

Kapitel 3.3: Zusammenhangskomponenten

*Algorithmen und Datenstrukturen
WS 2025/26*

Prof. Dr. Sándor Fekete

SATZ 3.3

Wenn ein Weg zwischen zwei Knoten s und t in einem Graphen existiert, dann existiert auch ein Pfad.

$$W^* = s, e_1, v_1, \dots, v_i, e_{i+1}, \dots, v_j, e_k, \dots, t$$

$\uparrow$ $\uparrow$
erster letzter
Besuch Besuch

Dann gäbe es einen noch kürzeren Weg:

$$W^{**} = s, e_1, v_1, \dots, v_i, e_k, \dots, t$$

$\uparrow$
nur einmal
besucht!

Das widerspricht der Annahme, dass W^* kürzestmöglich ist.
Also hat W^* keinen doppelt besuchten Knoten, ist also ein Pfad! $\square$

DEFINITION 3.6 (Wald, Baum)

- (1) Ein Wald ist ein kreisfreier Graph.
- (2) Ein Baum ist eine Zusammenhangskomponente in einem Wald.
(Also: ein kreisfreier, zusammenhängender Graph)
- (3) Ein aufspannender Baum ist ein Baum, der alle Knoten verbindet. (Manchmal auch: Spannbaum. Englisch: "spanning tree")

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 $w \in V \setminus Y$ mit $e = \{v, w\} \in E$
 - 2.3.2. Setze $R := R \cup \{w\}$, $Y := Y \cup \{w\}$, $T := T \cup \{e\}$}}
3. STOP

Satz 3.8. *Der Algorithmus 3.7 ist*

(1) endlich

(2) korrekt.

- (2) Zu jedem Zeitpunkt ist (Y, T) ein s enthaltender Baum, denn
- (a) alle Knoten in G sind von s aus erreichbar (und umgekehrt)
 - (b) neu eingefügte Kanten verbinden die bisherige Knotenmenge Y nur mit bislang nicht erreichbaren Knoten, können also keinen Kreis schließen.

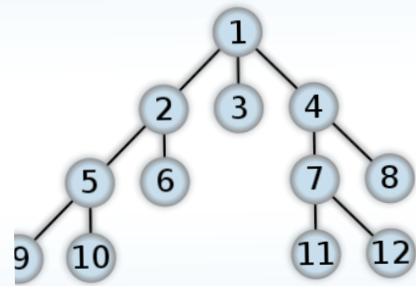
Jetzt müssen wir noch zeigen, dass alle erreichbaren Knoten auch korrekt identifiziert werden.

Angenommen, am Ende gibt es einen Knoten $w \in V \setminus Y$, der in G von s aus erreichbar ist.

Sei p ein Schrittlad in G , und sei $\{x, y\}$ eine Kante von p mit $x \in Y$, $y \notin Y$.

Da x zu Y gehört, wurde x auch zu R hinzugefügt.

Der Algorithmus stoppt aber nicht, bevor x aus R entfernt wurde. Das wird in (2.2.1) nur vorgenommen, wenn es in (2.2) keine Kante $\{x, y\}$ mit $y \notin Y$ gibt - im Widerspruch zur Annahme. □



Kapitel 3.4:
Warteschlange und Stapel
Algorithmen und Datenstrukturen
WS 2025/26

Prof. Dr. Sándor Fekete

Algorithmus 3.7

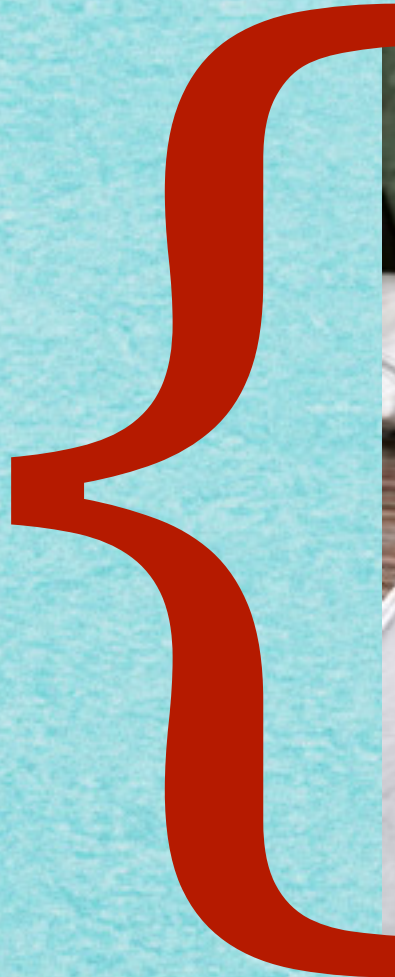
INPUT: Graph $G = (V, E)$, Knoten s

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 $w \in V \setminus Y$ mit $e = \{v, w\} \in E$
 - 2.3.2. Setze $R := R \cup \{w\}$, $Y := Y \cup \{w\}$, $T := T \cup \{e\}$}}
3. STOP

Wie verwalten wir R?

R



Wähle v!

Waschkorb



***A WEBCOMIC OF ROMANCE,
SARCASM, MATH, AND LANGUAGE.***



HOME ORGANIZATION TIP:
JUST GIVE UP.

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 $w \in V \setminus Y$ mit $e = \{v, w\} \in E$
 - 2.3.2. Setze $R := R \cup \{w\}$, $Y := Y \cup \{w\}$, $T := T \cup \{e\}$}}
3. STOP

A & D

I get the job done.
What the hell do you
want?

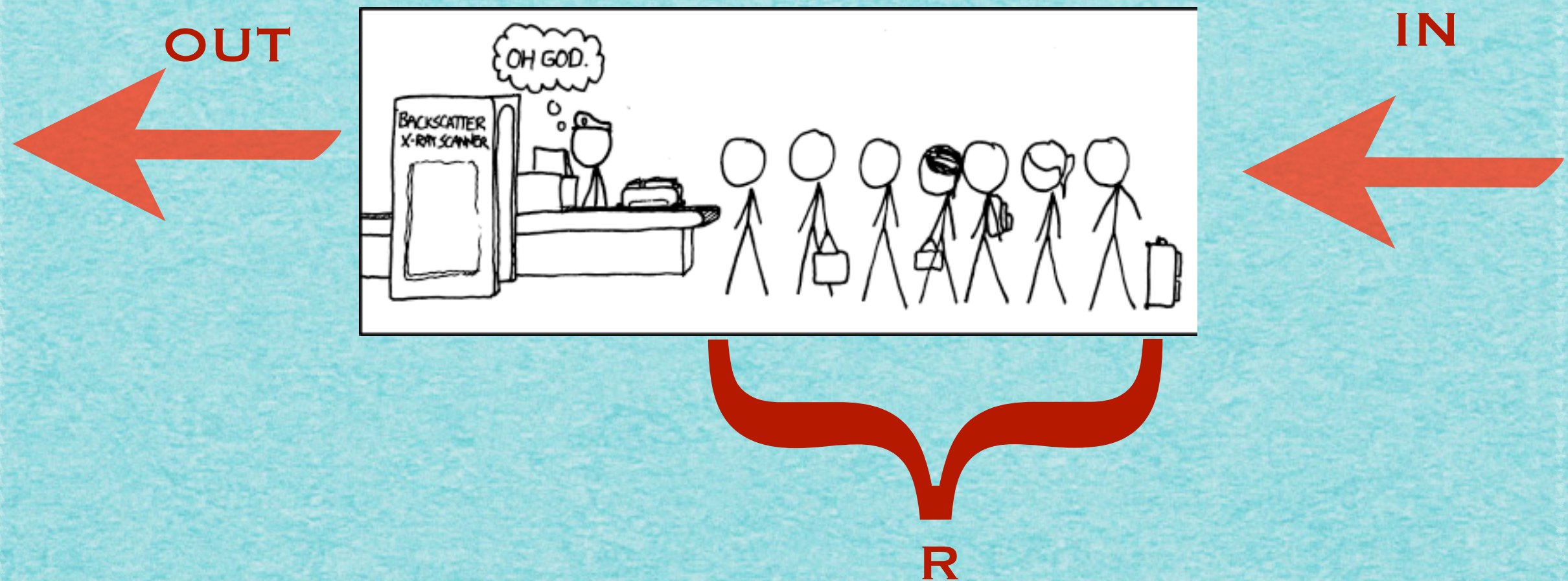
CAN YOU MAKE IT
WITHOUT KILLING
YOURSELF?



Algorithmus

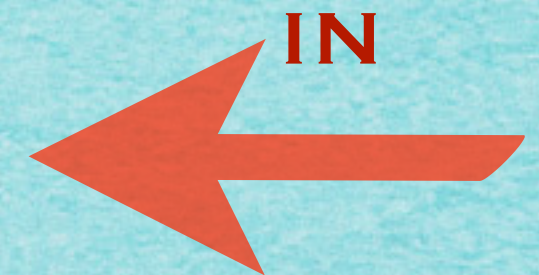
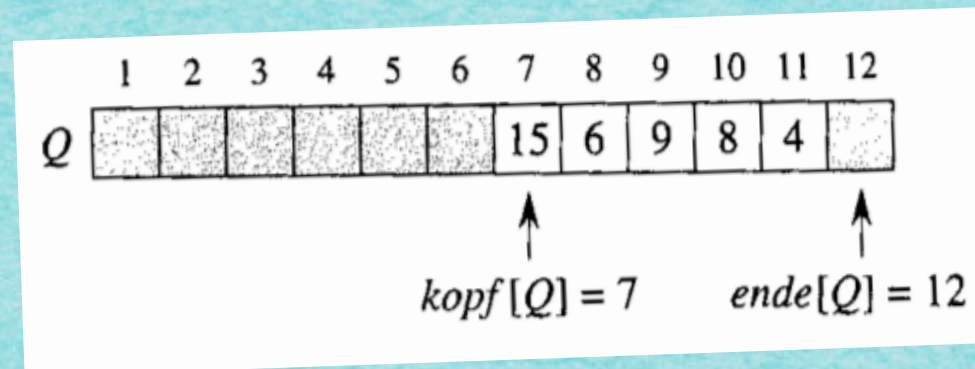
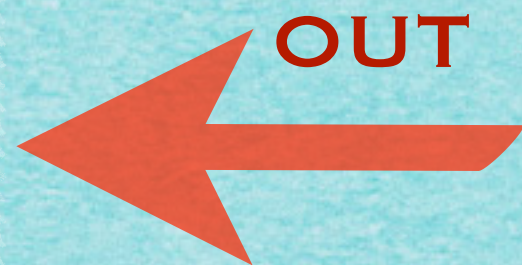
DATENSTRUKTUR

DATENSTRUKTUR I

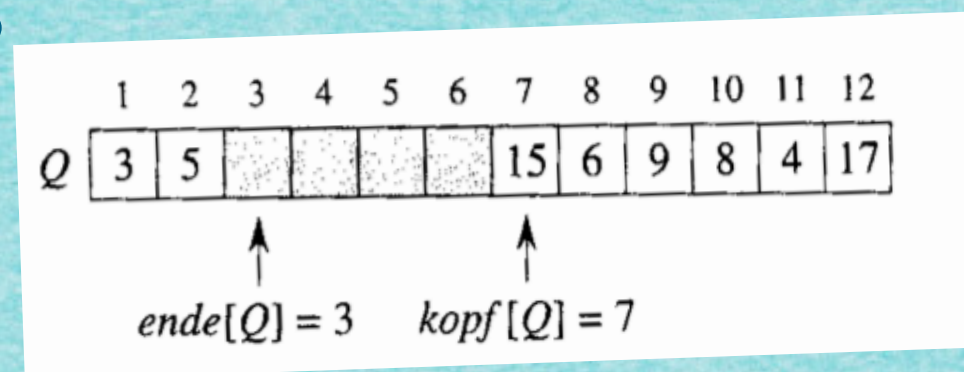


WARTESCHLANGE: FIRST IN - FIRST OUT

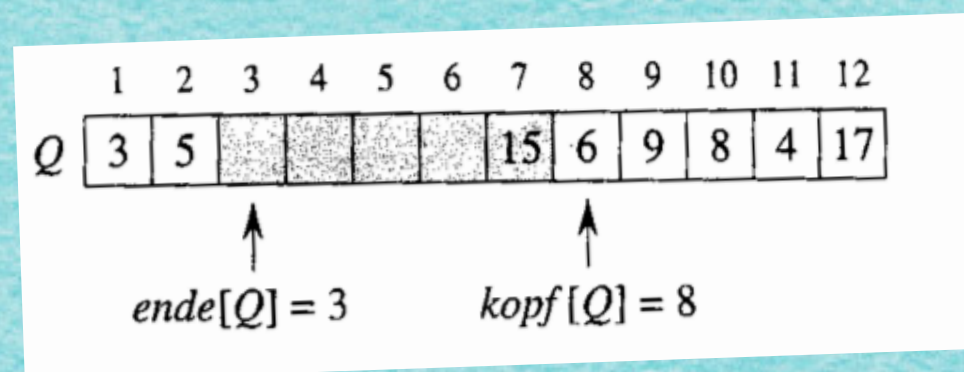
WARTESCHLANGE AUF ARRAY UMGESETZT



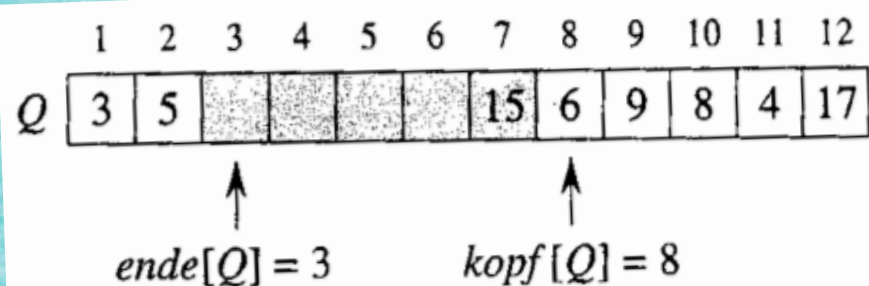
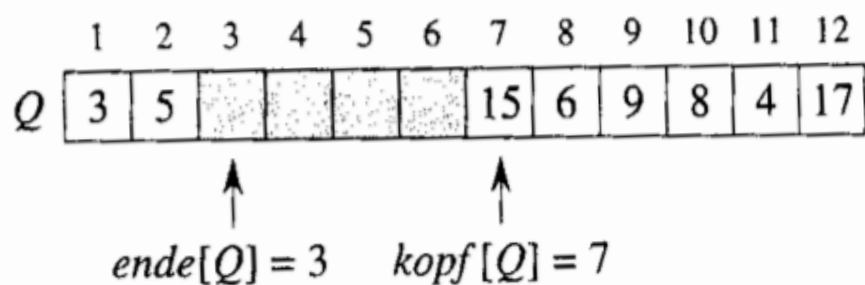
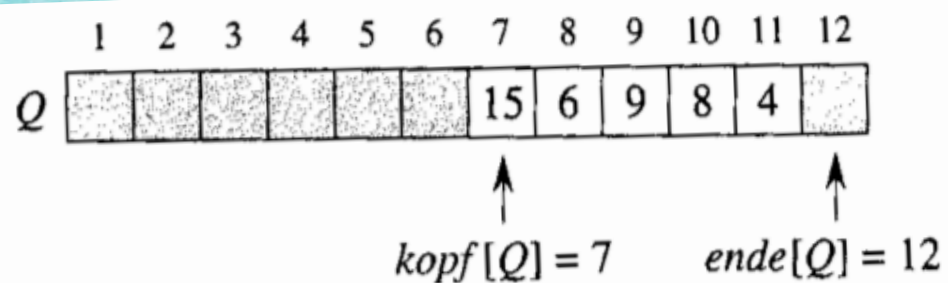
ENQUEUE: 17, 3, 5



DEQUEUE:



WARTESCHLANGE AUF ARRAY UMGESETZT



ENQUEUE(Q, x)

```

1   $Q[ende[Q]] \leftarrow x$ 
2  if  $ende[Q] = länge[Q]$ 
3      then  $ende[Q] \leftarrow 1$ 
4      else  $ende[Q] \leftarrow ende[Q] + 1$ 

```

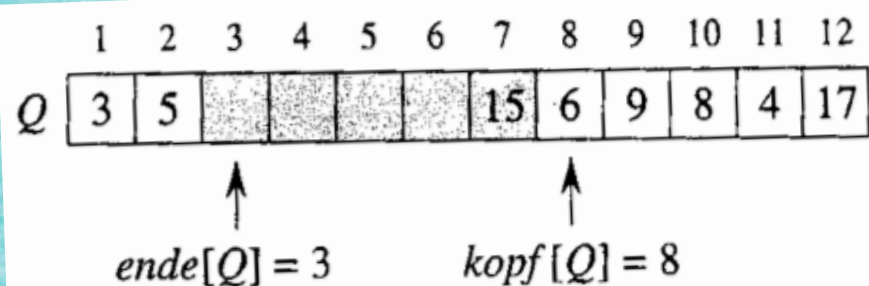
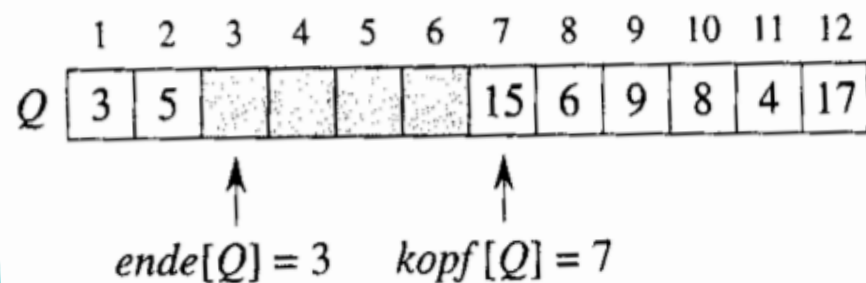
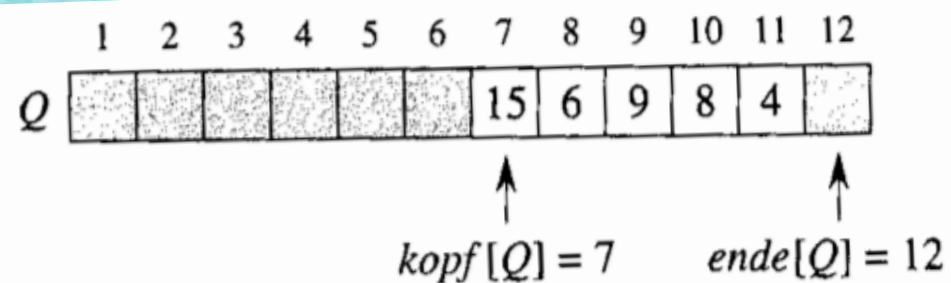
DEQUEUE(Q)

```

1   $x \leftarrow Q[kopf[Q]]$ 
2  if  $kopf[Q] = länge[Q]$ 
3      then  $kopf[Q] \leftarrow 1$ 
4      else  $kopf[Q] \leftarrow kopf[Q] + 1$ 
5  return  $x$ 

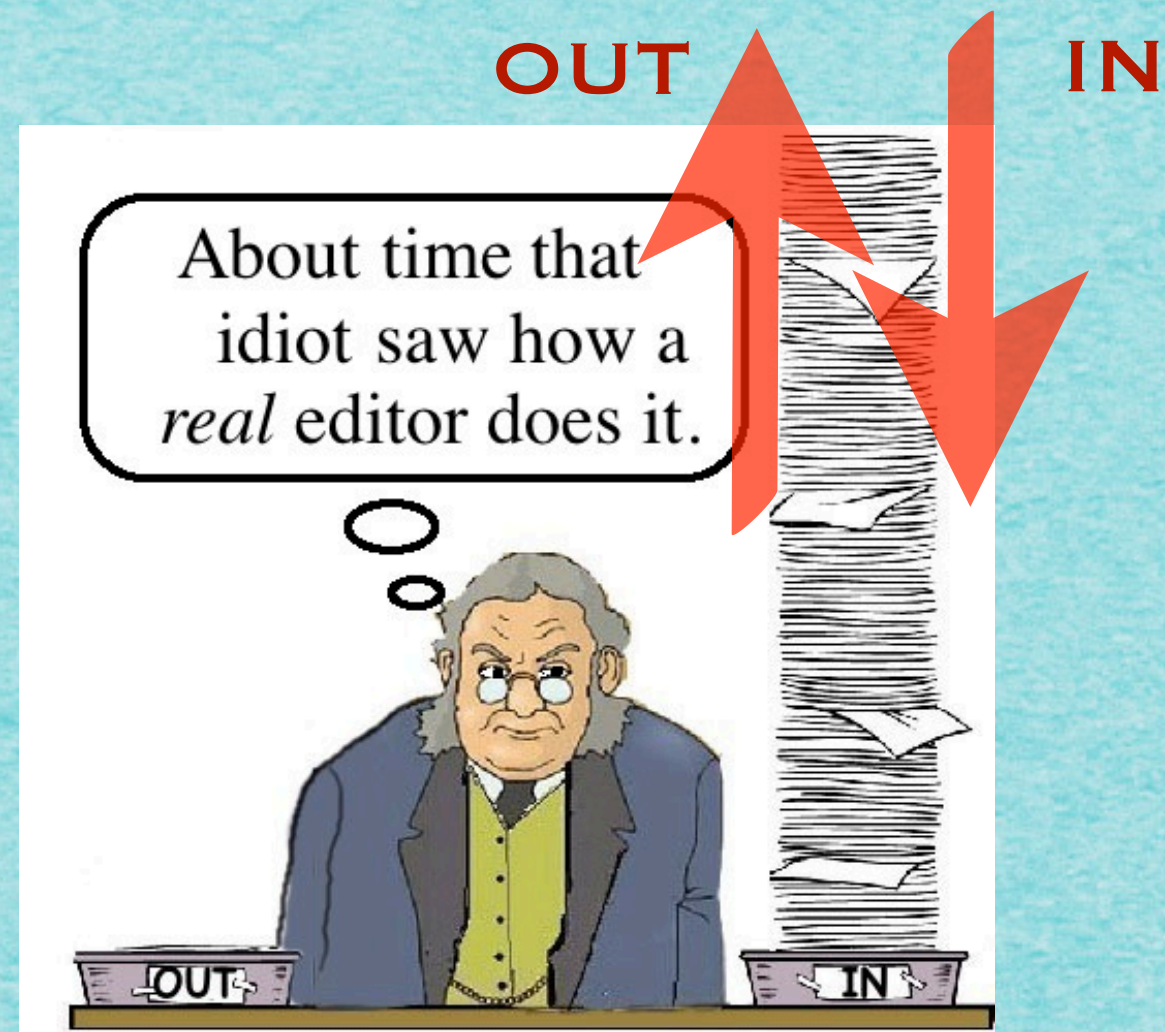
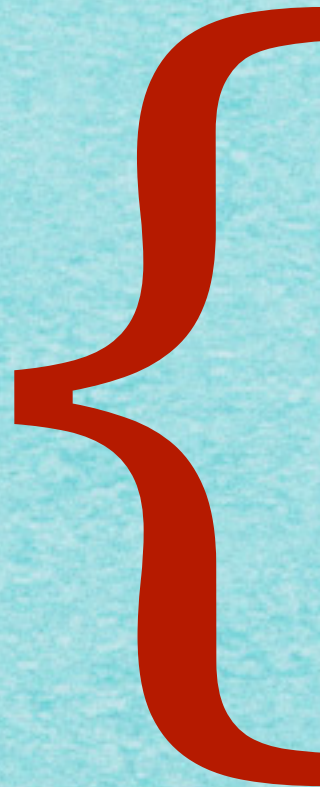
```


