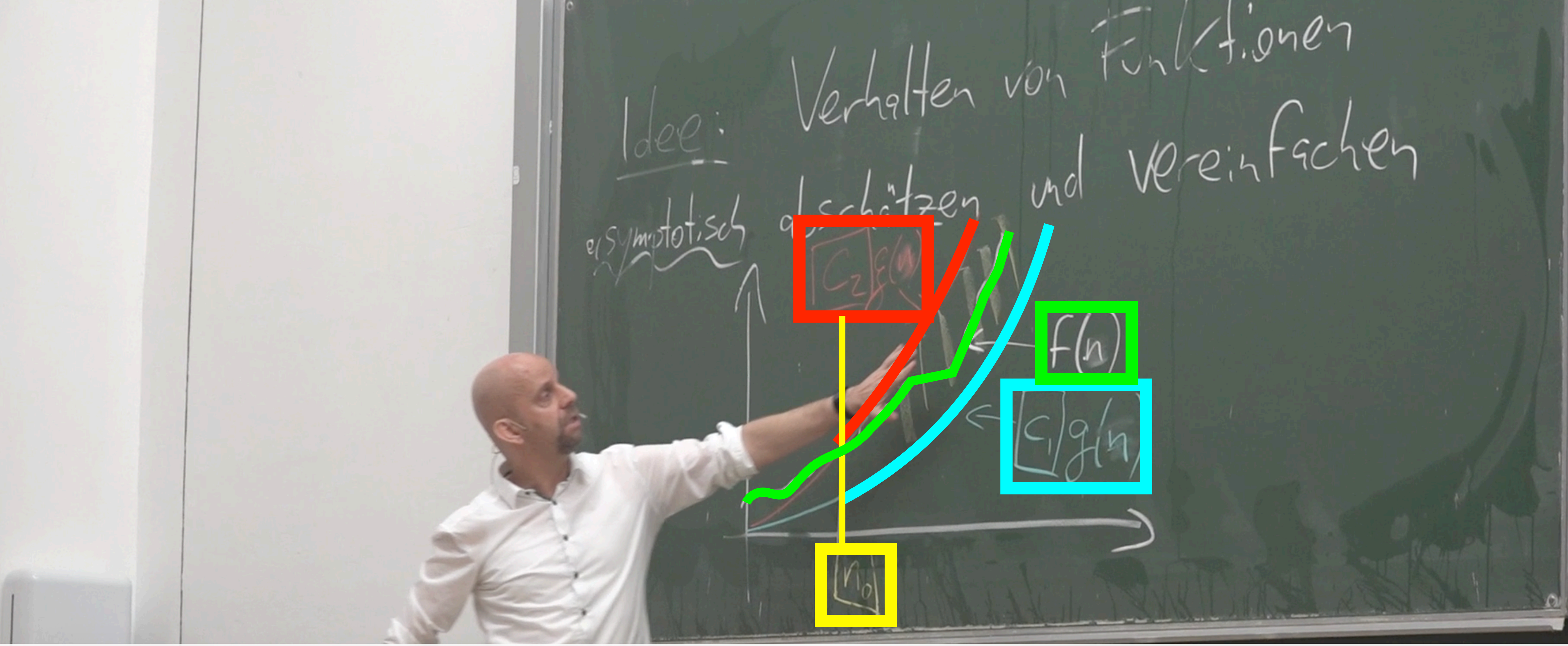
$$\boxed{c_1 g(n)} \leq \boxed{f(n)} \leq \boxed{c_2 g(n)}$$

$$\boxed{c_1} \leq \frac{f(n)}{g(n)} \leq \boxed{c_2}$$

Frage 7:

Die Funktion $f(n) := 5n^6 + 4n^4 + 7n^3 + 27n - 9$ liegt...

- ... nur in $O(n^6)$.
- ... nur in $\Omega(n^6)$.
- ... in $\Theta(n^6)$.



$$\boxed{c_1 g(n)} \le \boxed{f(n)} \le \boxed{c_2 g(n)}$$

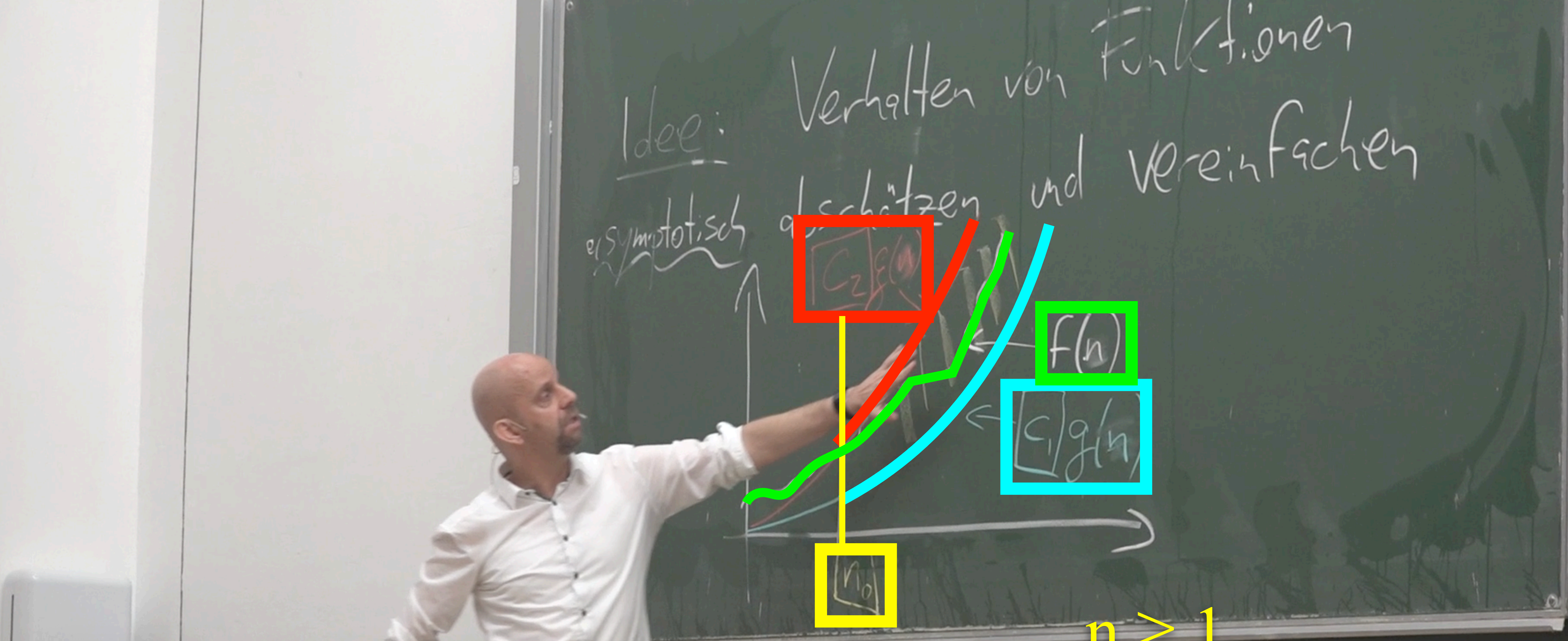
$$\boxed{c_1} \le \frac{f(n)}{g(n)} \le \boxed{c_2}$$

$$\frac{f(n)}{g(n)} = 5 + \frac{4}{n^2} + \frac{7}{n^3} + \frac{27}{n^5} - \frac{9}{n^6}$$

Frage 7:

Die Funktion $f(n) := 5n^6 + 4n^4 + 7n^3 + 27n - 9$ liegt...