WARTESCHLANGE AUF ARRAY UMGESETZT



DATENSTRUKTUR II

R



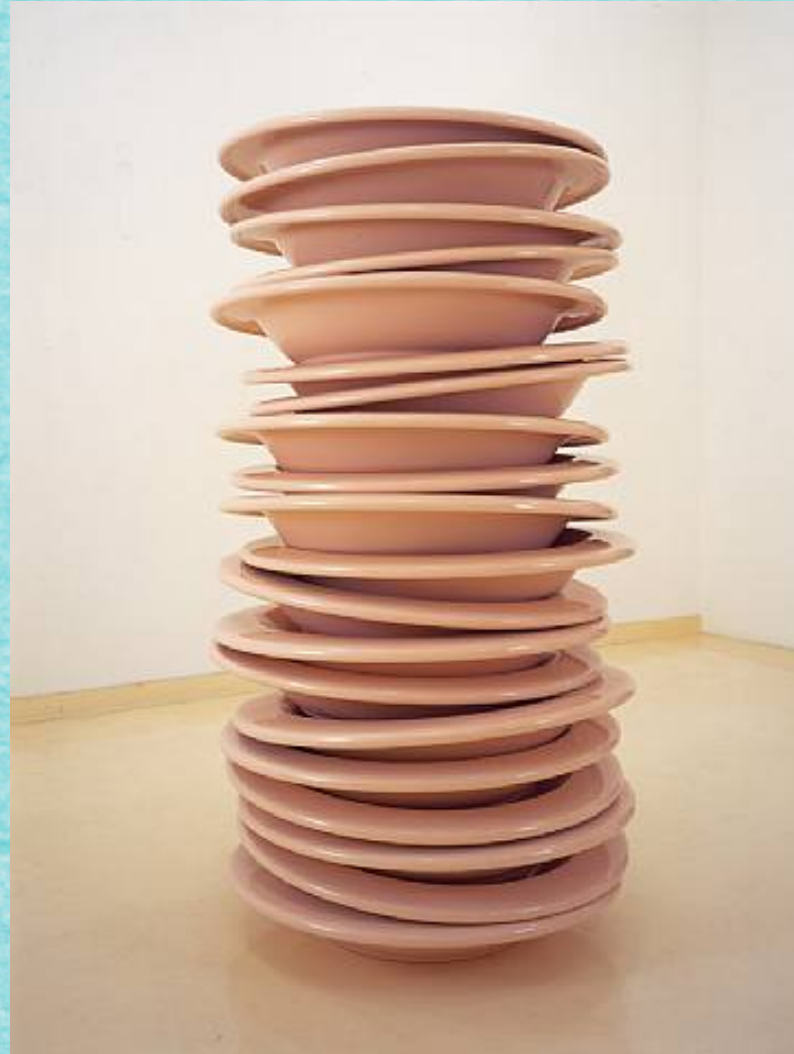
STAPEL: LAST IN - FIRST OUT

DATENSTRUKTUR II



STAPEL: LAST IN - FIRST OUT

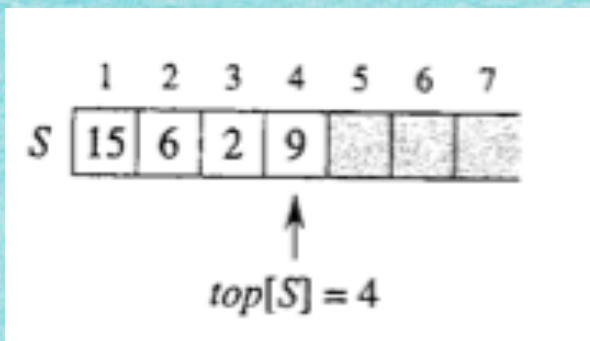
DATENSTRUKTUR II



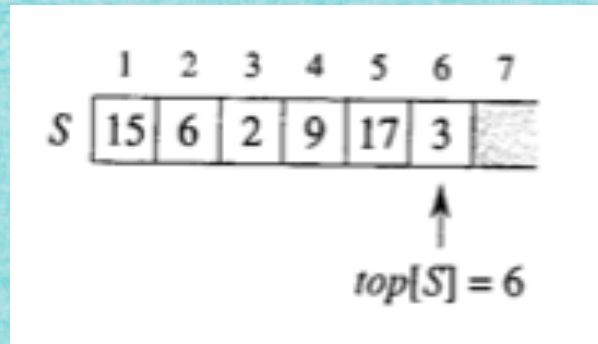
STAPEL: LAST IN - FIRST OUT

STACK

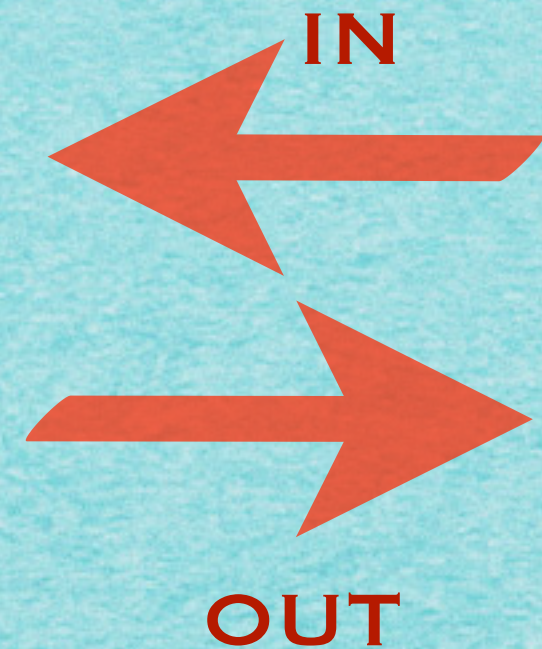
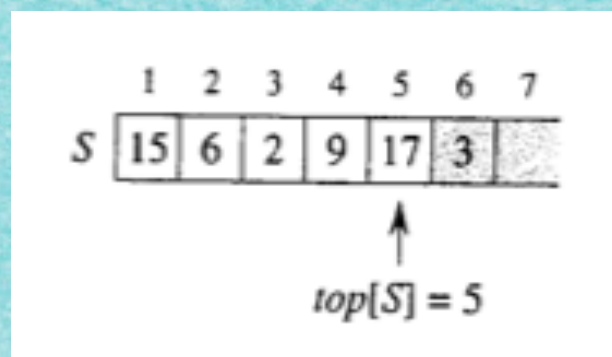
AUF ARRAY UMGESETZT



PUSH: 17, 3



POP



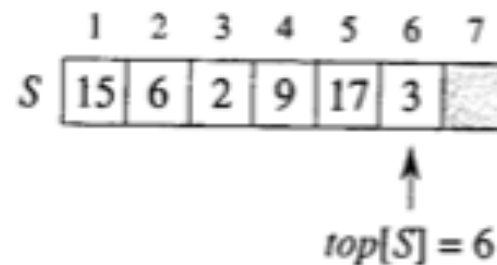
STACK

AUF ARRAY UMGESETZT



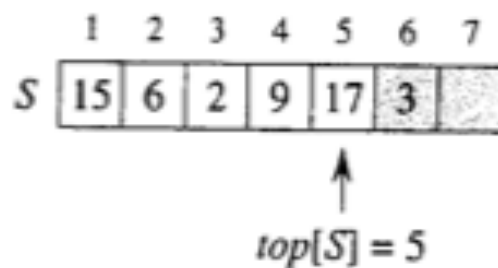
```

STACK-EMPTY(S)
1  if  $top[S] = 0$ 
2      then return WAHR
3      else return FALSCH
    
```



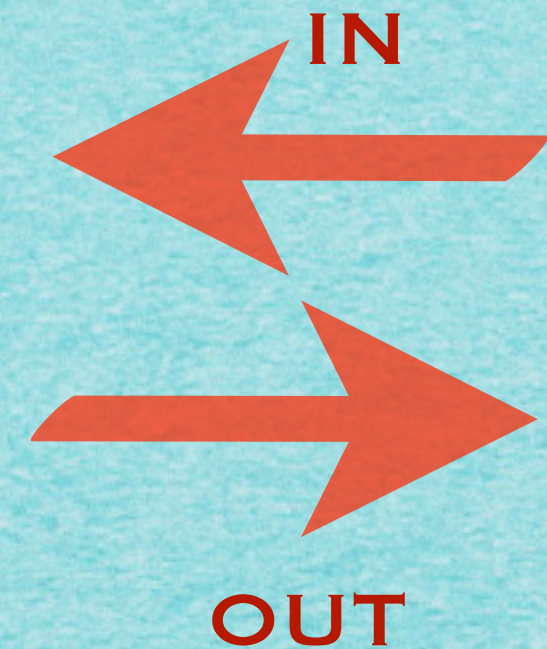
```

PUSH(S, x)
1   $top[S] \leftarrow top[S] + 1$ 
2   $S[top[S]] \leftarrow x$ 
    
```



```

POP(S)
1  if STACK-EMPTY(S)
2      then error "Unterlauf"
3      else  $top[S] \leftarrow top[S] - 1$ 
4              return  $S[top[S] + 1]$ 
    
```



KARRIERECHANCEN!

KARRIERE SPIEGEL

● Home ● Berufsstart ● Berufsleben ● Ausland ● Stellensuche ● Jobs

Thema Auto und Beruf - KarriereSPIEGEL »

Nachrichten > KarriereSPIEGEL > Berufsstart > Auto und Beruf - KarriereSPIEGEL > Auto-Industrie

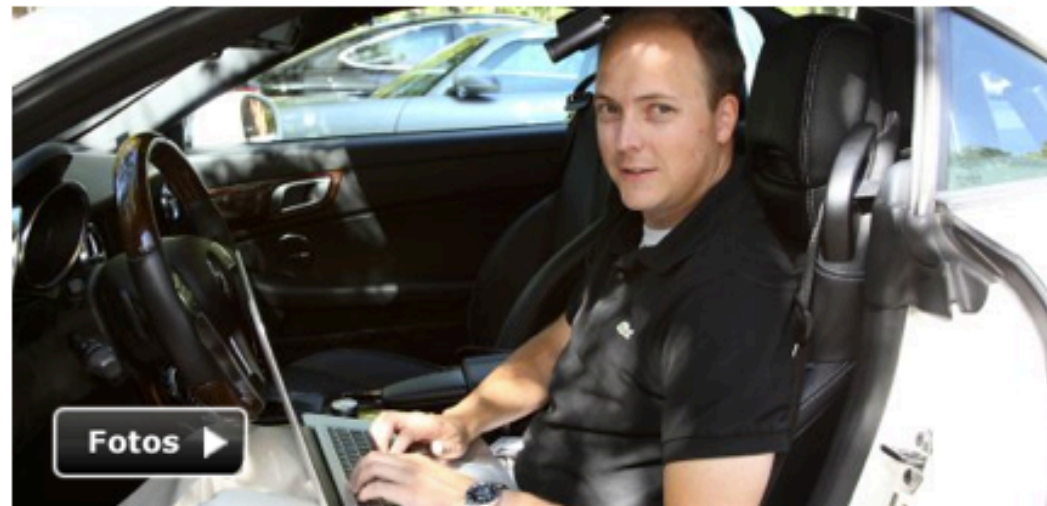
05.10.2012

Drucken | Senden | Feedback | Merken

Informatiker im Autobau

Mehr Software als im Kampfjet

Von Peter Ilg



Fotos ▶

Daimler

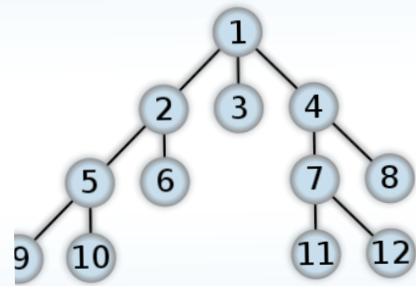
KARRIERECHANCEN!

Am liebsten reine Informatiker

Insgesamt 30 Mitarbeiter werkeln bei Daimler an der Systemarchitektur des Infotainment im Auto, die meisten sind Informatiker oder Ingenieure der Elektrotechnik. Vor sechs Jahren waren es noch halb so viele. "Bei Berufseinsteigern sind uns reine Informatiker am liebsten", sagt Matthias Stümpfle, der Leiter der Abteilung: "Sie sind in der Informatik gründlich ausgebildet, das Branchen-Know-how bringen wir ihnen bei."

KARRIERECHANCEN!



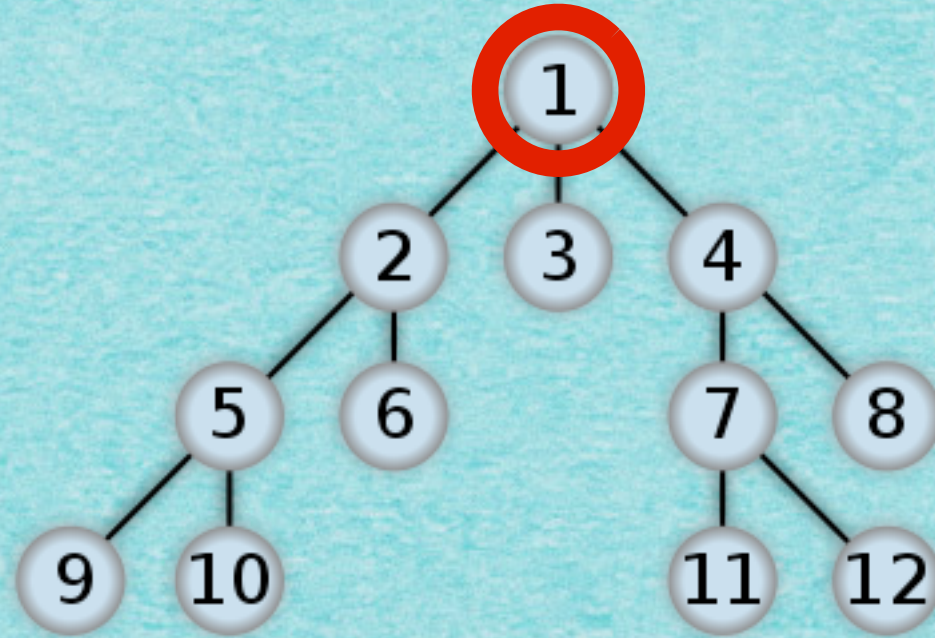


Kapitel 3.5: Tiefensuche und Breitensuche

*Algorithmen und Datenstrukturen
WS 2025/26*

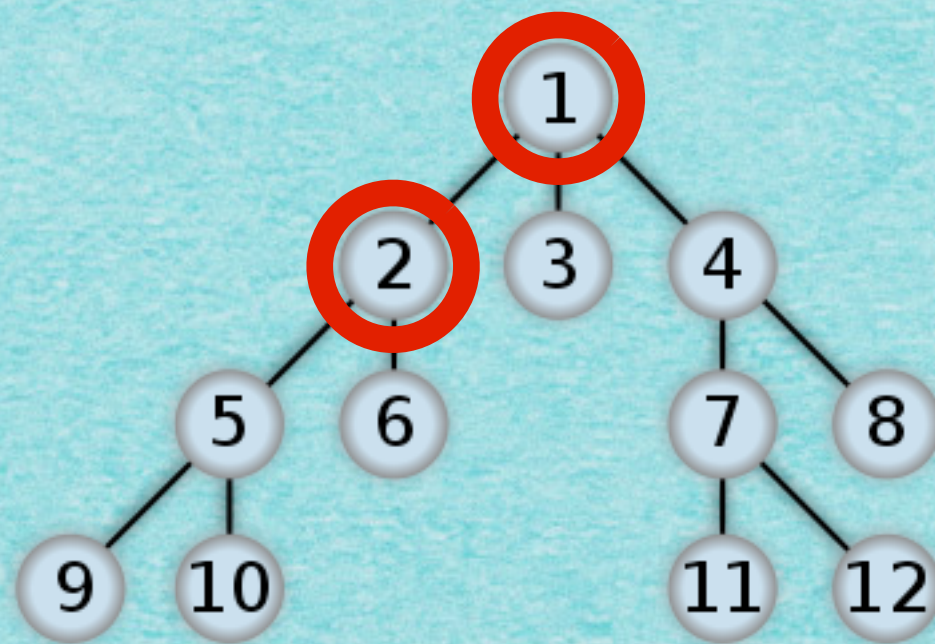
Prof. Dr. Sándor Fekete

Graphenscan mit WARTESCHLANGE



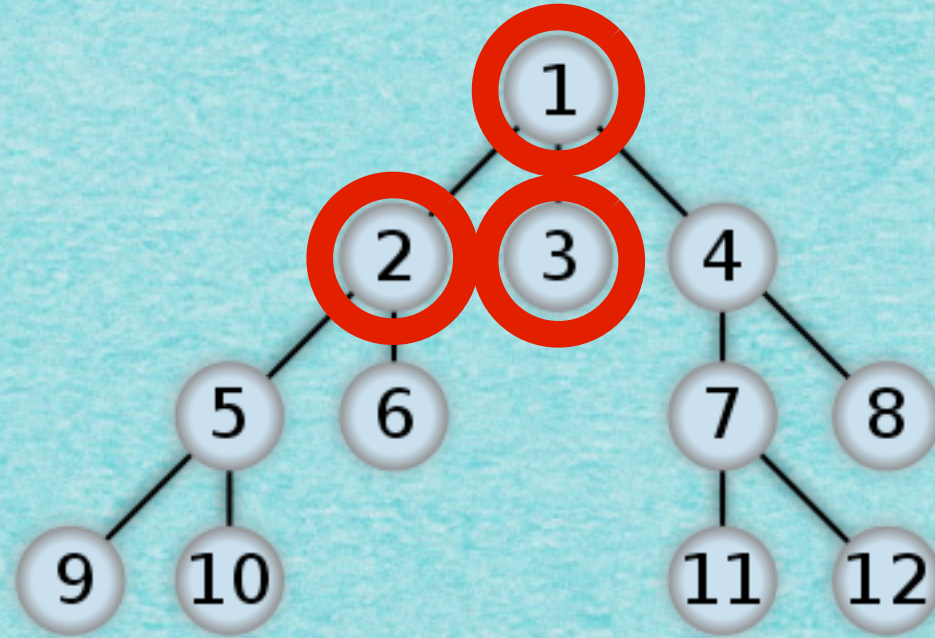
R: 1

Graphenscan mit WARTESCHLANGE



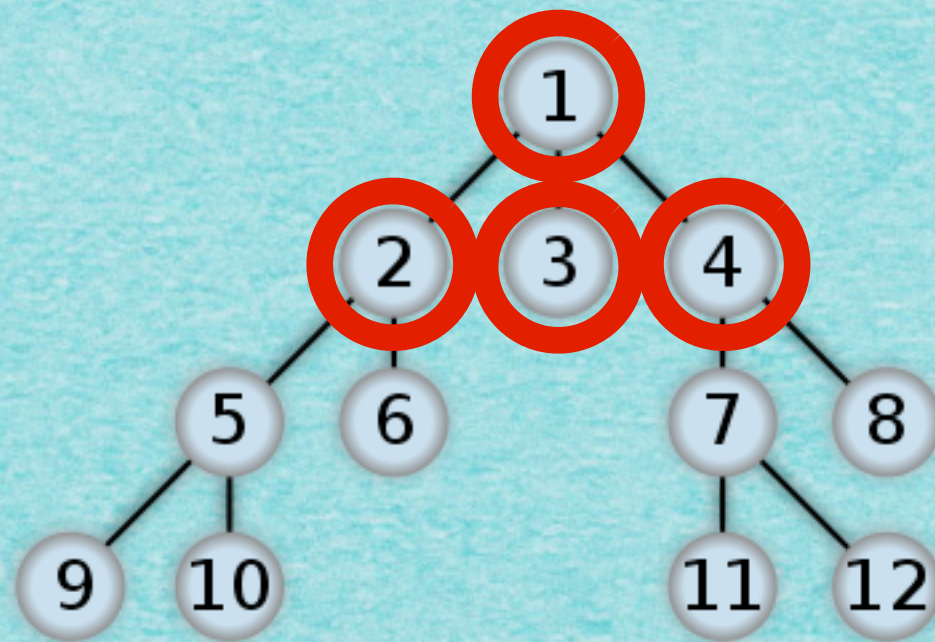
R: 1,2

Graphenscan mit WARTESCHLANGE



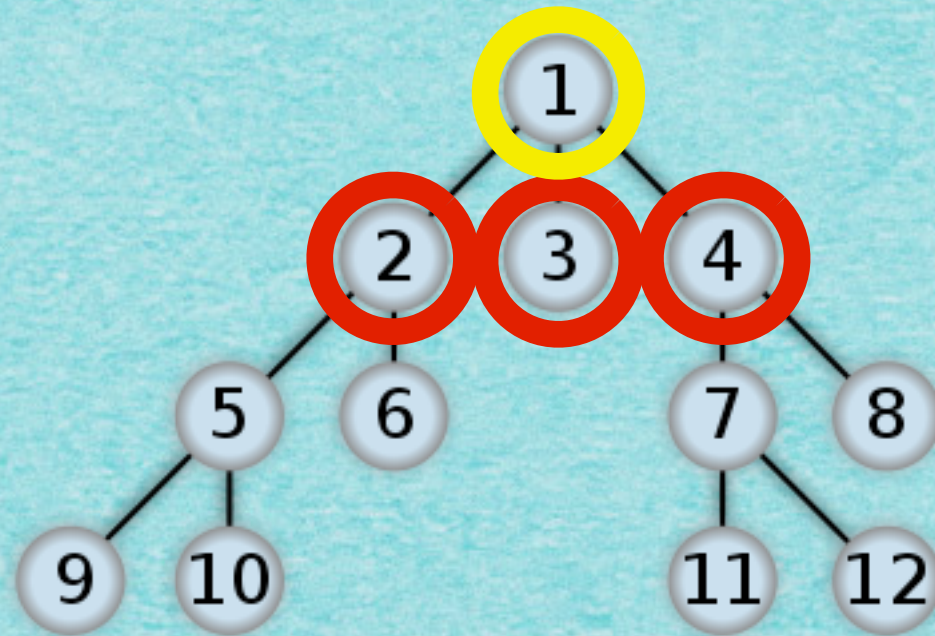
R: 1,2,3

Graphenscan mit WARTESCHLANGE



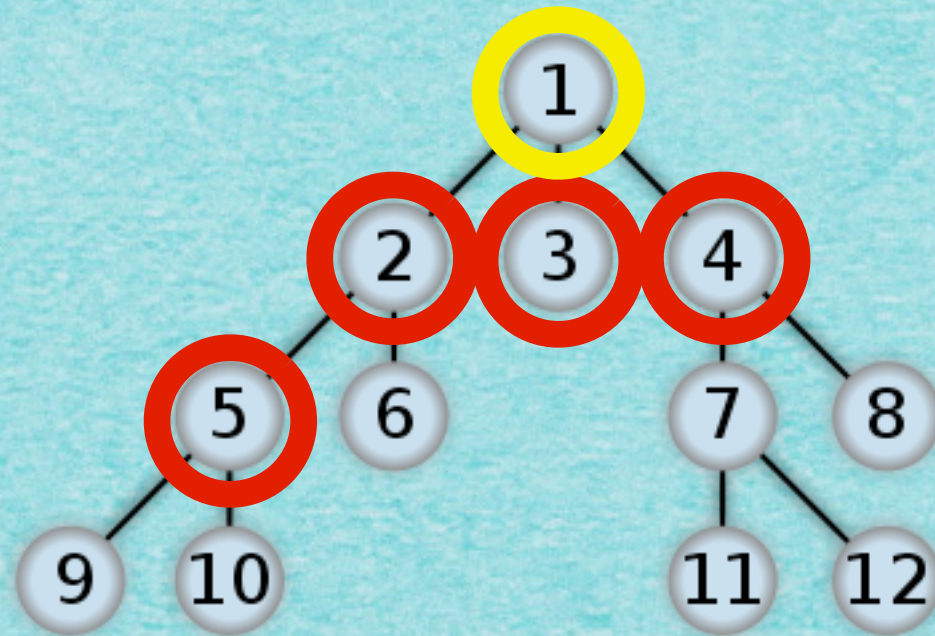
R: 1,2,3,4

Graphenscan mit WARTESCHLANGE



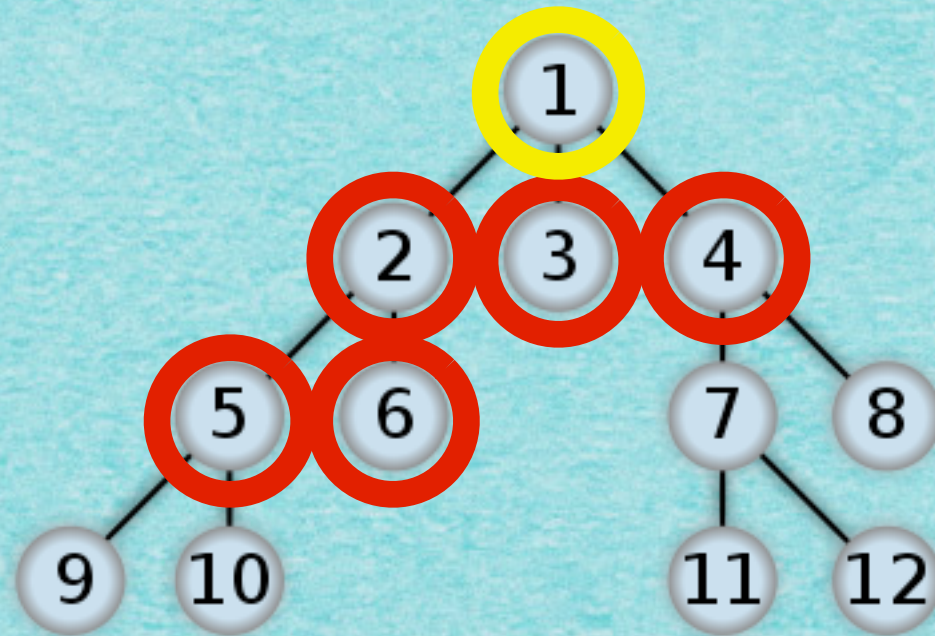
R: 2,3,4

Graphenscan mit WARTESCHLANGE



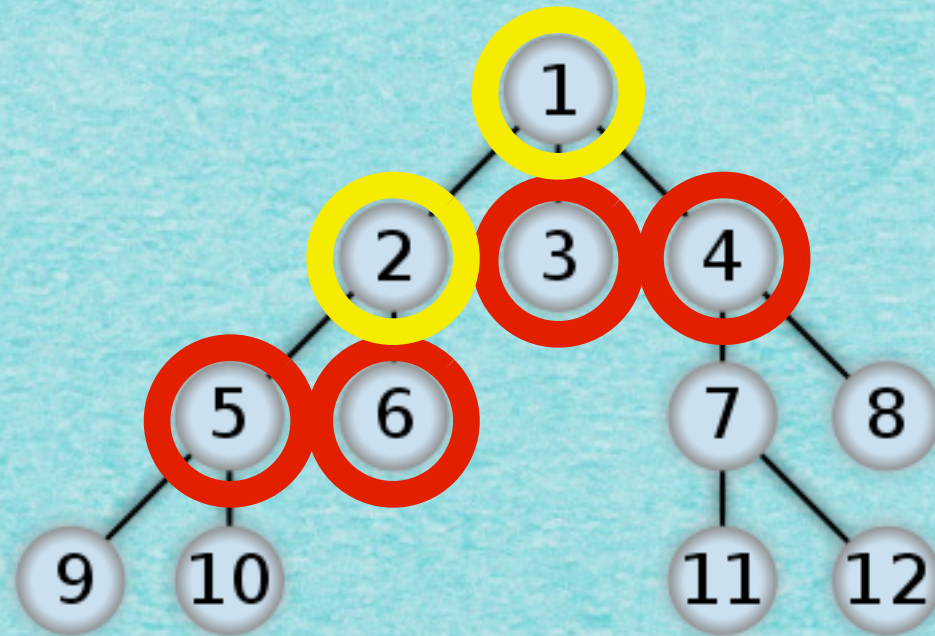
R: 2,3,4,5

Graphenscan mit WARTESCHLANGE



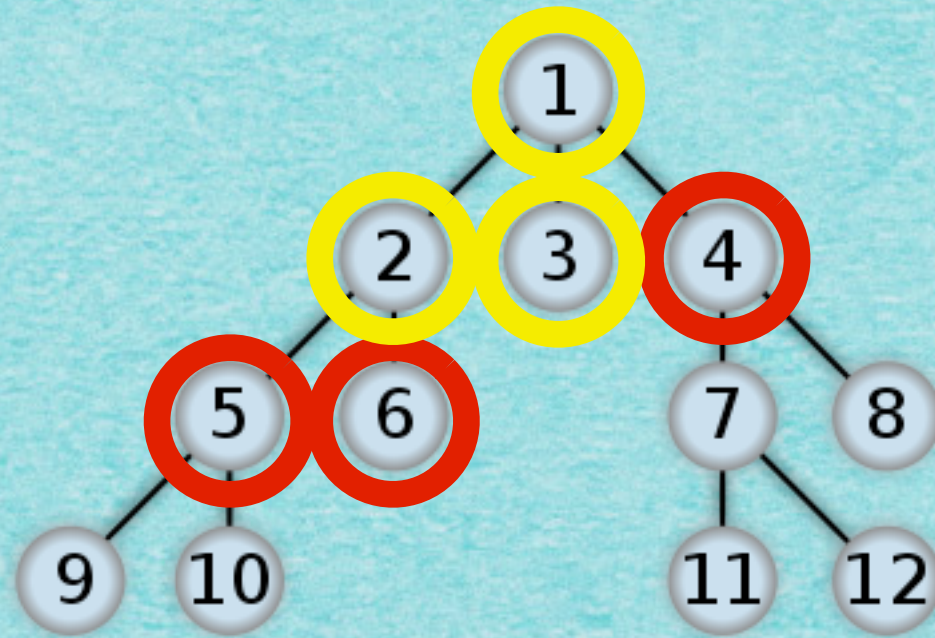
R: 2,3,4,5,6