- ... nur in $O(n^6)$.
- ... nur in $\Omega(n^6)$.
- ... in $\Theta(n^6)$.



$n \geq 1$

$$c_1 g(n) \leq f(n) \leq c_2 g(n)$$

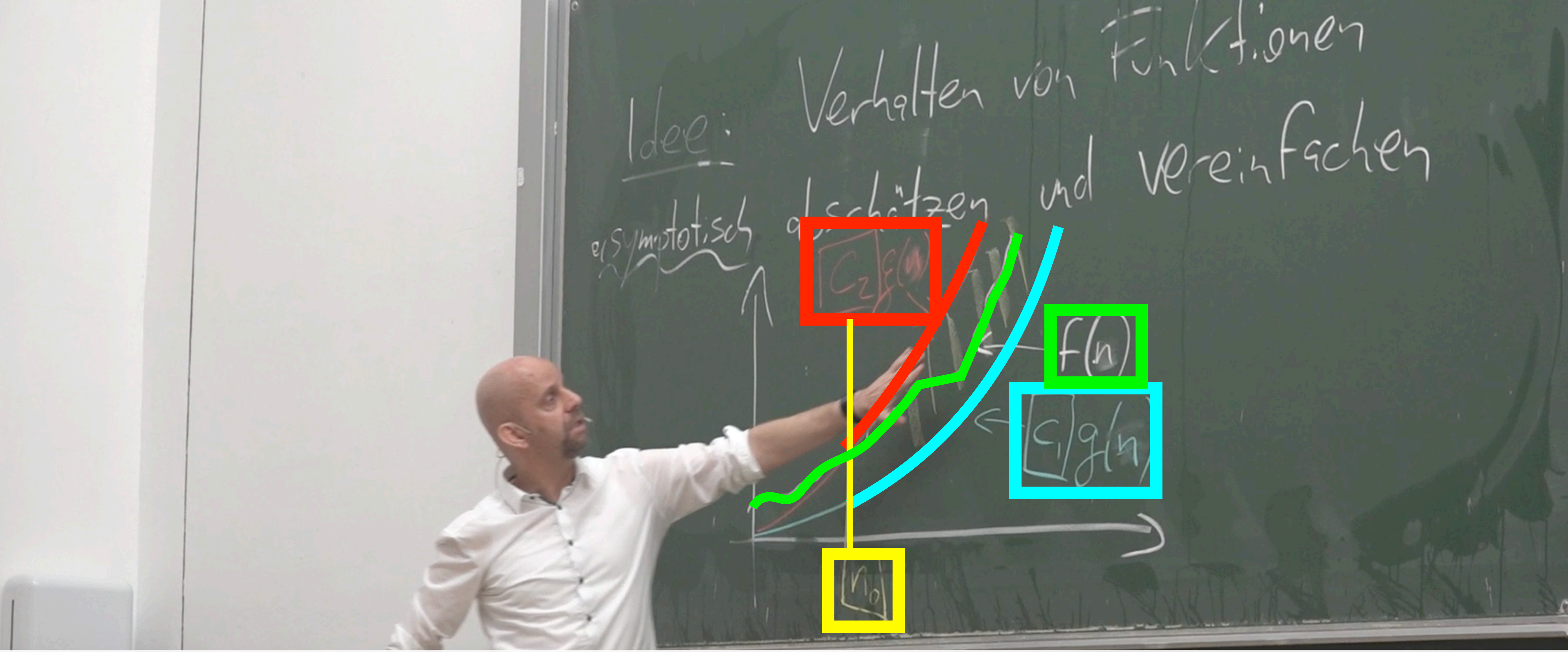
$$c_1 \leq \frac{f(n)}{g(n)} \leq c_2$$

$$5 \leq \frac{f(n)}{g(n)} = 5 + \frac{4}{n^2} + \frac{7}{n^3} + \frac{27}{n^5} - \frac{9}{n^6} \leq 43$$

Frage 7:

Die Funktion $f(n) := 5n^6 + 4n^4 + 7n^3 + 27n - 9$ liegt...

- ... nur in $O(n^6)$.
- ... nur in $\Omega(n^6)$.
- ... in $\Theta(n^6)$. ✓



$$\leq \boxed{c_1 g(n)} \leq \boxed{f(n)} \leq \boxed{c_2 g(n)}$$

$$\boxed{c_1} \leq \frac{f(n)}{g(n)} \leq \boxed{c_2}$$

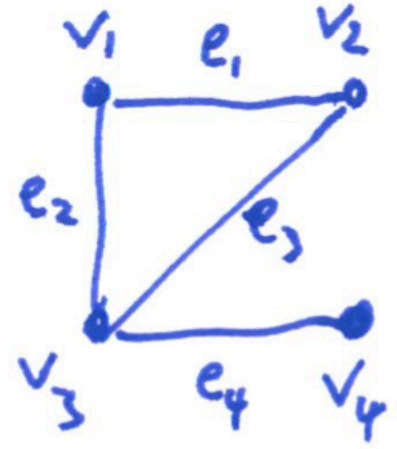
Frage 8:

Die Funktion $f(n) := 2^n$ liegt...

- ... nur in $O(3^n)$. ✓
- ... nur in $\Omega(3^n)$.
- ... in $\Theta(3^n)$.

$$\frac{f(n)}{g(n)} = \frac{2^n}{3^n} = \left(\frac{2}{3}\right)^n$$

(4) Adjazenzliste



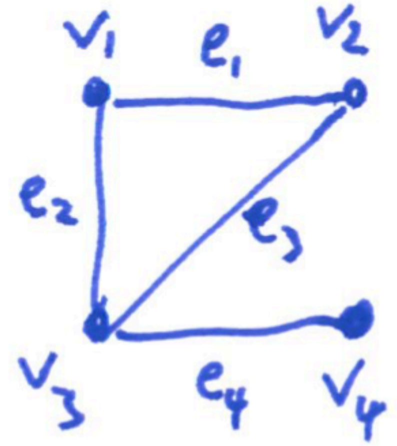
v_1 : v_2 , v_3 ;

v_2 : v_1 , v_3 ;

v_3 : v_1 , v_2 , v_4 ;

v_4 : v_3 ;

(4) Adjazenzliste



$v_1 : v_2, v_3;$

$v_2 : v_1, v_3;$

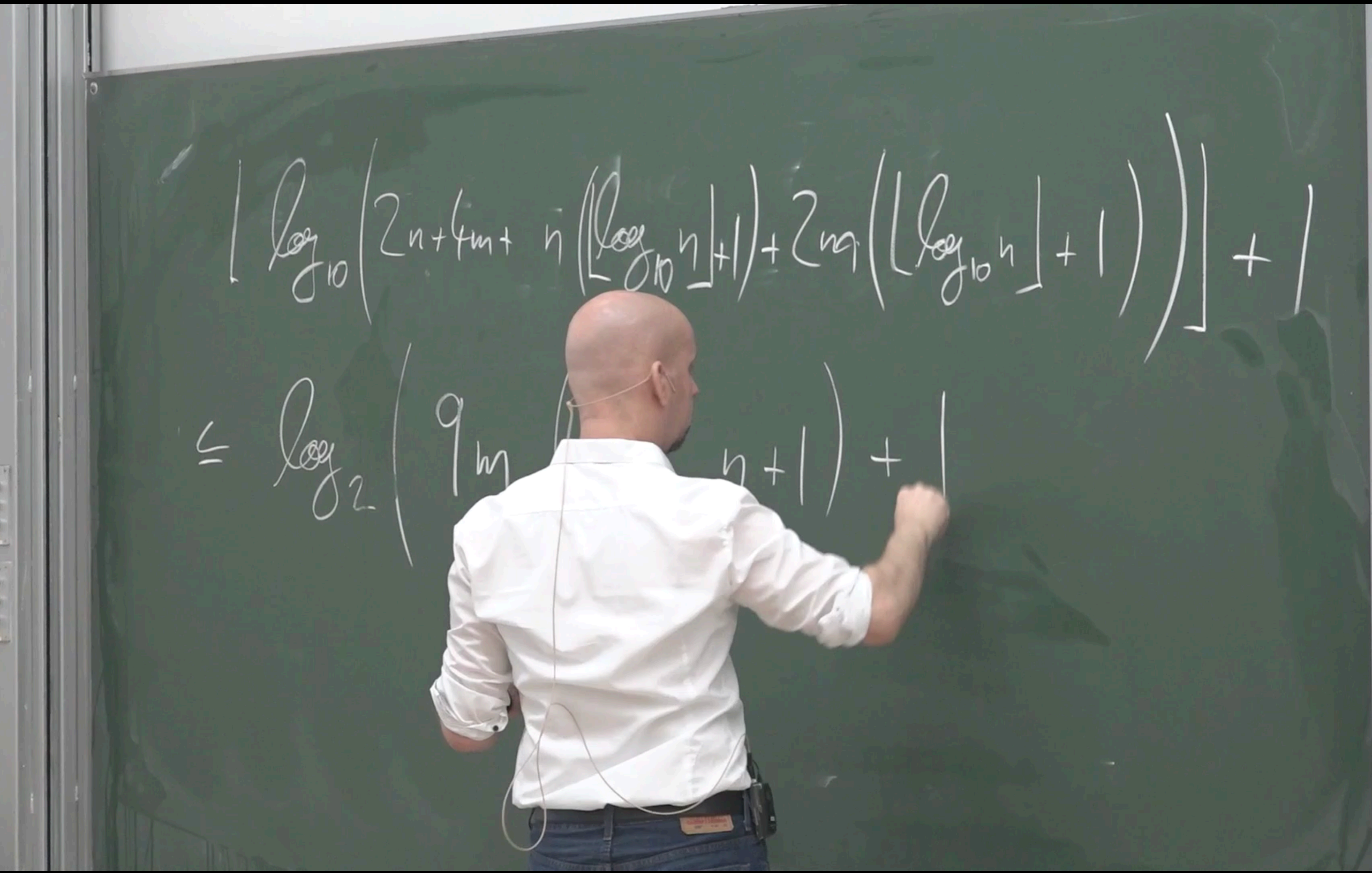
$v_3 : v_1, v_2, v_4;$

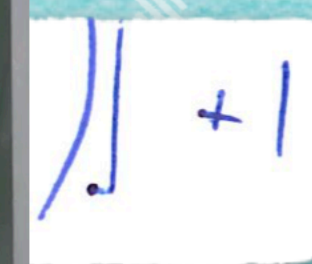
$v_4 : v_3;$

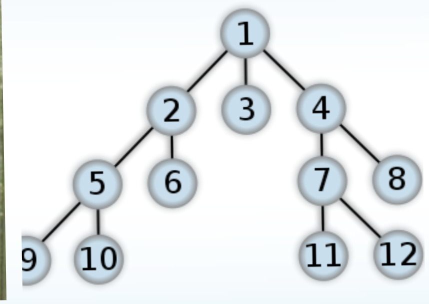
$$\downarrow \qquad \downarrow \qquad \downarrow \qquad \downarrow$$

$$v_2, v_3; \; v_1, v_3; \; v_1, v_2, v_4; \; v_3$$

$$\log_2 \left\lfloor \left(2n + 4m + n\left(\lfloor \log_2 n \rfloor + 1\right) + 2m\left(\lfloor \log_2 n \rfloor + 1\right) \right) \right\rfloor + 1$$