Graphenscan mit WARTESCHLANGE



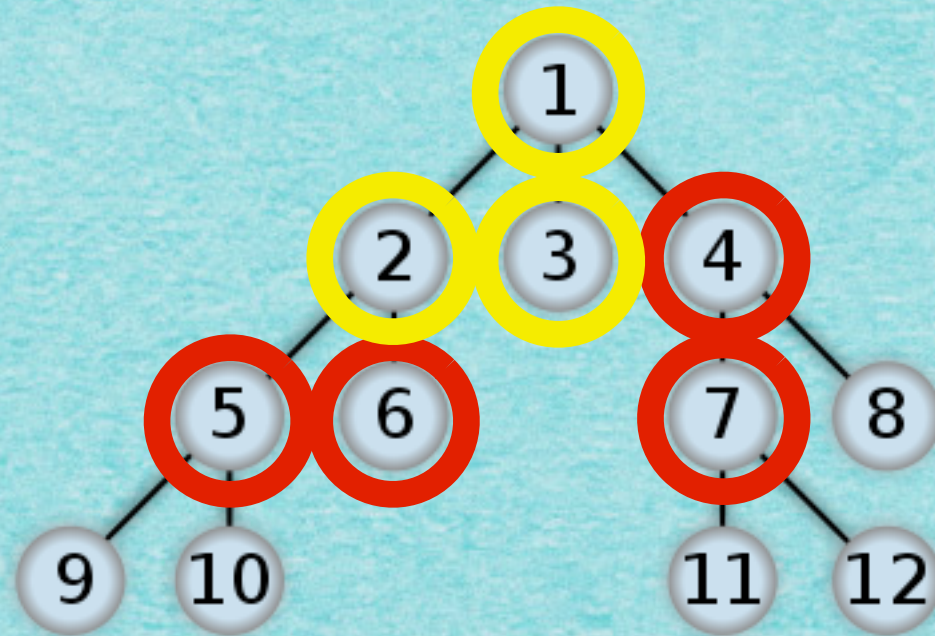
R: 3,4,5,6

Graphenscan mit WARTESCHLANGE



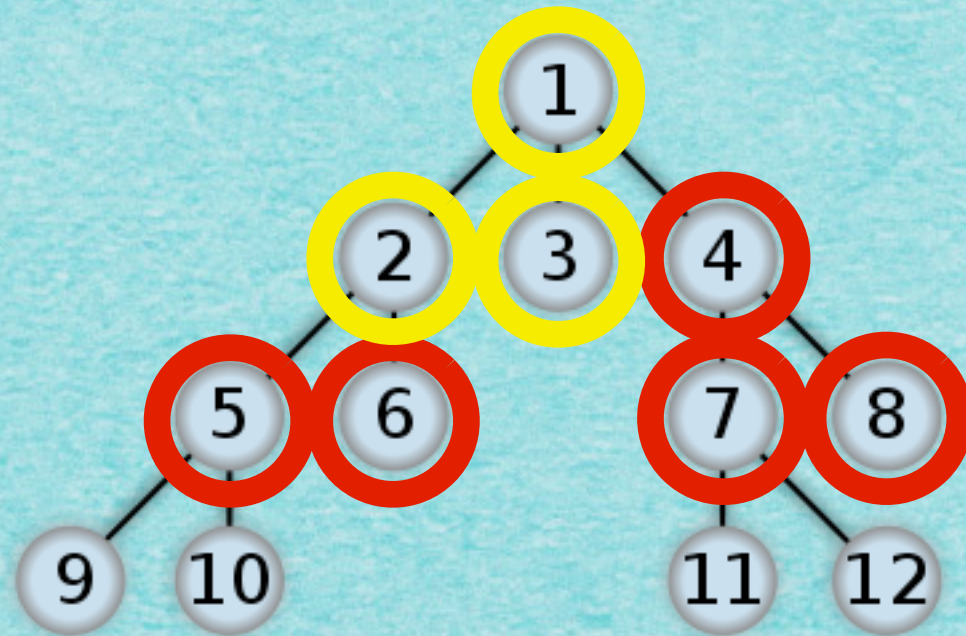
R: 4,5,6

Graphenscan mit WARTESCHLANGE



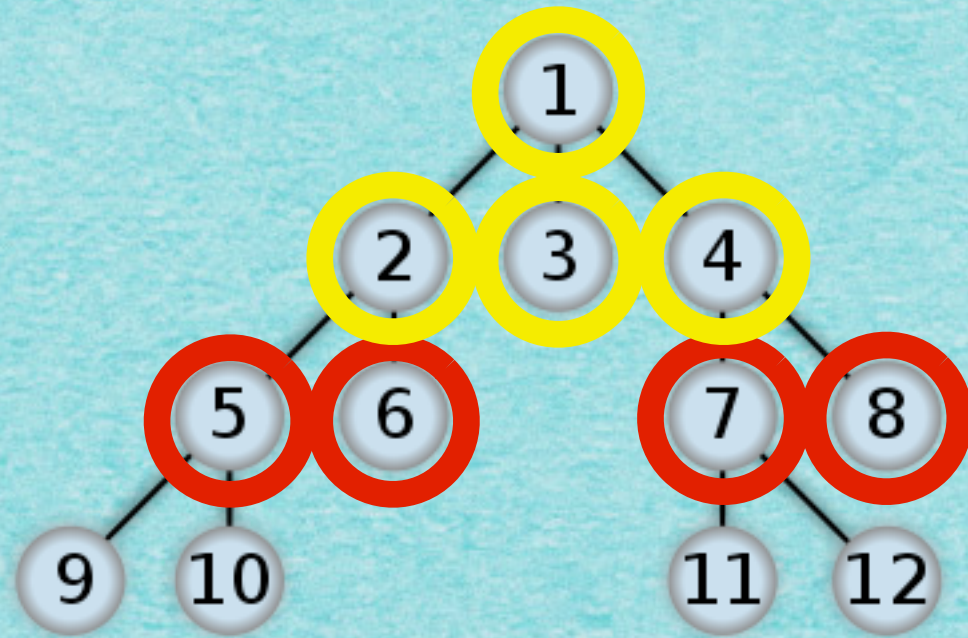
R: 4,5,6,7

Graphenscan mit WARTESCHLANGE



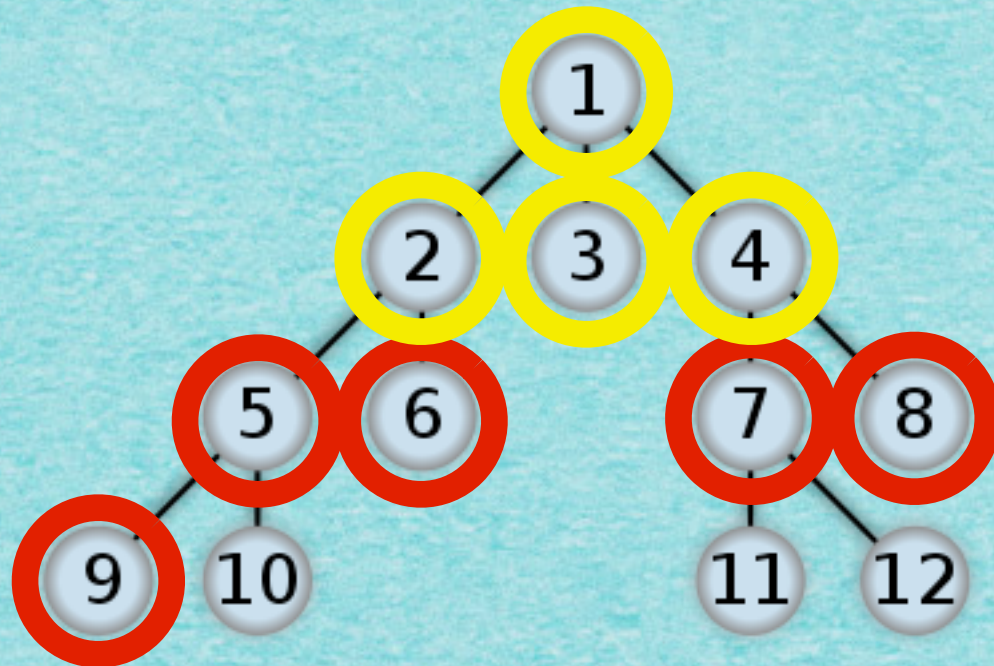
R: 4,5,6,7,8

Graphenscan mit WARTESCHLANGE



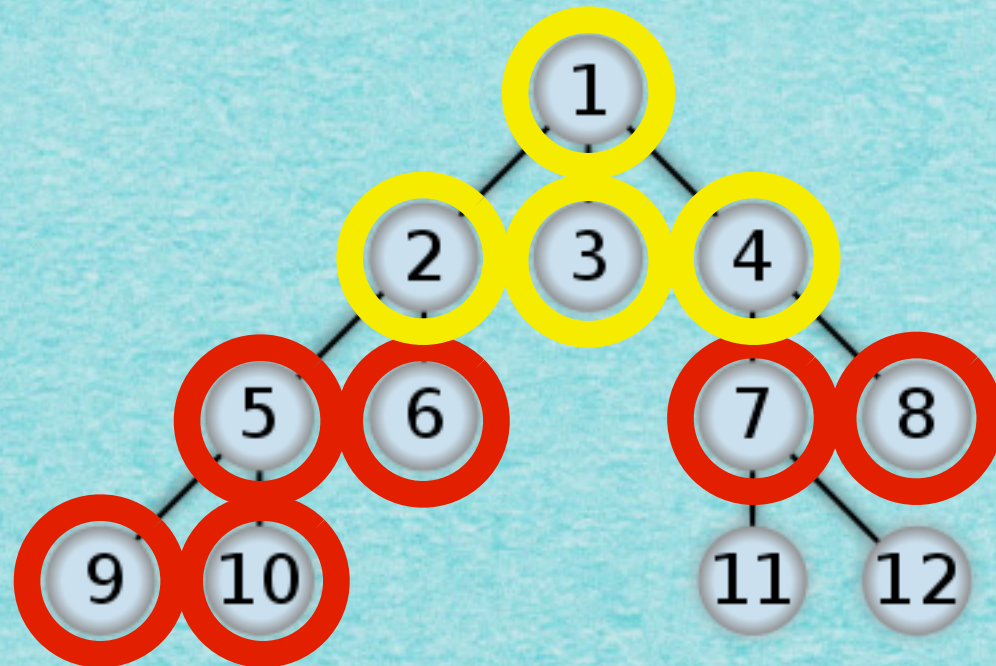
R: 5,6,7,8

Graphenscan mit WARTESCHLANGE



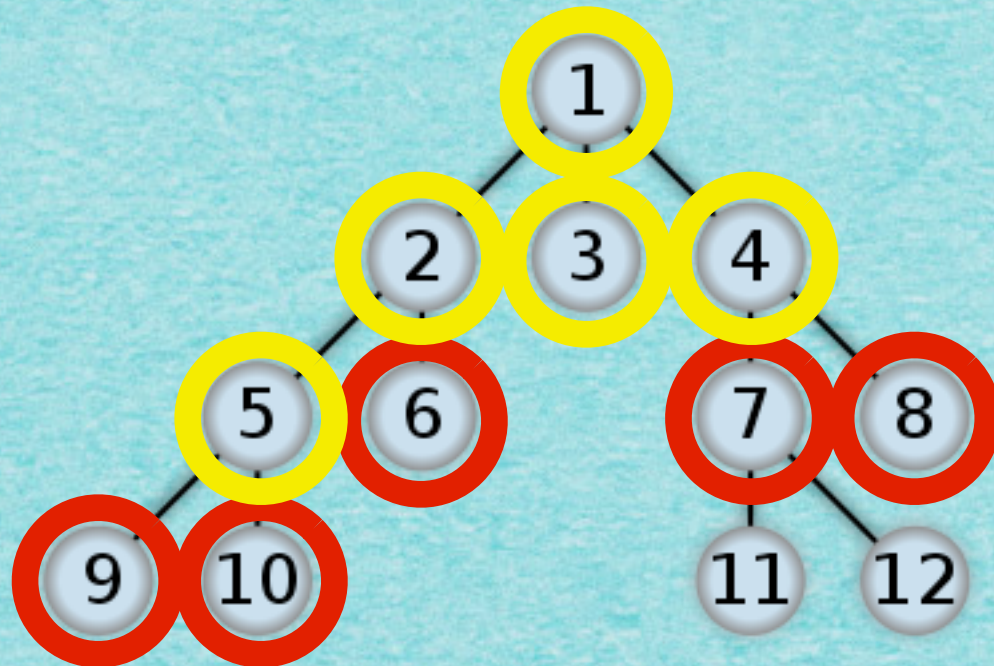
R: 5,6,7,8,9

Graphenscan mit WARTESCHLANGE



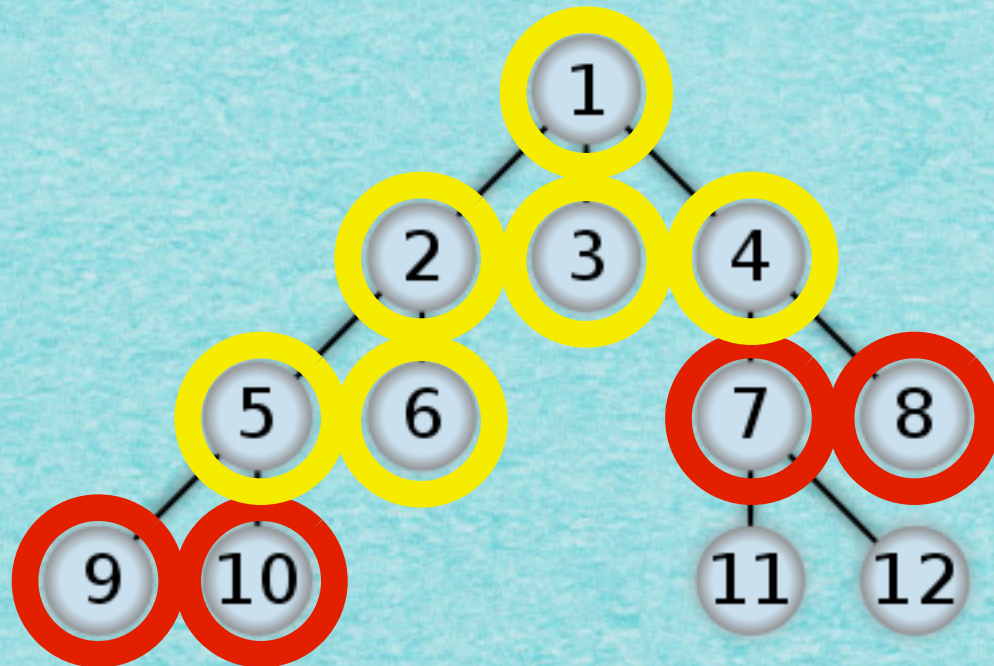
R: 5,6,7,8,9,10

Graphenscan mit WARTESCHLANGE



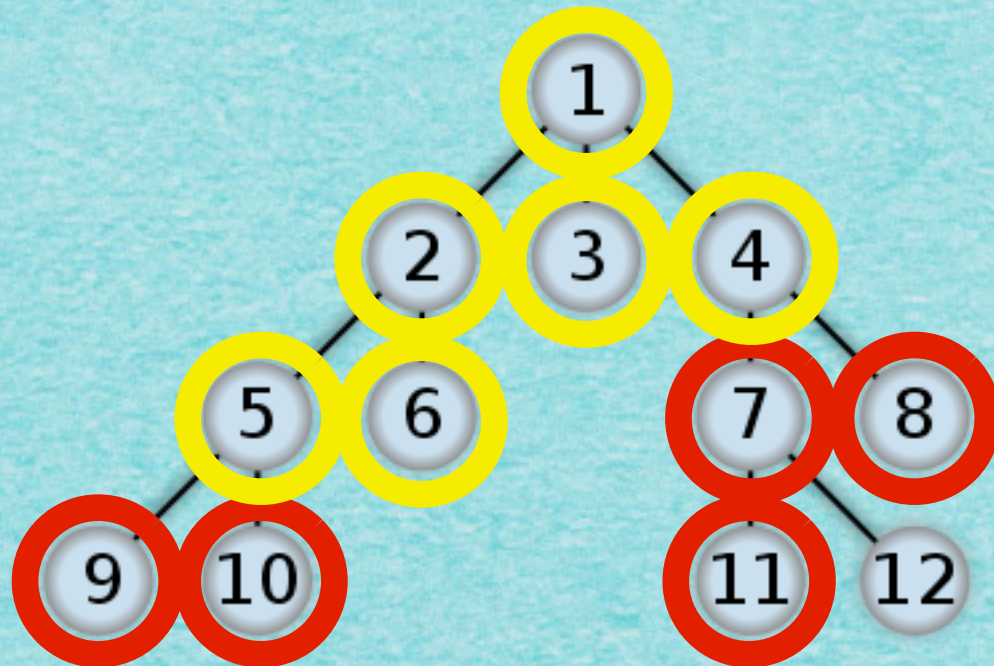
R: 6,7,8,9,10

Graphenscan mit WARTESCHLANGE



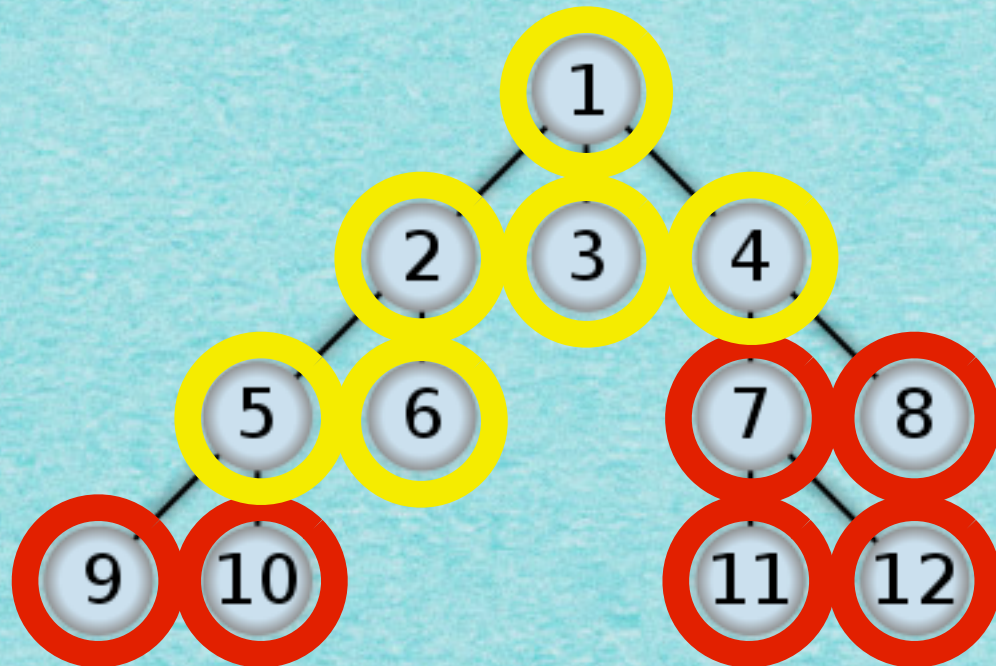
R: 7,8,9,10

Graphenscan mit WARTESCHLANGE



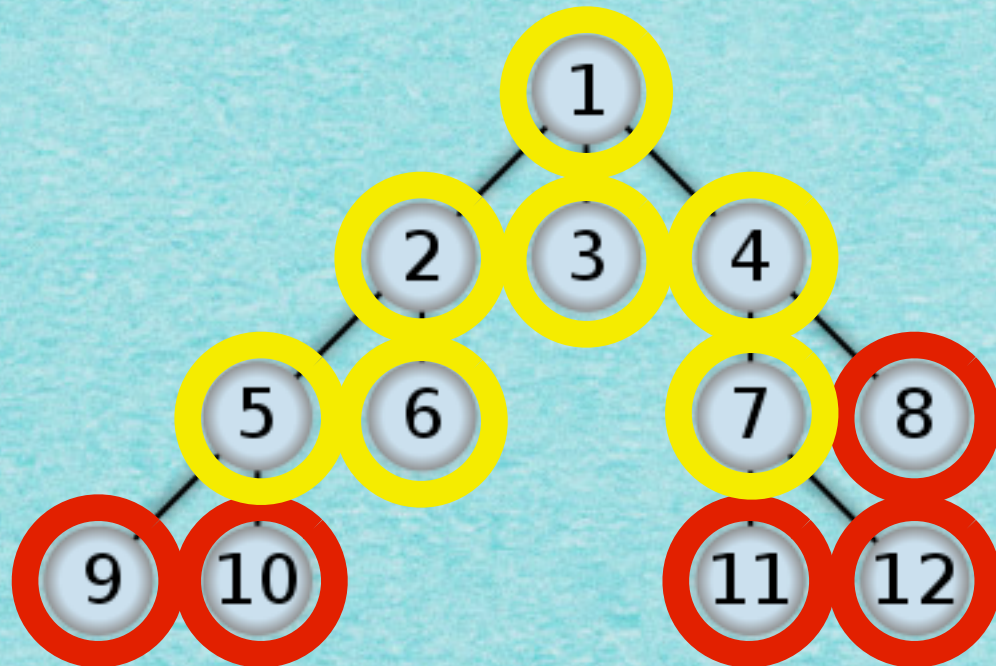
R: 7,8,9,10,11

Graphenscan mit WARTESCHLANGE



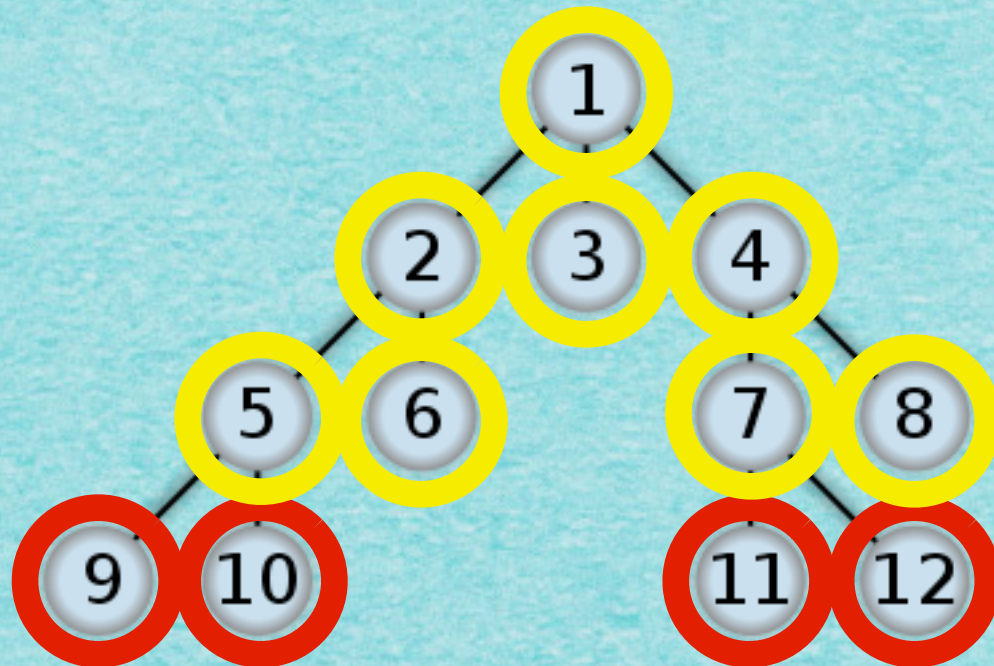
R: 7,8,9,10,11,12

Graphenscan mit WARTESCHLANGE



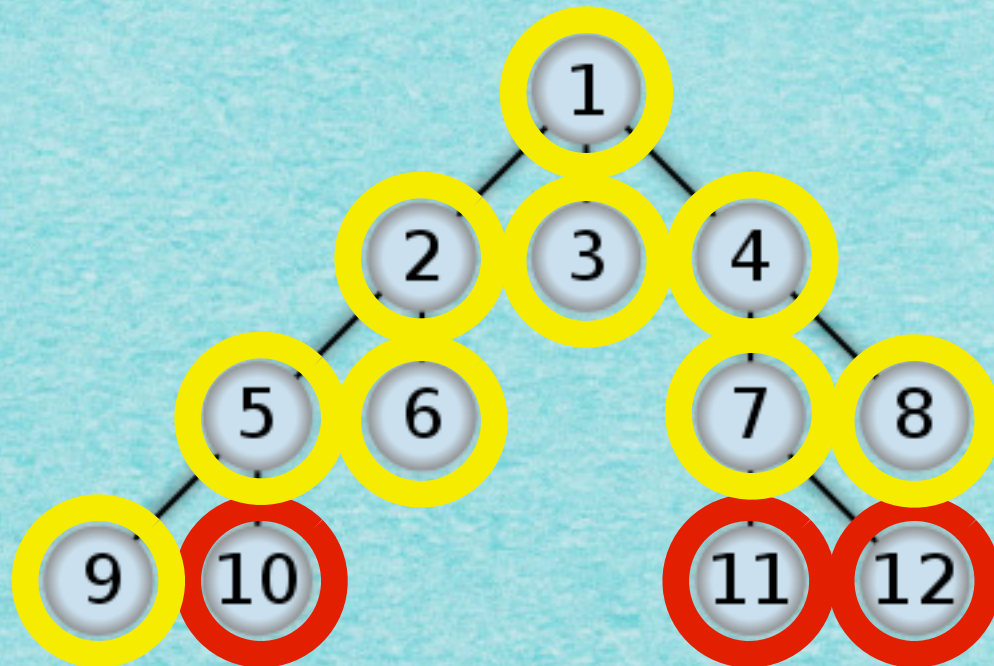
R: 8,9,10,11,12

Graphenscan mit WARTESCHLANGE



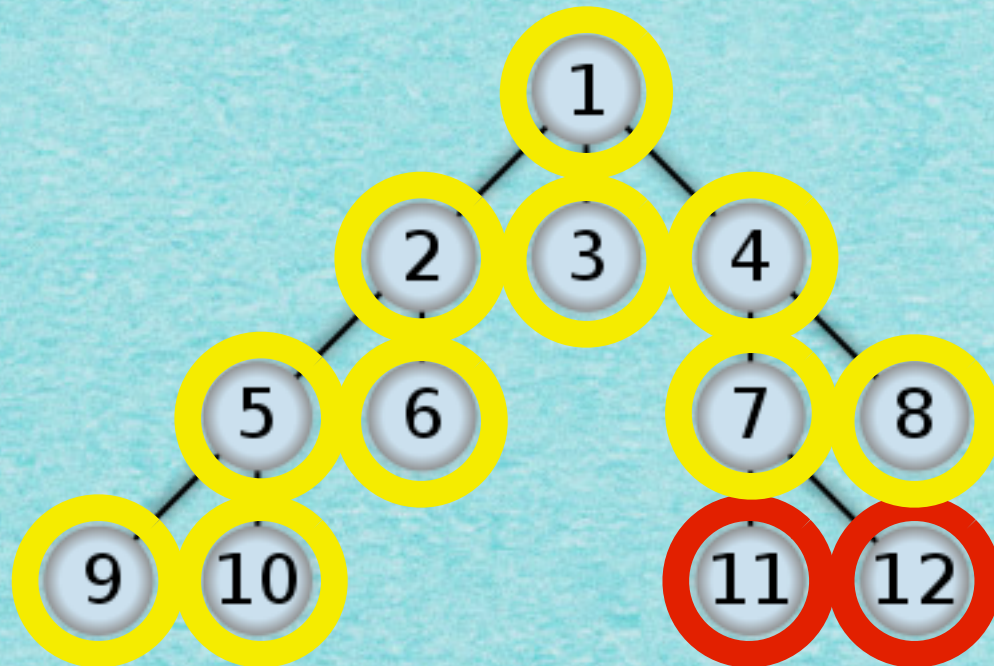
R: 9,10,11,12

Graphenscan mit WARTESCHLANGE



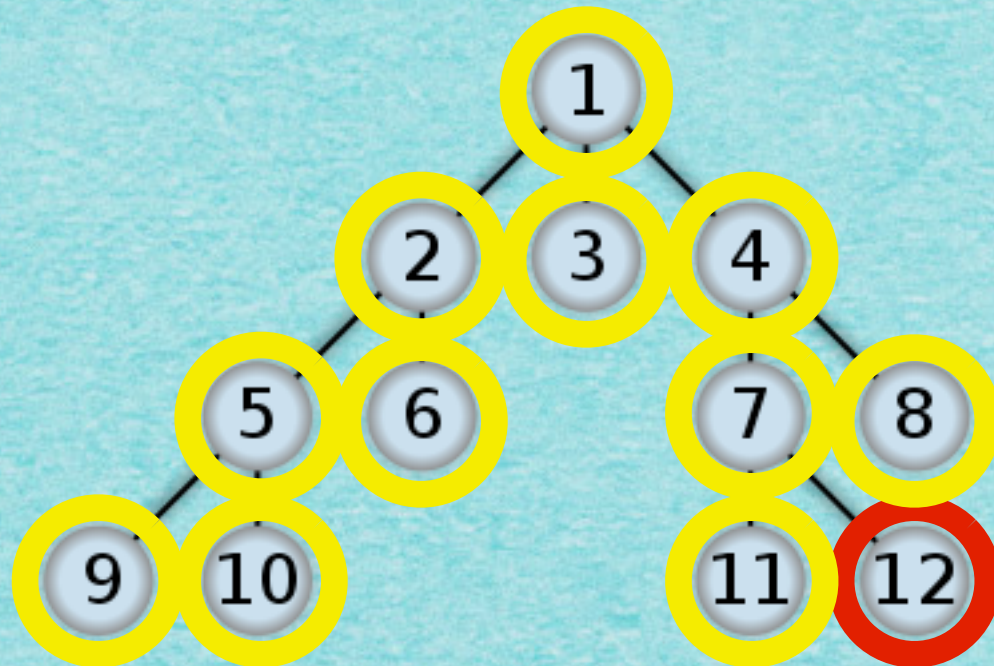
R:10,11,12

Graphenscan mit WARTESCHLANGE



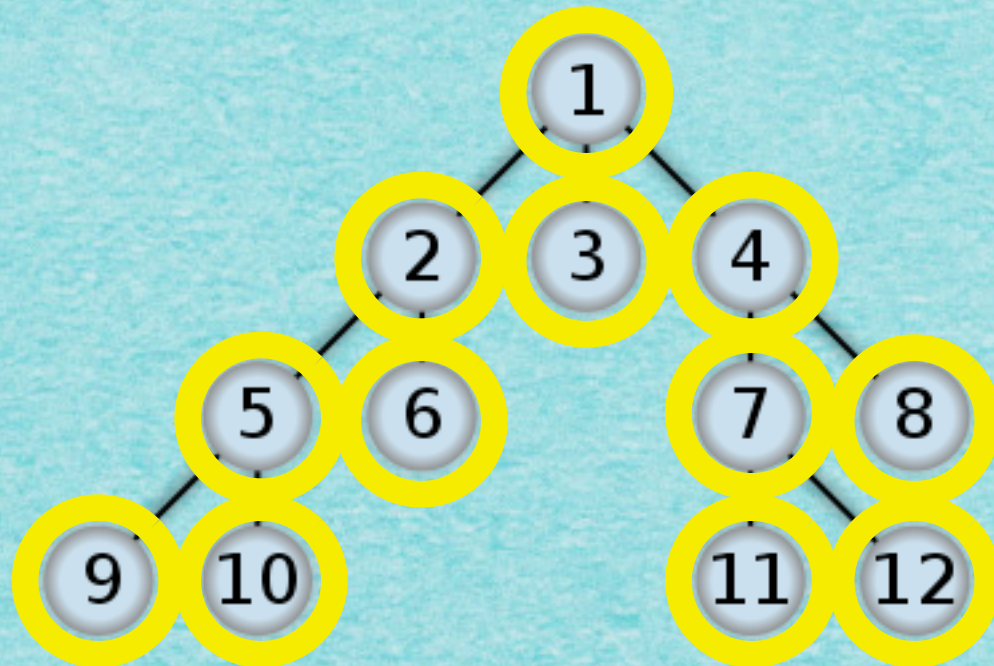
R:1 1, 12

Graphenscan mit WARTESCHLANGE



R:1 2

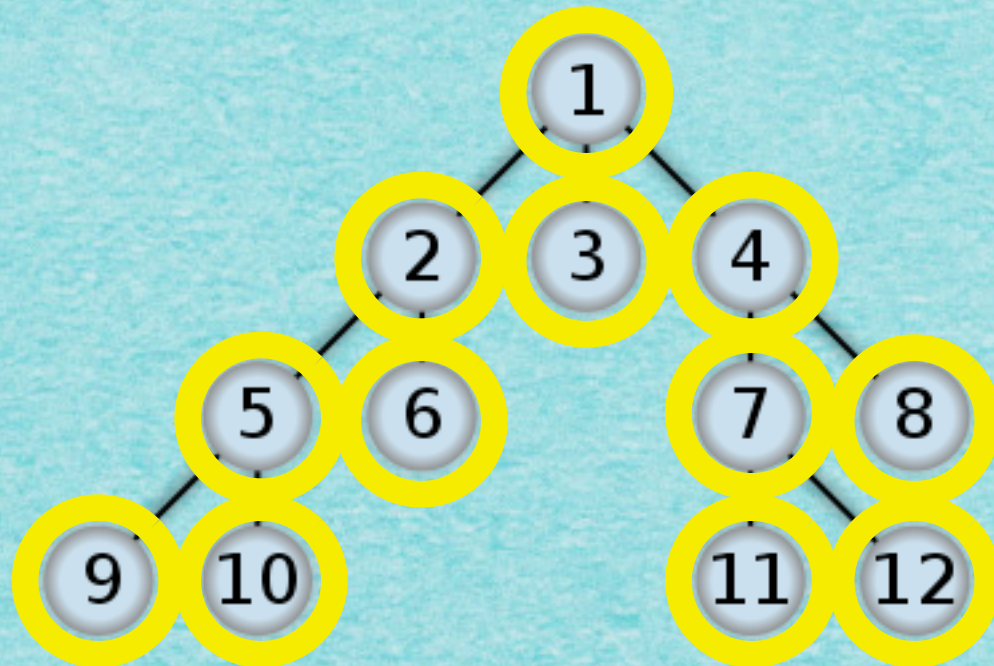
Graphenscan mit WARTESCHLANGE



R:

STOP!

Graphenscan mit WARTESCHLANGE

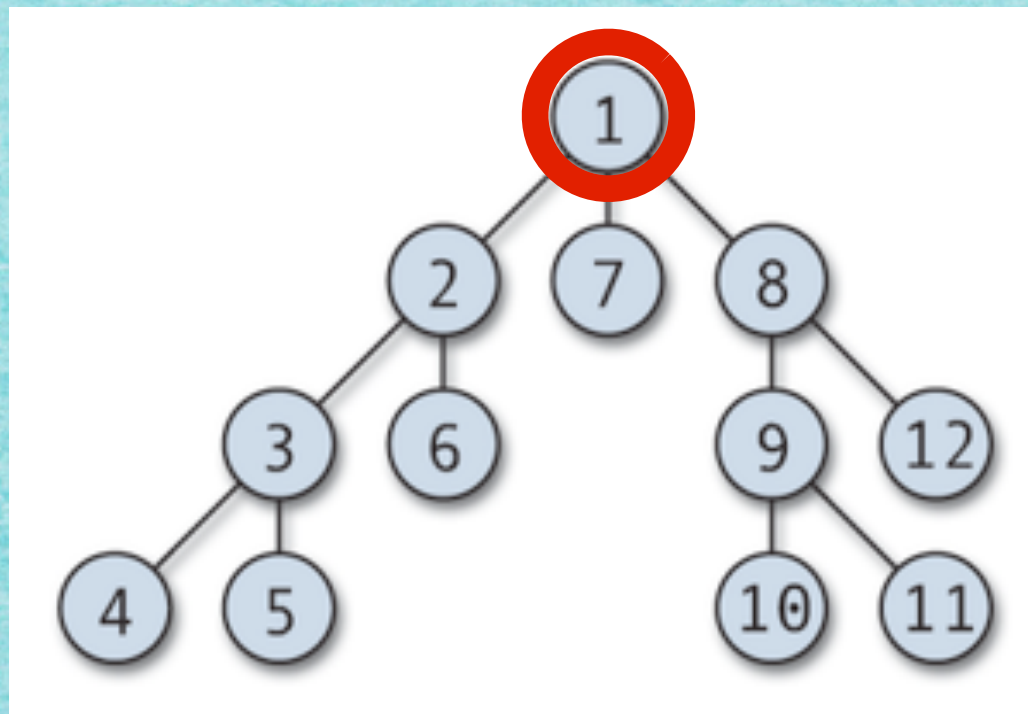


R:

STOP!

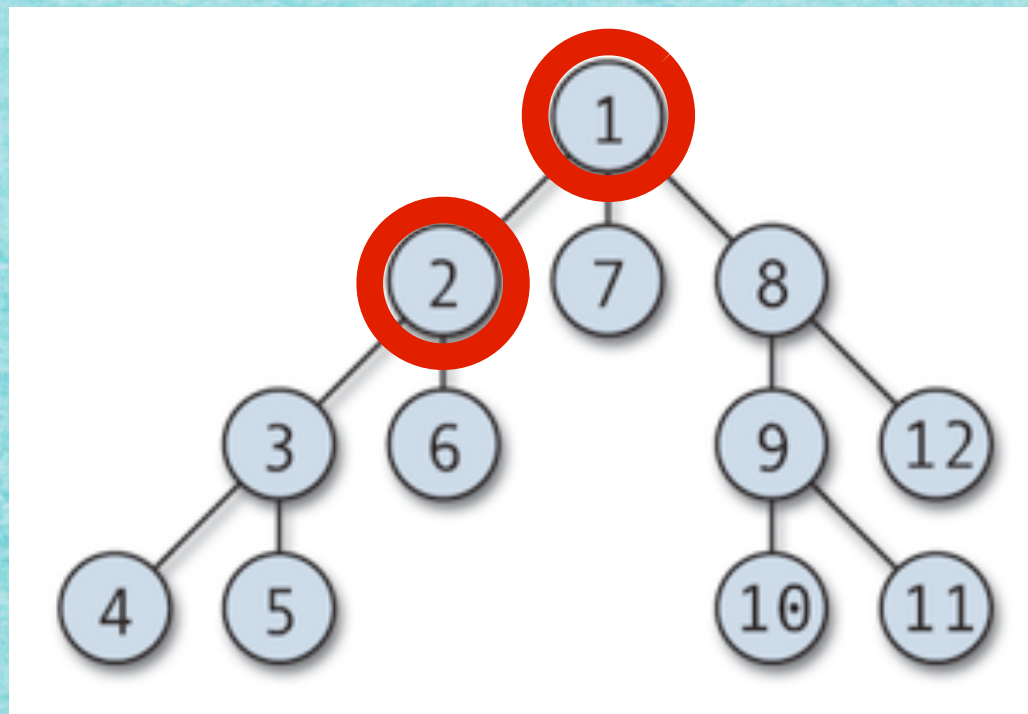
BREITENSUCHE - “BREADTH-FIRST SEARCH” (BFS)