⌊log10(2n+4m+n(⌊log10 n⌋+1)+2m(⌊log10 n⌋+1))⌋ + 1



Kapitel 3.8: Laufzeit von DFS und BFS

Algorithmen und Datenstrukturen
WS 2023/24

Prof. Dr. Sándor Fekete

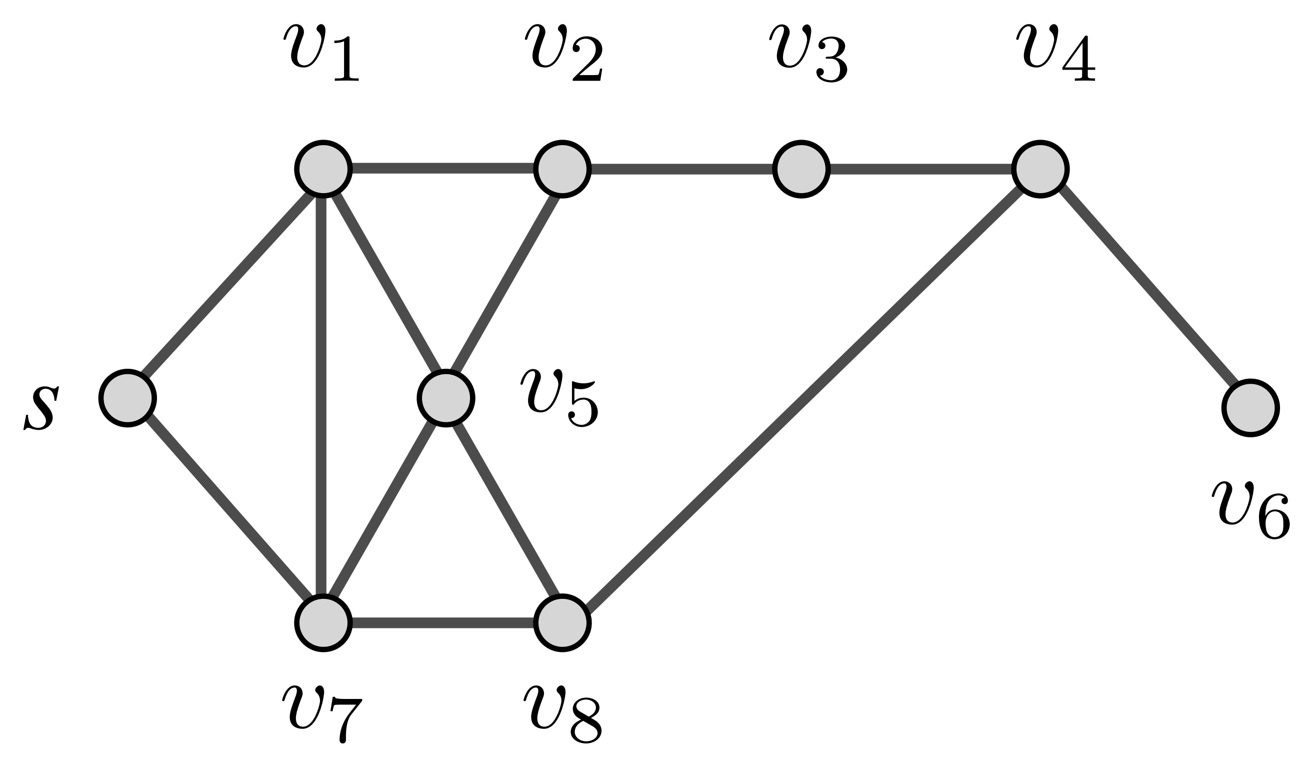
Algorithmus 3.7

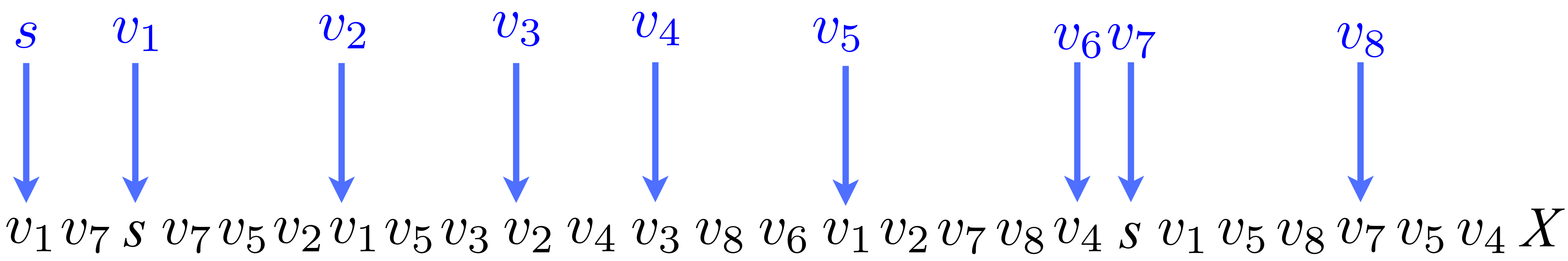
INPUT: Graph G = (V,E), Knoten s

OUTPUT: Knotenmenge Y ⊆ V, die von s aus erreichbar ist,

Kantenmenge T ⊆ E, die die Erreichbarkeit sicherstellt

```
1.  Sei R:={s}, Y:={s}, T:=∅
2.  WHILE (R≠∅) DO {
      2.1. Wähle v∈ R
      2.2. IF (es gibt kein w∈ V\Y mit e={v,w}∈E) THEN
          2.2.1. R:=R\{v}
      2.3. ELSE {
          2.3.1.Wähle ein w∈V\Y mit e={v,w} ∈ E
          2.3.2. Setze R := R u {w}, Y :=  Y u {w}, T := T u {e}
          }
    }
3.  STOP
```



Satz 3.13

Der Graphen-Scan-Algorithmus 3.7 lässt sich so implementieren, dass die Laufzeit $O(n+m)$ ist.

Algorithmus 3.7

INPUT: Graph G = (V,E), Knoten s

OUTPUT: Knotenmenge $Y \subseteq V$, die von s aus erreichbar ist,

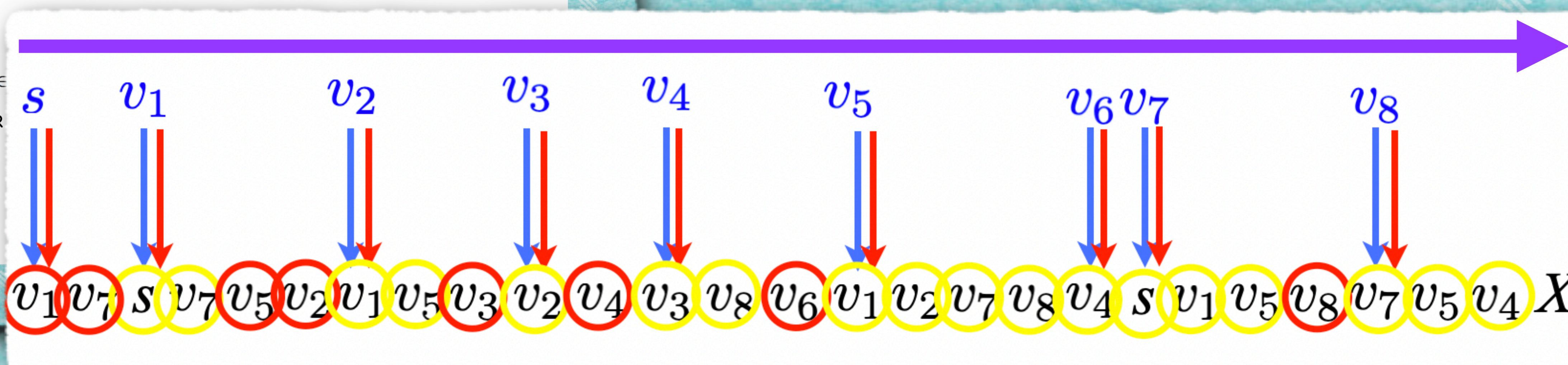
Kantenmenge $T \subseteq E$, die die Erreichbarkeit sicherstellt