Graphenscan mit STAPEL



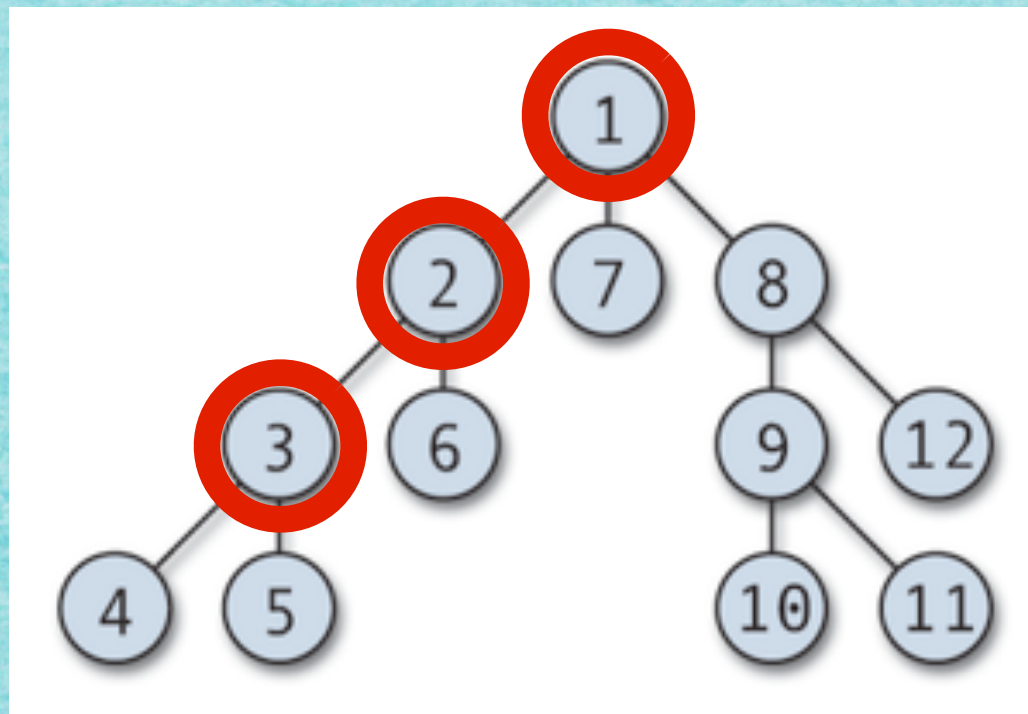
R: 1

Graphenscan mit STAPEL



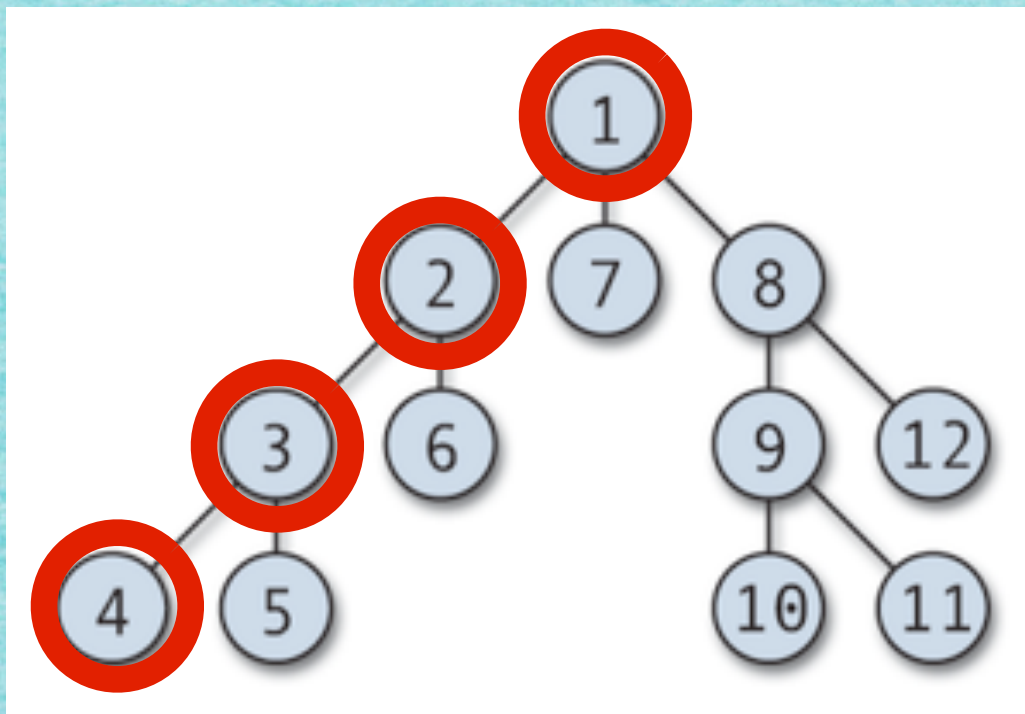
R: 1,2

Graphenscan mit STAPEL



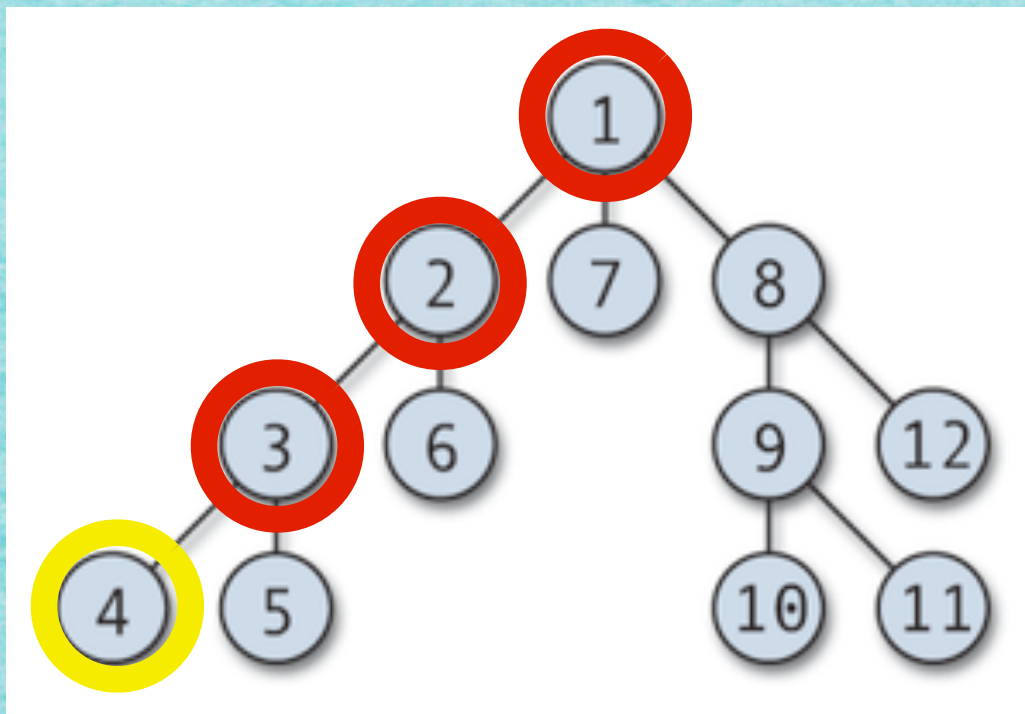
R: 1,2,3

Graphenscan mit STAPEL



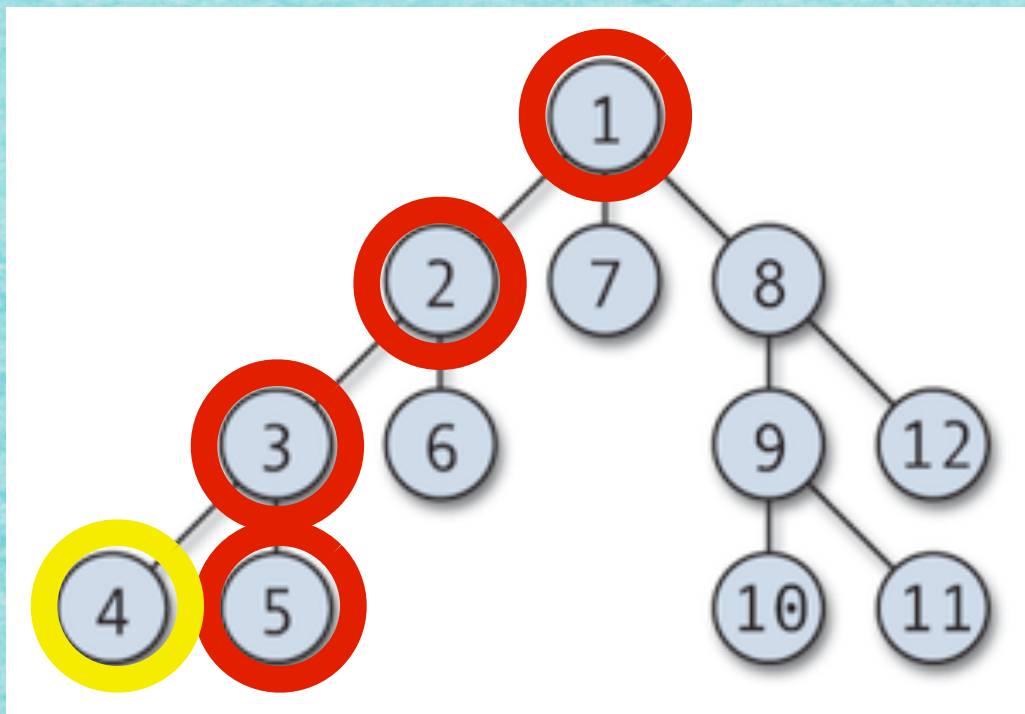
R: 1,2,3,4

Graphenscan mit STAPEL



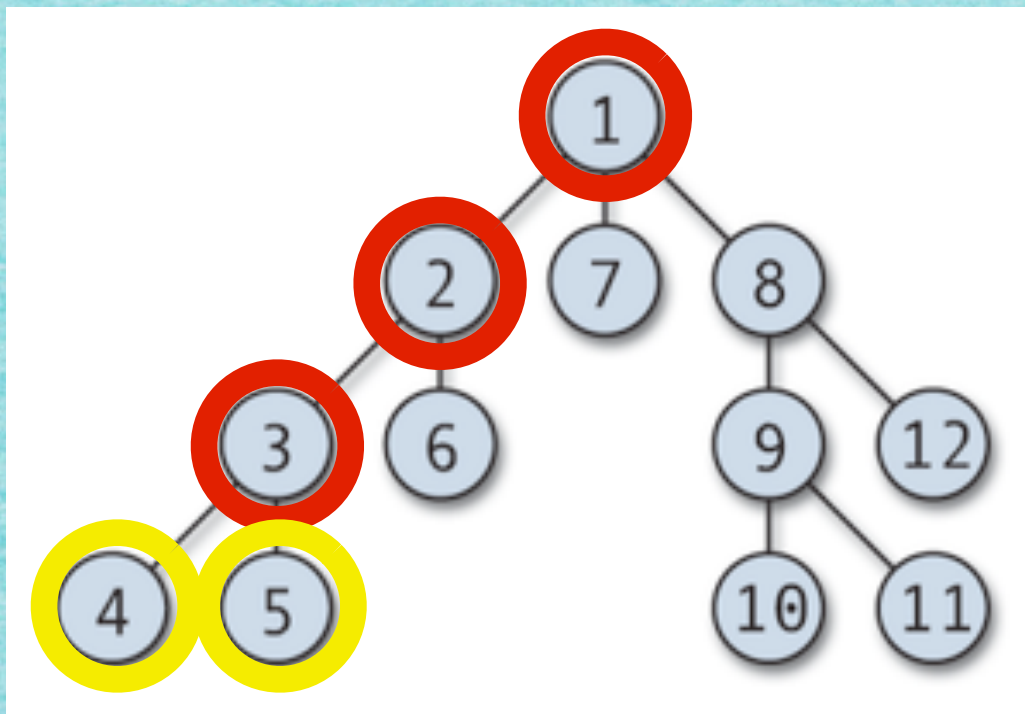
R: 1,2,3

Graphenscan mit STAPEL



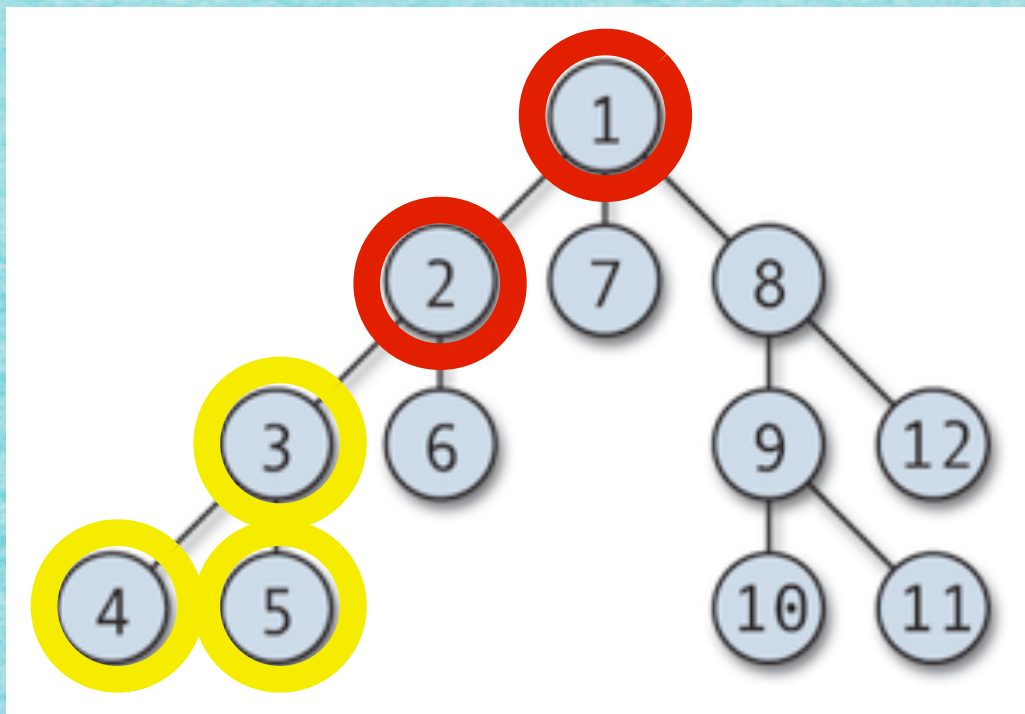
R: 1,2,3,5

Graphenscan mit STAPEL



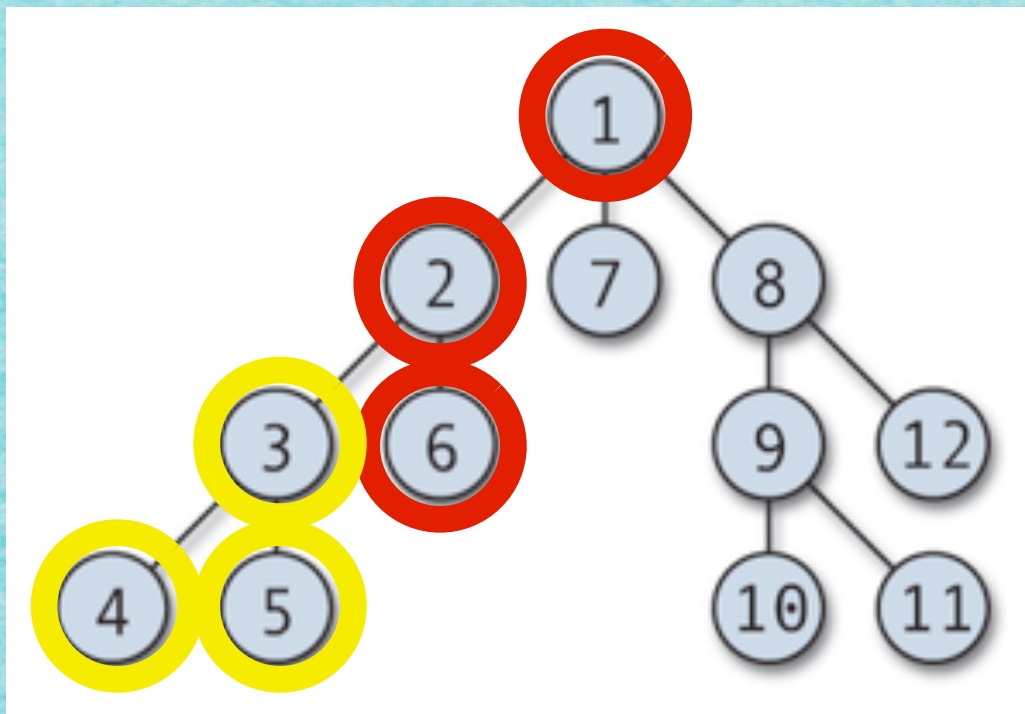
R: 1,2,3,

Graphenscan mit STAPEL



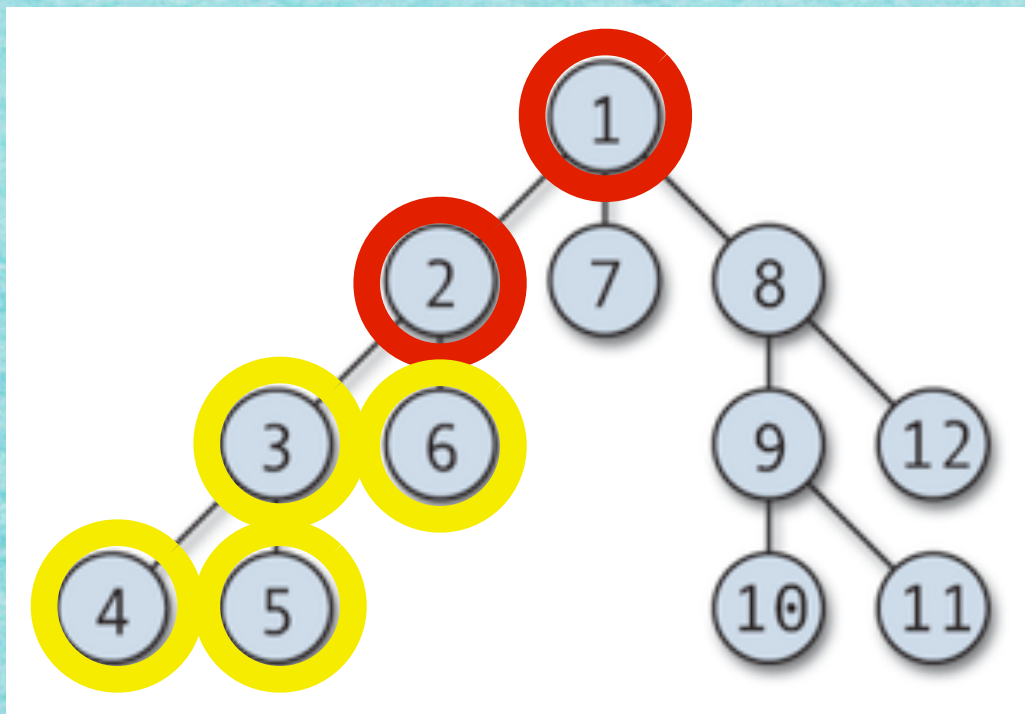
R: 1,2

Graphenscan mit STAPEL



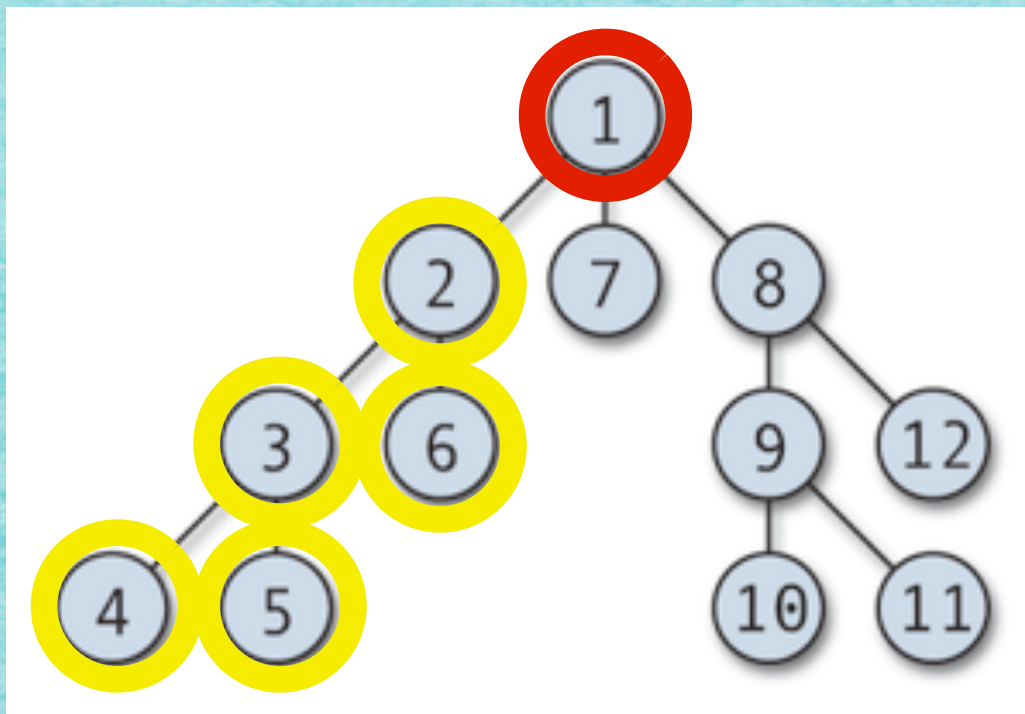
R: 1,2,6

Graphenscan mit STAPEL



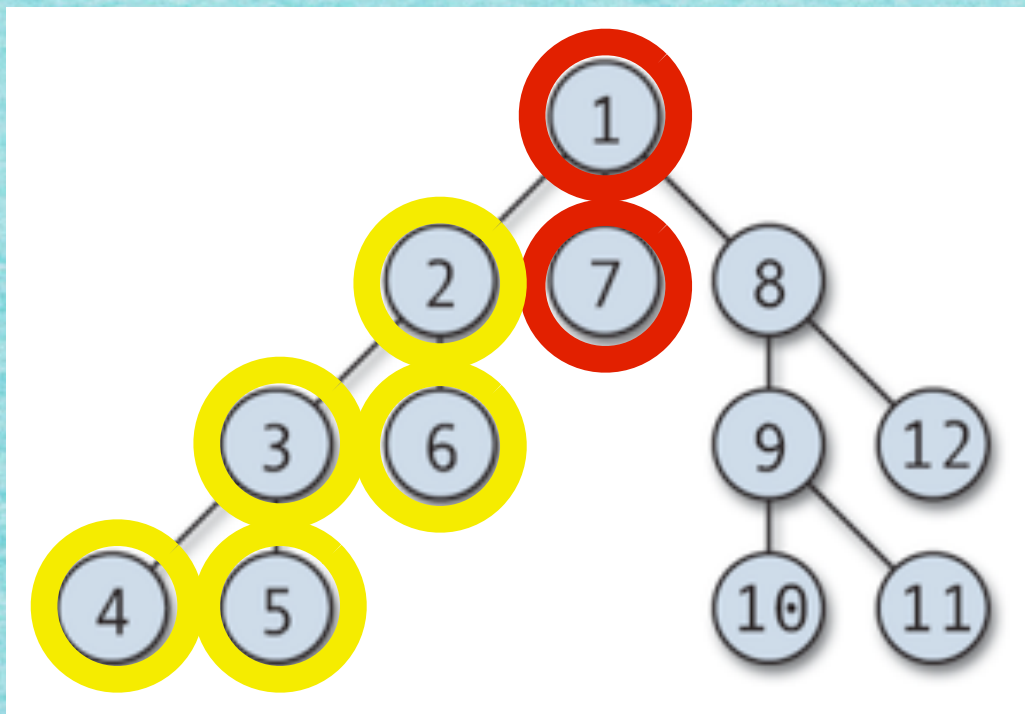
R: 1,2

Graphenscan mit STAPEL



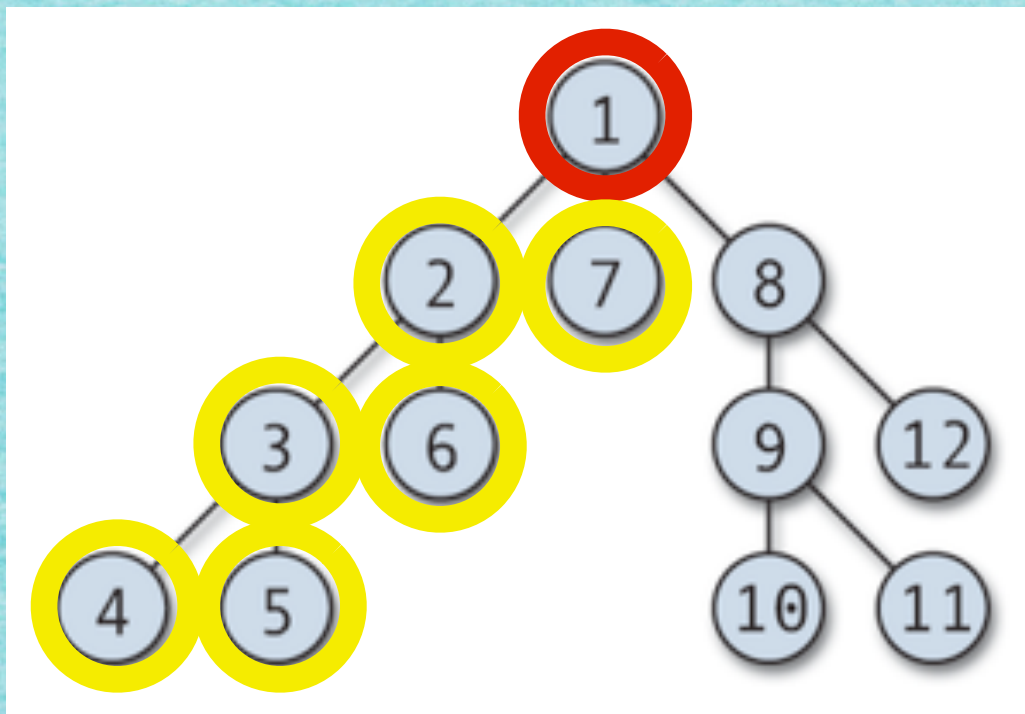
R: 1

Graphenscan mit STAPEL



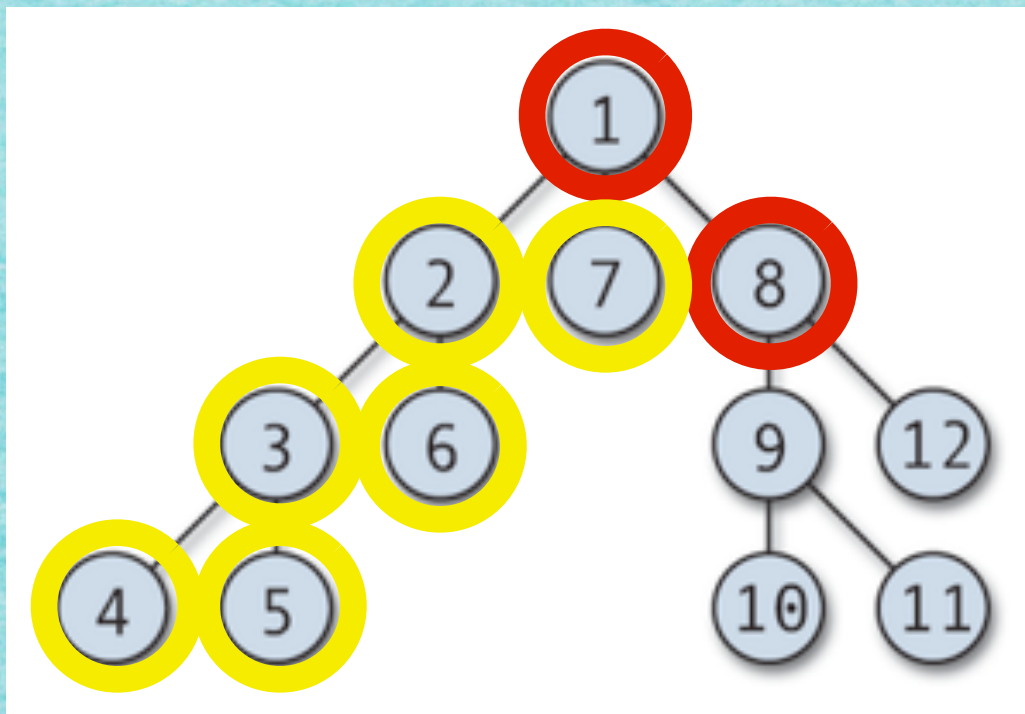
R: 1,7

Graphenscan mit STAPEL



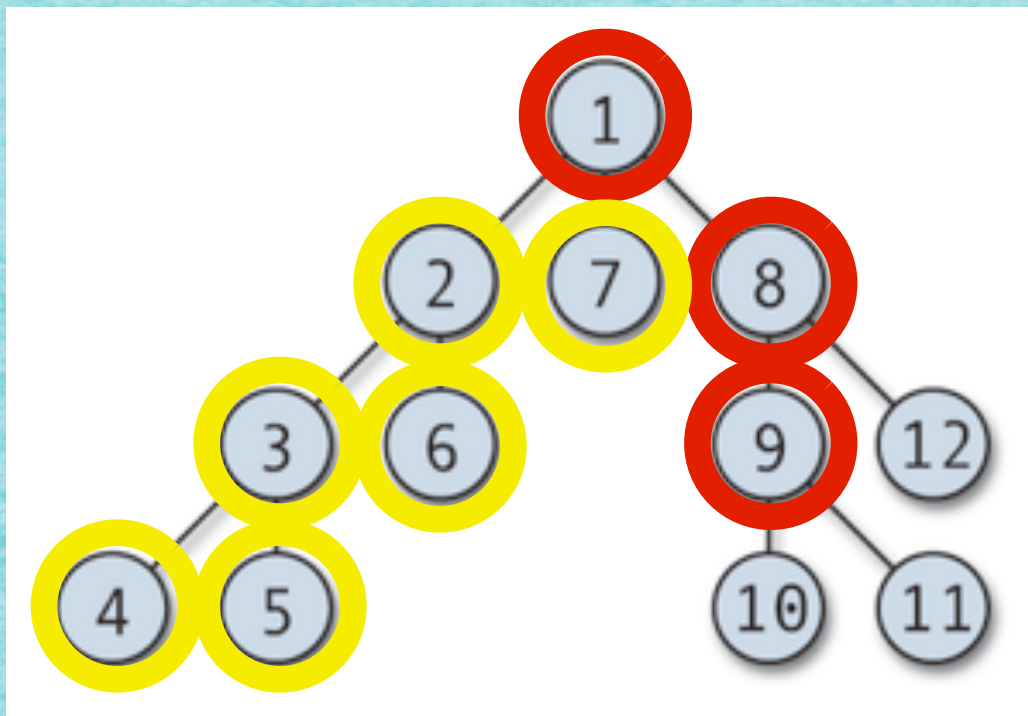
R: 1

Graphenscan mit STAPEL



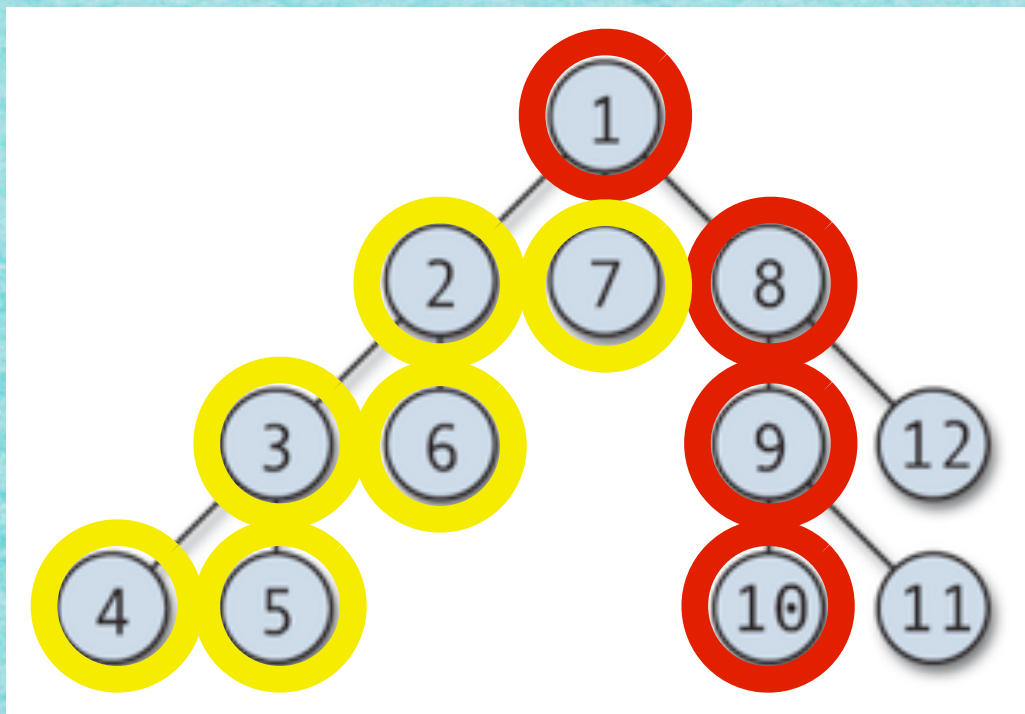
R: 1,8

Graphenscan mit STAPEL



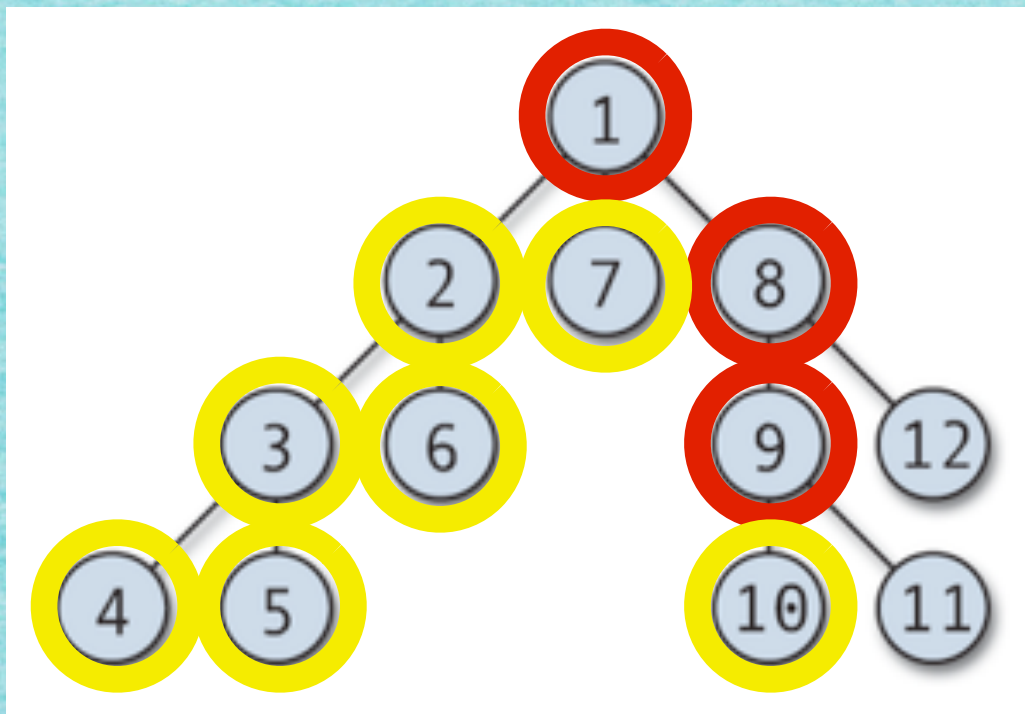
R: 1,8,9

Graphenscan mit STAPEL



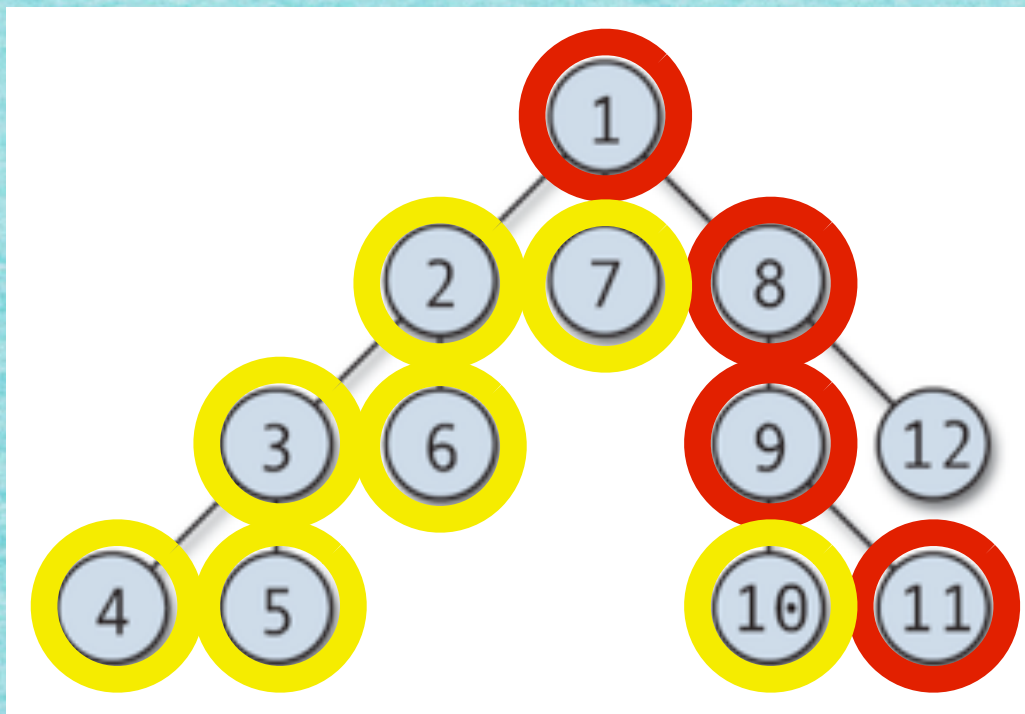
R: 1,8,9,10

Graphenscan mit STAPEL



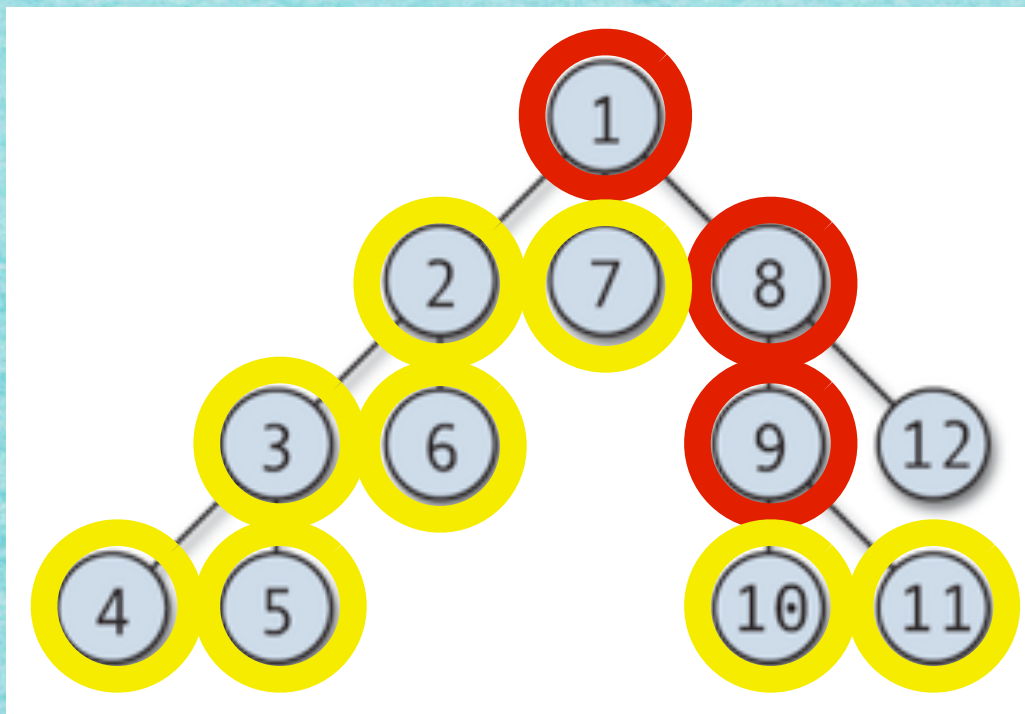
R: 1,8,9

Graphenscan mit STAPEL



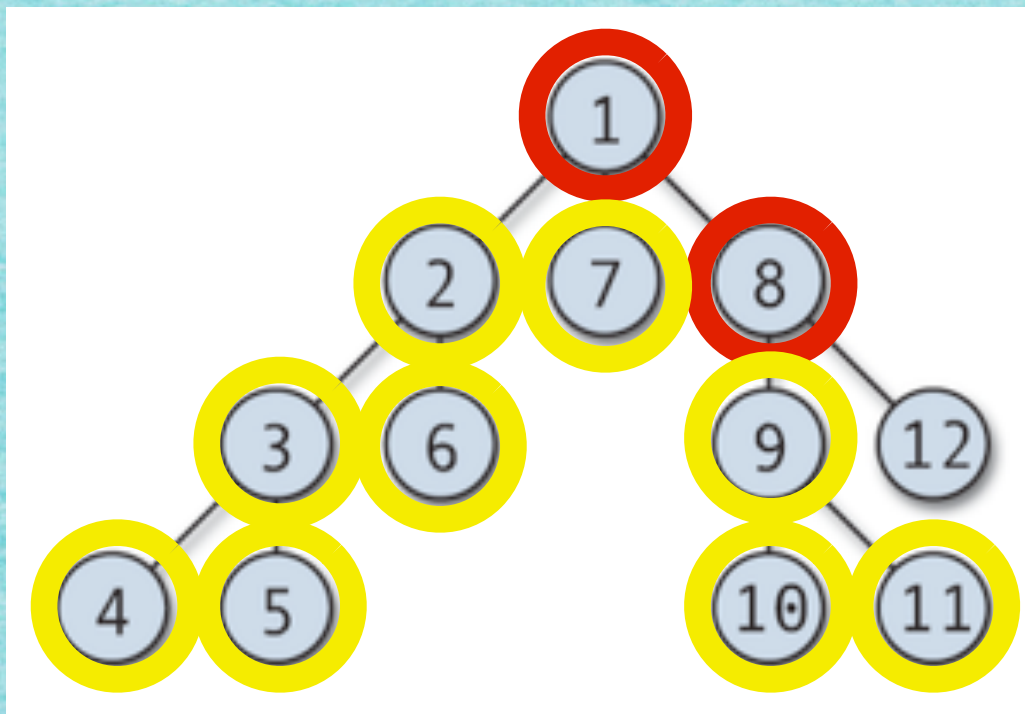
R: 1,8,9,11

Graphenscan mit STAPEL



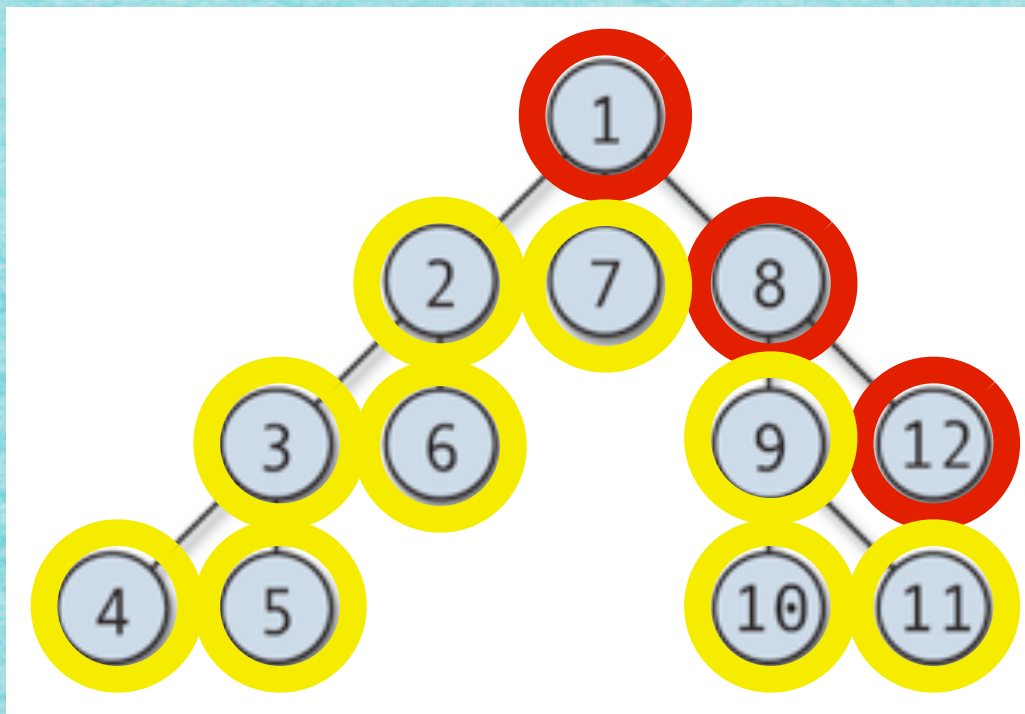
R: 1,8,9

Graphenscan mit STAPEL



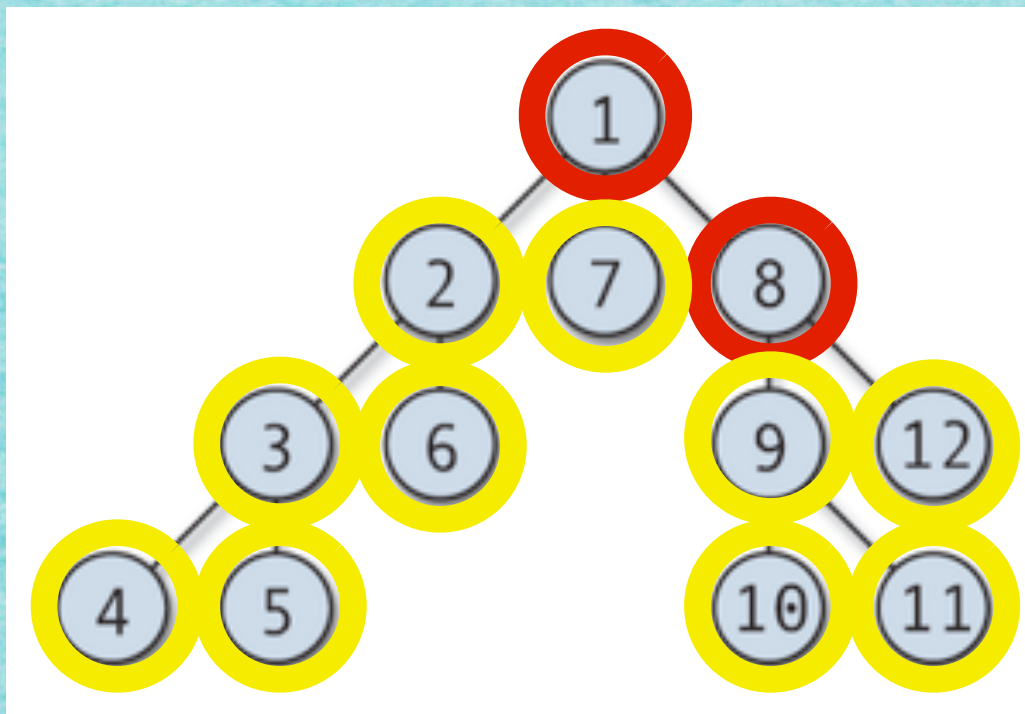
R: 1,8

Graphenscan mit STAPEL



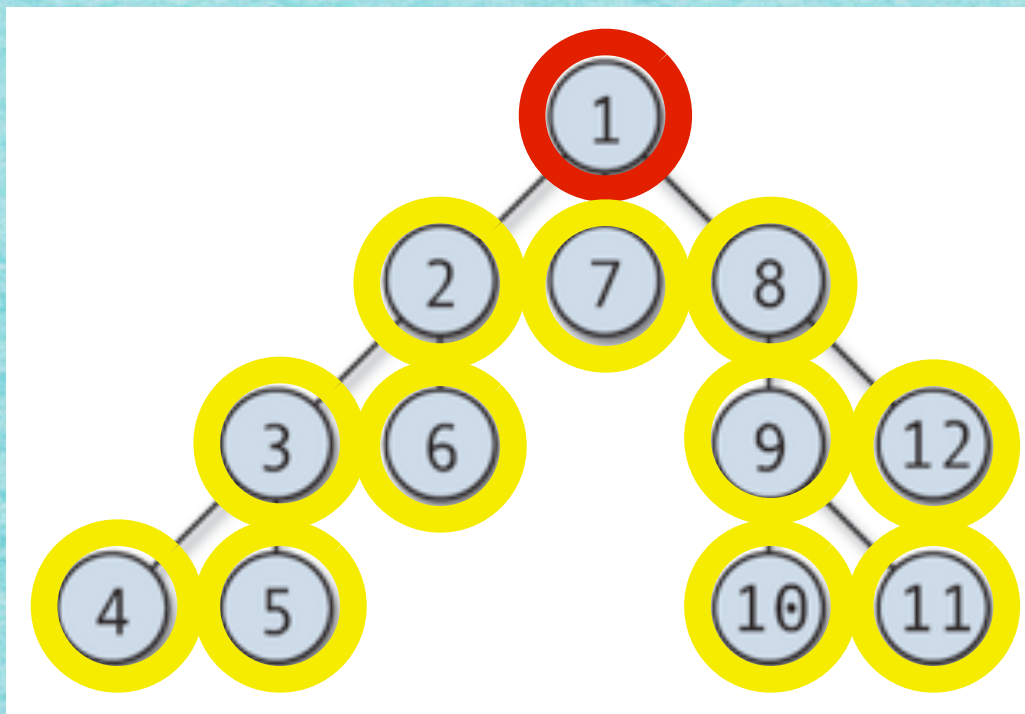
R: 1,8,12

Graphenscan mit STAPEL



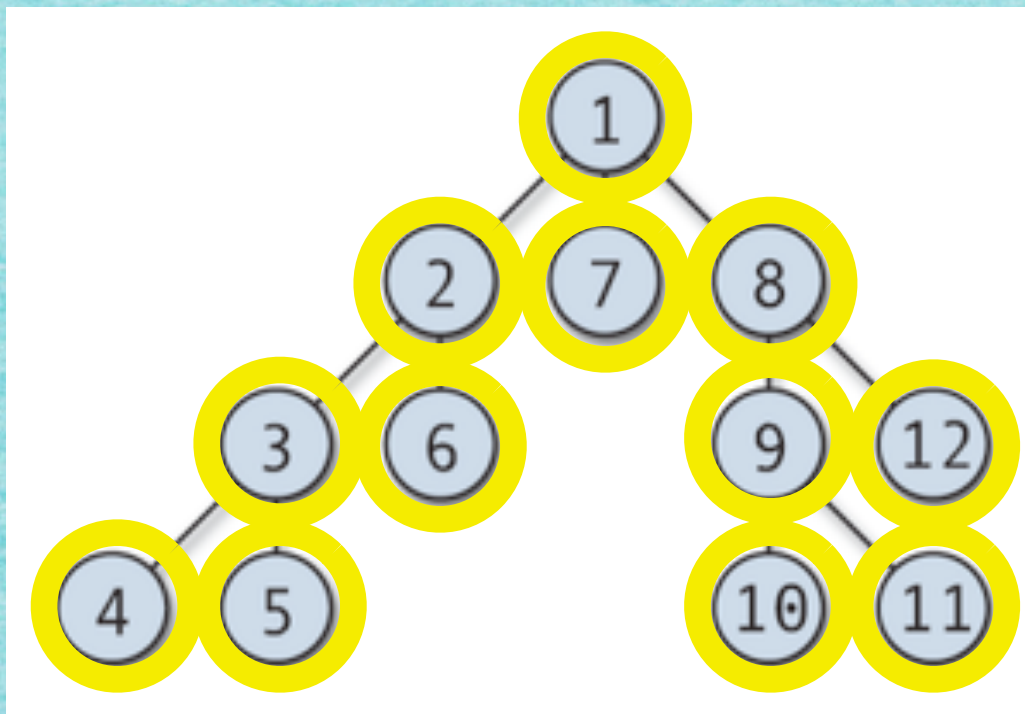
R: 1,8

Graphenscan mit STAPEL