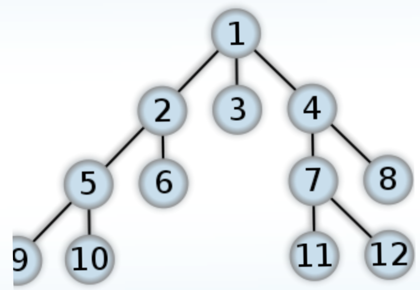
1. Sei $R:=\{s\}$, $Y:=\{s\}$, $T:=\emptyset$
2. WHILE ($R \neq \emptyset$) DO {
   - 2.1. Wähle $v \in R$
   - 2.2. IF (es gibt kein $w \in V \setminus Y$ mit $e=\{v,w\} \in E$) THEN
     - 2.2.1. $R:=R \setminus \{v\}$
   - 2.3. ELSE {
     - 2.3.1. Wähle ein $w \in$
     - 2.3.2. Setze R := R

     }

   }
3. STOP

Adjazenzliste!





Kapitel 3.9: Eigenschaften von DFS und BFS

Algorithmen und Datenstrukturen
WS 2023/24

Prof. Dr. Sándor Fekete

3.9 DFS vs. BFS

Einfach gesagt:

- ***DFS ist eine bestmögliche individuelle Suchstrategie mit lokaler Information.***
- ***BFS ist eine bestmögliche kooperative Suchstrategie mit globaler Information.***

3.9 DFS vs. BFS

Einfach gesagt:

- ***DFS ist eine bestmögliche individuelle Suchstrategie mit lokaler Information.***
- ***BFS ist eine bestmögliche kooperative Suchstrategie mit globaler Information.***

3.9 DFS vs. BFS

Einfach gesagt:

- ***DFS ist eine bestmögliche individuelle Suchstrategie mit lokaler Information.***
- ***BFS ist eine bestmögliche kooperative Suchstrategie mit globaler Information.***

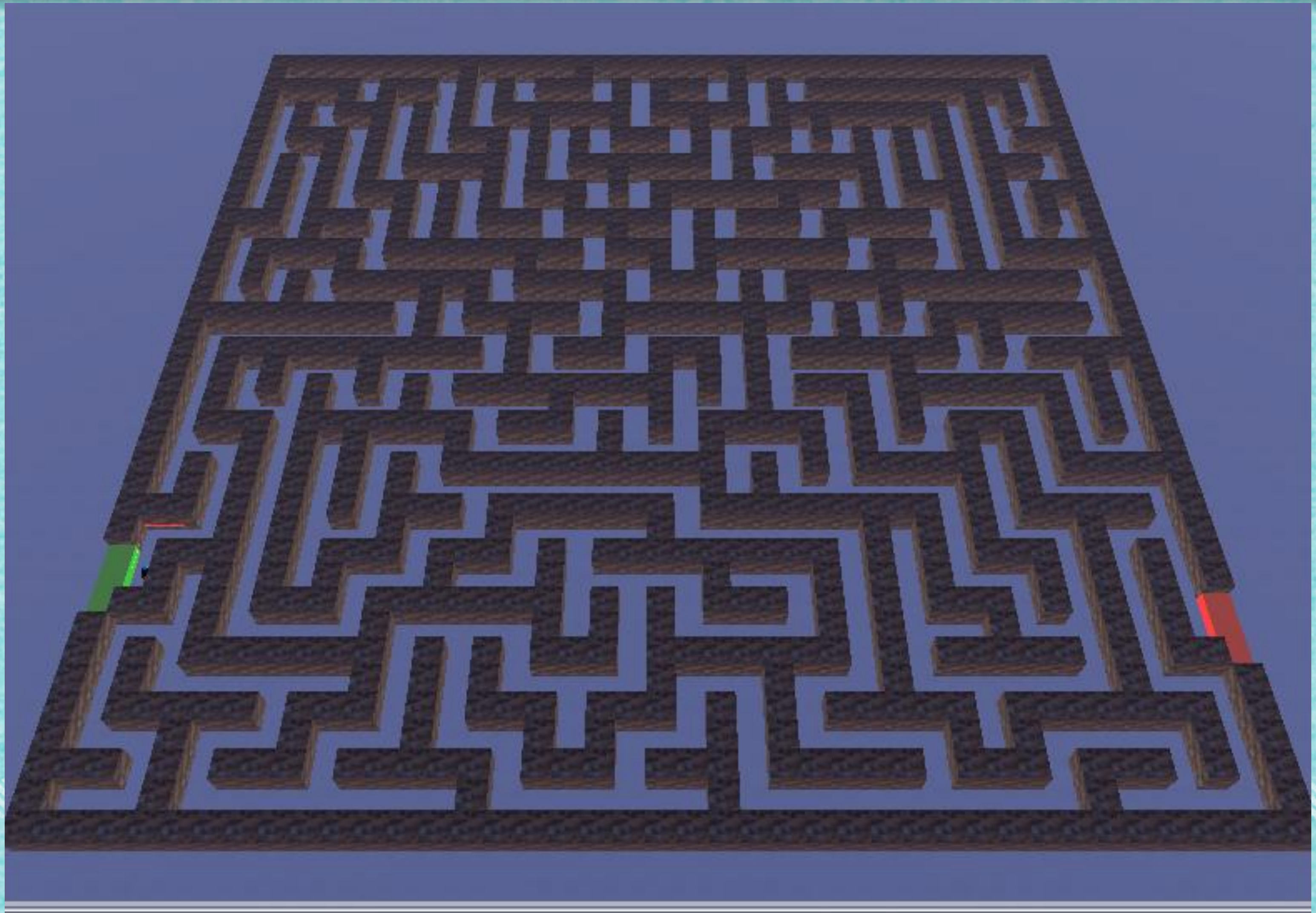
Konkret:

- ***DFS ist gut geeignet für die Suche nach einem Ausweg aus einem Labyrinth.***
- ***BFS ist gut geeignet für die Suche nach kürzesten Wegen in einem Graphen.***

Satz 3.16 (Lokale Suche mit DFS)

(1) DFS findet in jedem zusammenhängenden Graphen mit n Knoten einen Weg der Länge höchstens 2n-1, der alle Knoten besucht.

(2) Für jede lokale Suchstrategie gibt es einen Graphen mit n Knoten, so dass der letzte Knoten erst nach einer Weglänge von 2n-1 besucht wird.



Satz 3.16 (Lokale Suche mit DFS)

(1) DFS findet in jedem zusammenhängenden Graphen mit n Knoten einen Weg der Länge höchstens 2n-1, der alle Knoten besucht.

(2) Für jede lokale Suchstrategie gibt es einen Graphen mit n Knoten, so dass der letzte Knoten erst nach einer Weglänge von 2n-1 besucht wird.

Beweis: ***Übung!***

Algorithmus 3.7

3.9 BFS?

INPUT:	Graph G = (V,E), Knoten s
OUTPUT:	Knotenmenge $Y \subseteq V$, die von s aus erreichbar ist,
	Kantenmenge $T \subseteq E$, die die Erreichbarkeit sicherstellt

1. Sei $R:=\{s\}$, $Y:=\{s\}$, $T:=\emptyset$
2. WHILE ($R \neq \emptyset$) DO {
   - 2.1. wähle Element $v \in R$
   - 2.2. IF (es gibt kein $w \in V \setminus Y$ mit $e=\{v,w\} \in E$) THEN
     - 2.2.1. $R:=R \setminus \{v\}$
   - 2.3. ELSE {
     - 2.3.1. wähle ein $w \in V \setminus R$ mit $e=\{v,w\} \in E$;
     - 2.3.2. setze $R := R \cup \{w\}$, $Y := Y \cup \{w\}$, $T := T \cup \{e\}$;

     }

}

Auf die Schnelle mit der Welle

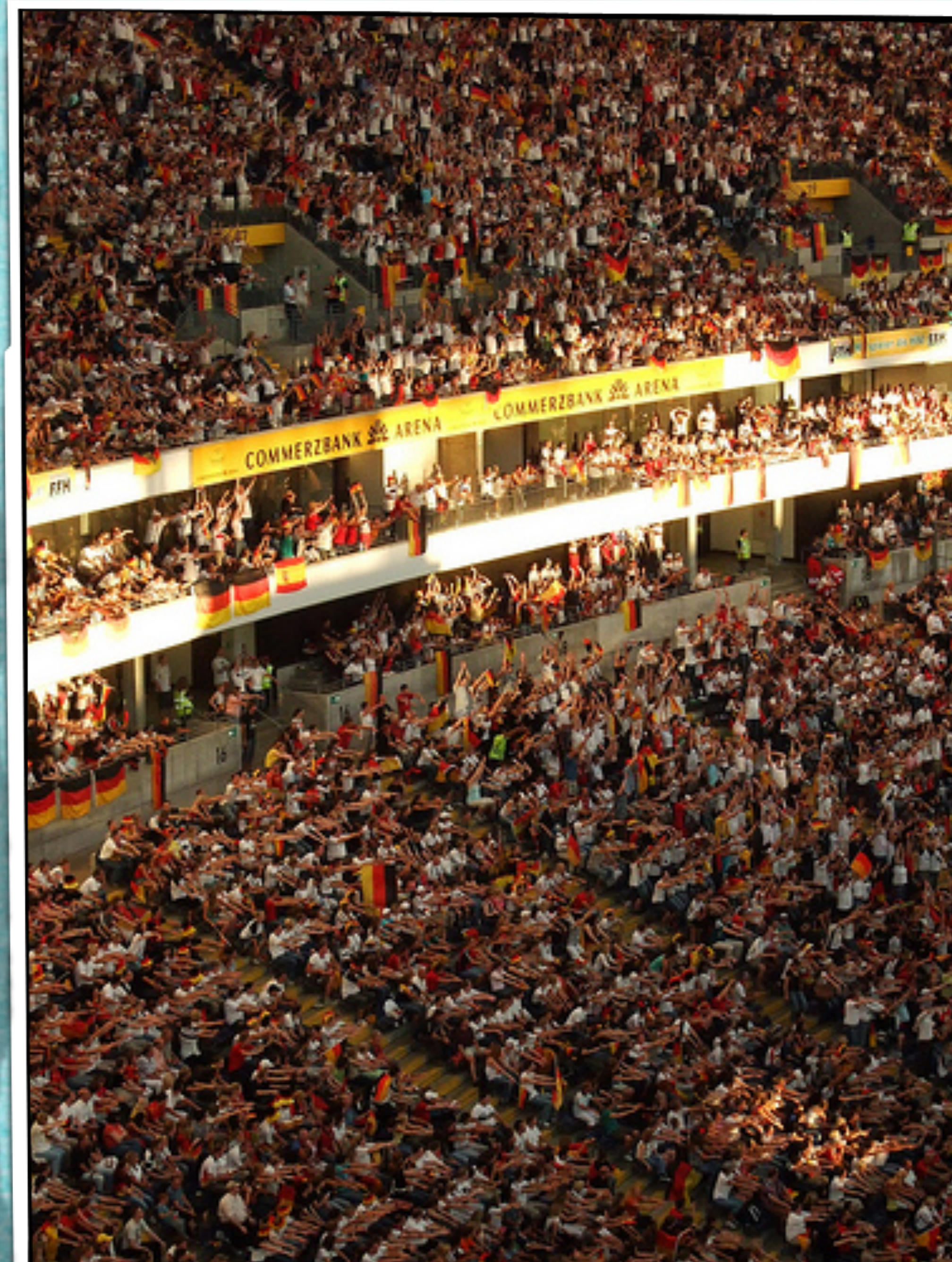
A. LOS bei „NULL“

B. Bis „ANGEKOMMEN!“:

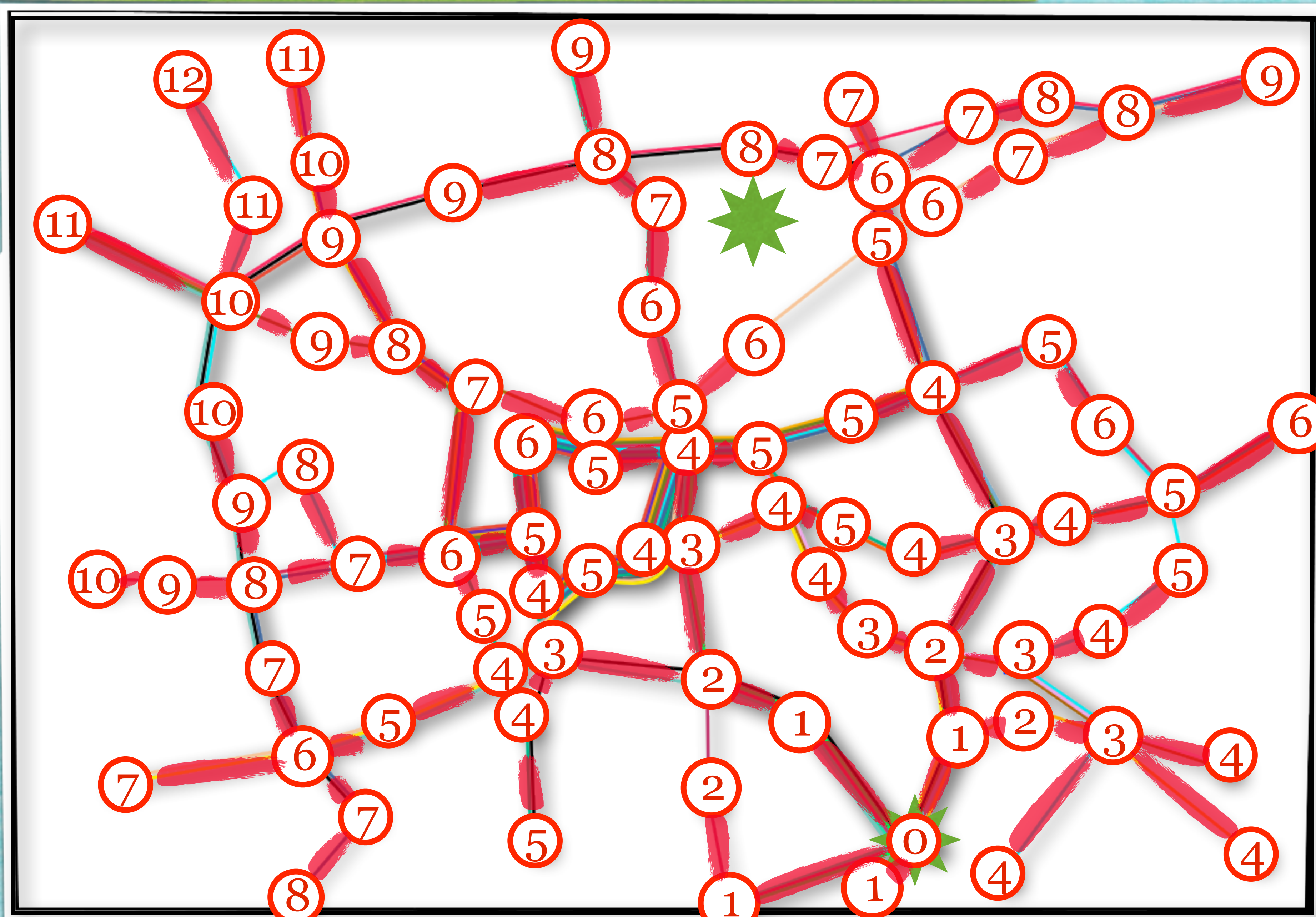
- Solange du noch nicht aufgestanden warst:
  - Wenn ein oder mehrere direkte Nachbarn aufstehen:
    1. Einen dieser Nachbarn merken
    2. In der nächsten Runde:
       2.1. aufstehen
       2.2. Zahl merken
    3. In der übernächsten Runde hinsetzen

C. Nach „ANGEKOMMEN!“:

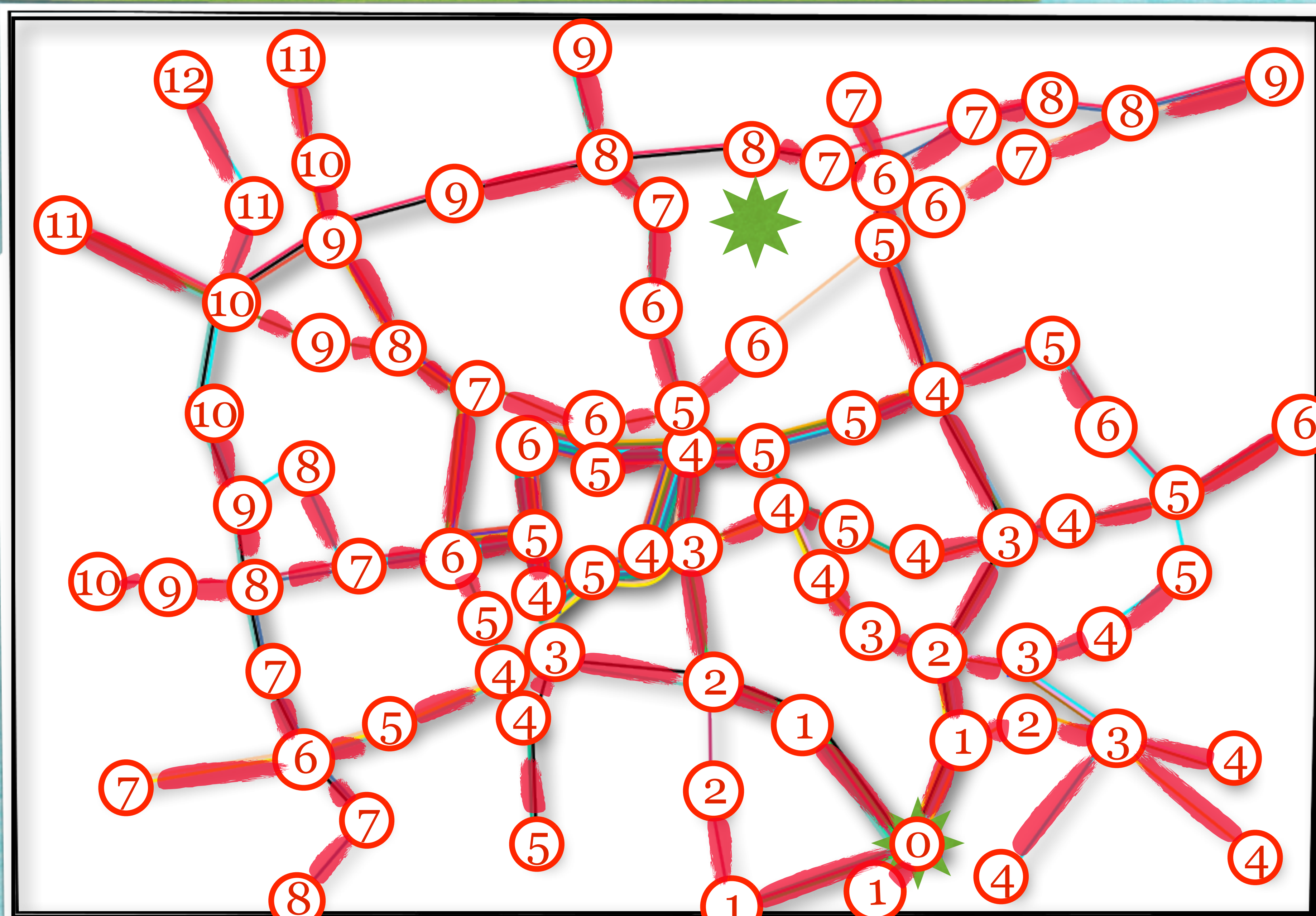
- Auf gemerkten Nachbarn zeigen



Wellenreiten in Graphen

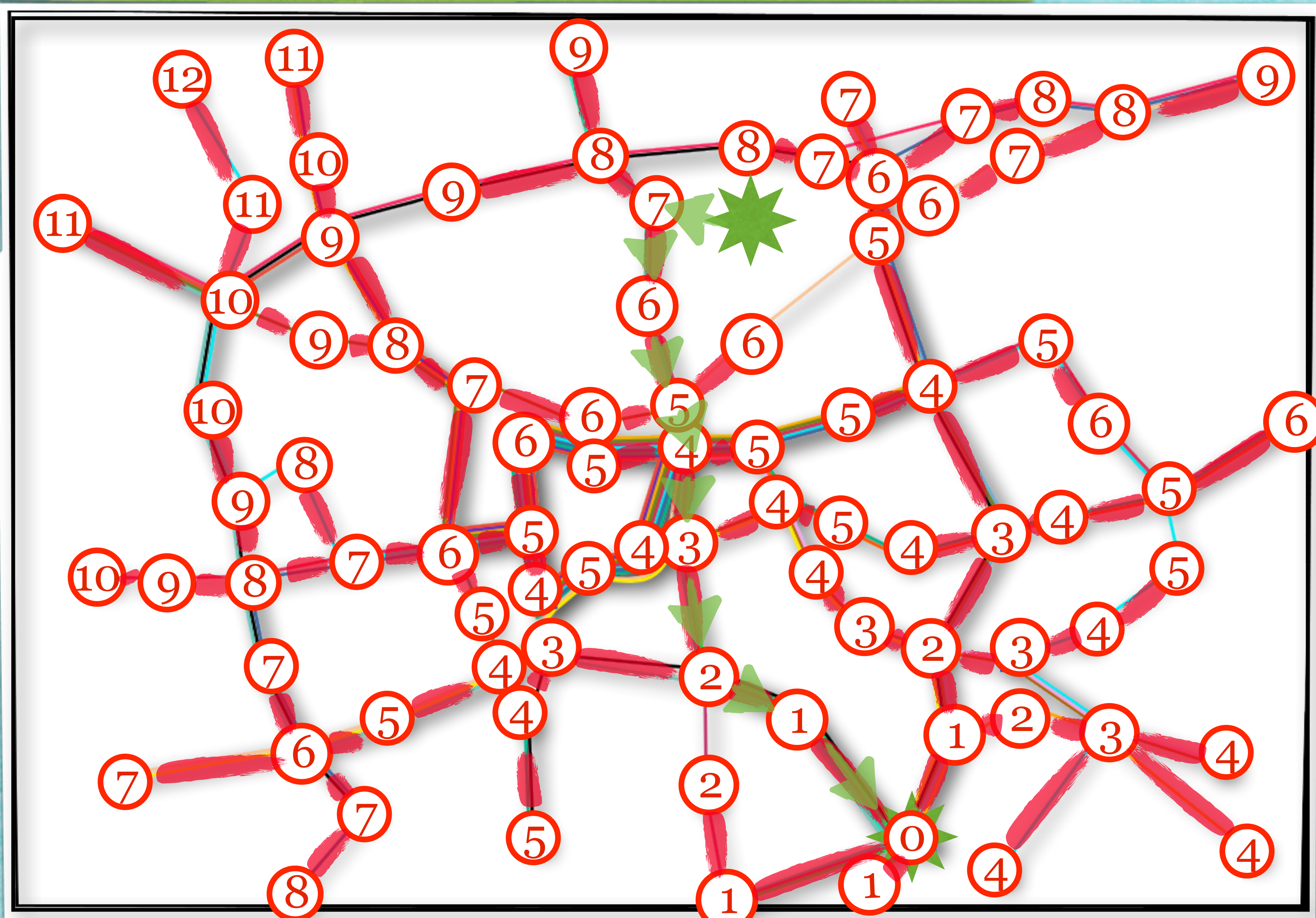


Wellenreiten in Graphen



Breitensuche

Wellenreiten in Graphen



Breitensuche

Algorithmus 3.17

INPUT: Graph $G = (V,E)$, Knoten s

OUTPUT: Knotenmenge $Y \subseteq V$, die von s aus erreichbar ist,

für jeden Knoten $v \in Y$ die Länge $l(v)$ eines kürzesten s-v-Weges,

Kantenmenge $T \subseteq E$, die die Erreichbarkeit sicherstellt

1. Sei $R:=\{s\}$, $Y:=\{s\}$, $T:=\emptyset$, $l(s):=0$

2. WHILE ($R\neq\emptyset$) DO {

    2.1. wähle Element $v \in R$

    2.2. IF (es gibt kein w V\Y mit $e=\{v,w\} \in E$) THEN

        2.2.1. $R:=R\setminus\{v\}$

    2.3. ELSE {

        2.3.1. wähle ein $w \in V\setminus R$ mit $e=\{v,w\} \in E$;

        2.3.2. setze $R := R \cup \{w\}$, $Y := Y \cup \{w\}$, $T := T \cup \{e\}$;

        2.3.3. setze $l(w):=l(v)+1$

        }

}

Satz 3.18

(1) *Das Verfahren 3.17 ist endlich.*

(2) *Die Laufzeit ist O(n+m).*

(3) *Am Ende ist für jeden erreichbaren Knoten v ∈ Y die Länge eines kürzesten Weges von s nach v im Baum (Y,T) durch l(v) gegeben.*

(4) *Am Ende ist für jeden erreichbaren Knoten v ∈ Y die Länge eines kürzesten Weges von s nach v im Graphen (V,E) durch l(v) gegeben.*

Beweis:

(1) Wie für Algorithmus 3.7 gelten alle Eigenschaften.
zusätzlich ist für jeden Knoten v ∈ Y per Induktion,
der Wert l(v) tatsächlich definiert.

(2) Die Laufzeit bleibt von Algorithmus 3.7 erhalten.

Mehr Details!

s.fekete@tu-bs.de