R: 1

Graphenscan mit STAPEL

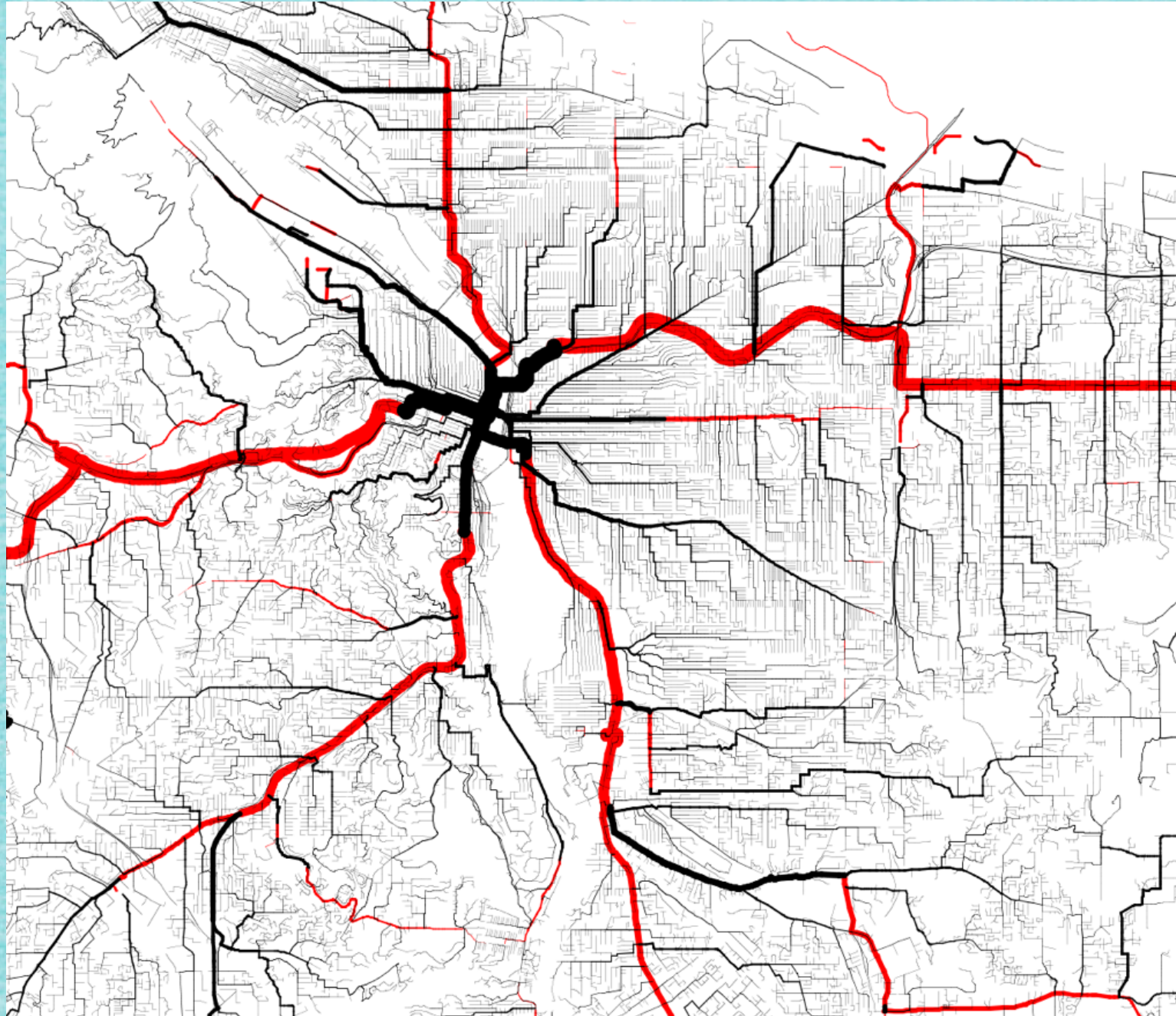


R:

STOP!

TIEFENSUCHE - “DEPTH-FIRST SEARCH” (DFS)

Breitensuche liefert kürzeste Wege von einer Quelle aus



Viele "Sammler"

Auf die Schnelle mit der Welle



Auf die Schnelle mit der Welle



Auf die Schnelle mit der Welle

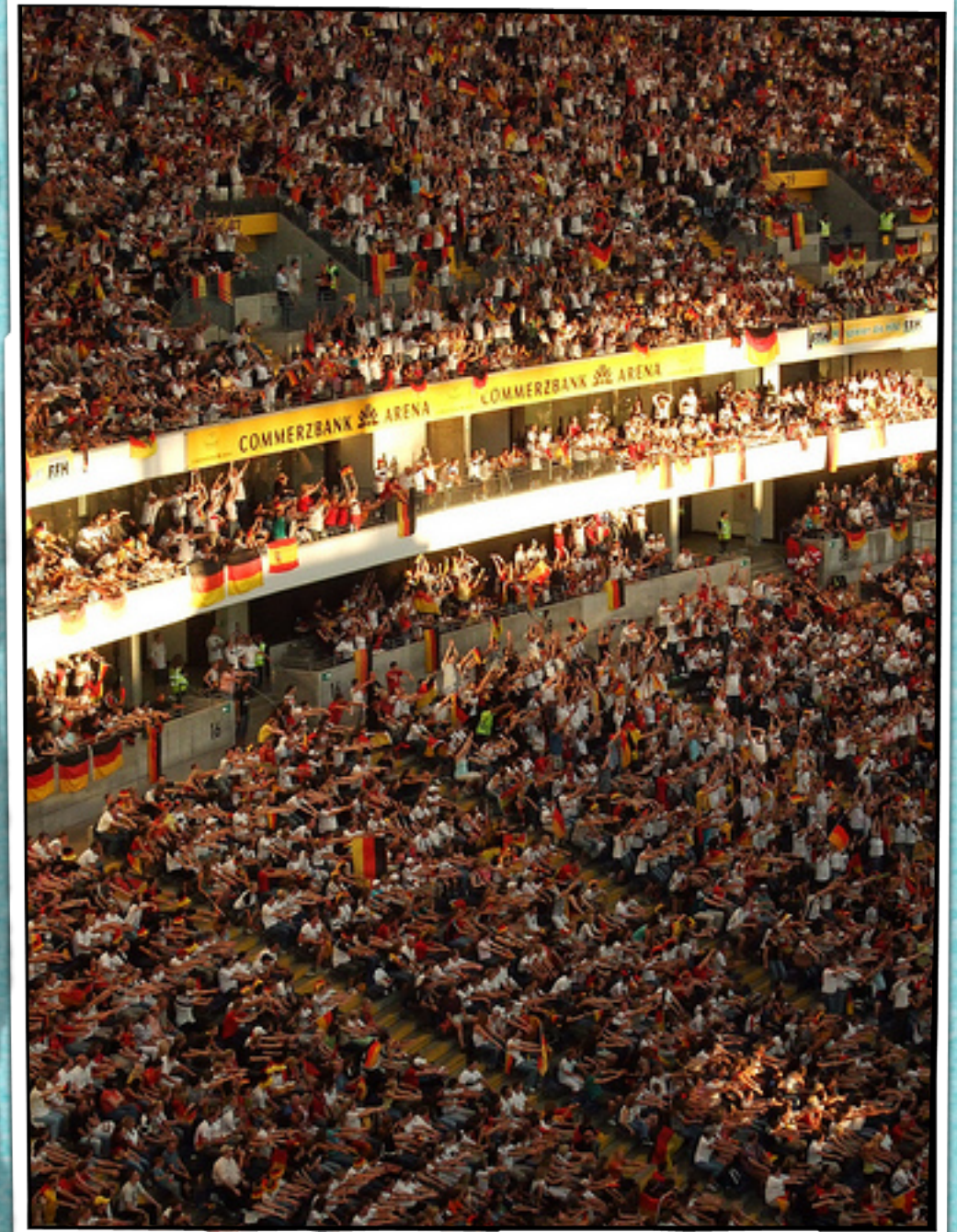
A. LOS bei „NULL“

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



Auf die Schnelle mit der Welle

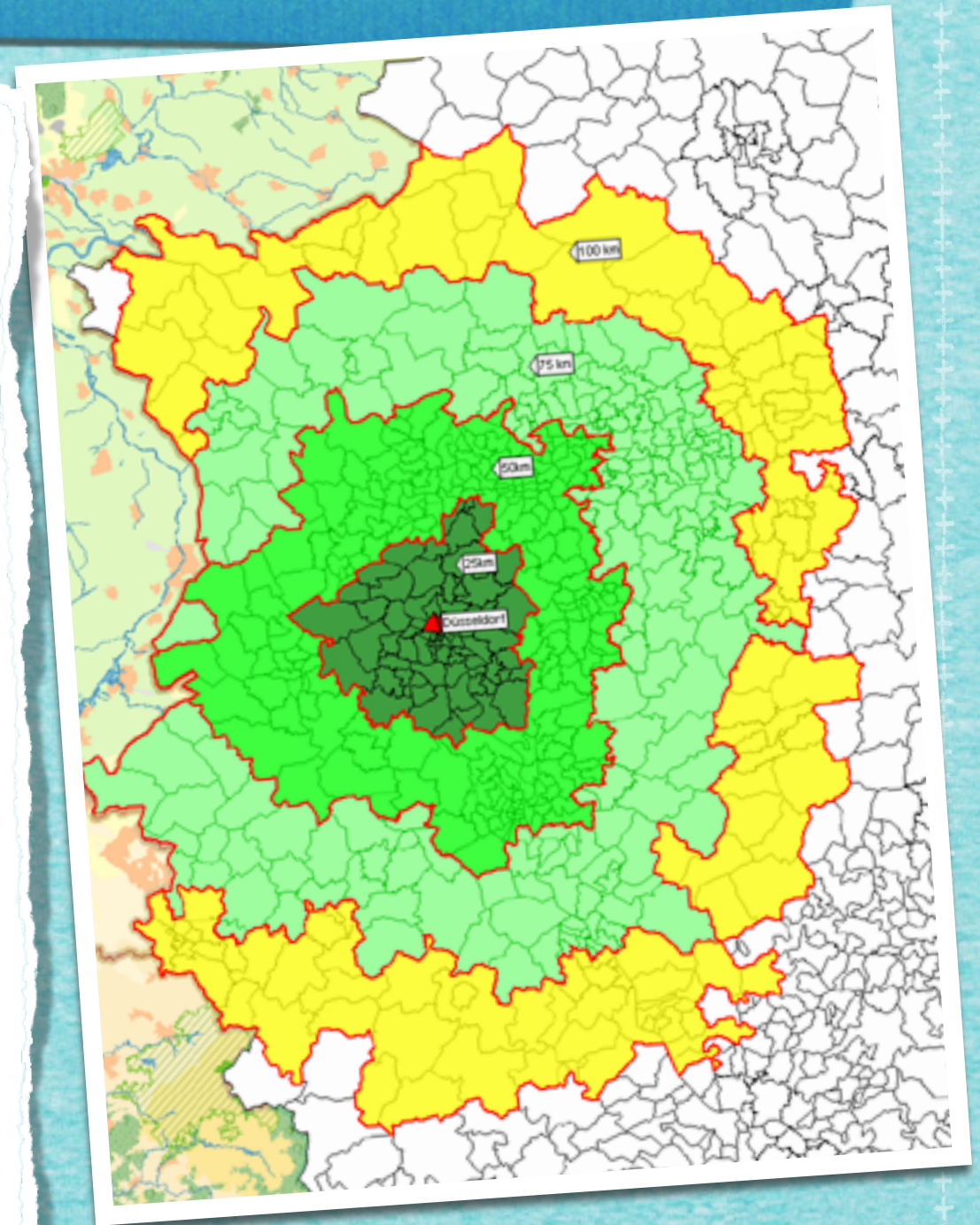
A. LOS bei „NULL“

B. Bis „ANGEKOMMEN!“:

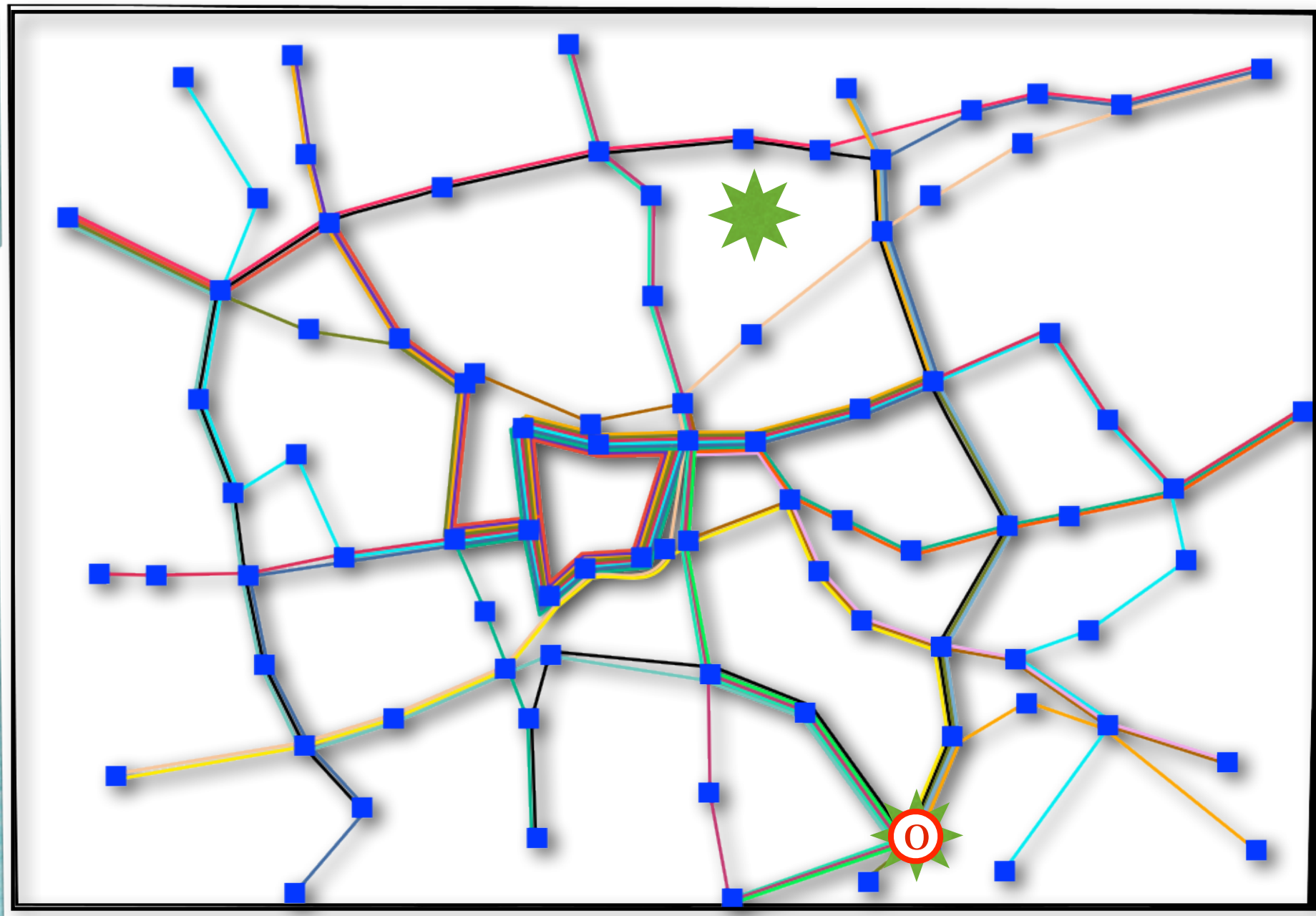
- Solange du noch nicht aufgestanden warst:
 - ▶ Wenn ein oder mehrere direkte Nachbarn aufstehen:
 1. Einen dieser Nachbarn merken
 2. In der nächsten Runde:
 - 2.1. aufstehen
 - 2.2. Zahl merken
 3. In der übernächsten Runde hinsetzen

C. Nach „ANGEKOMMEN!“:

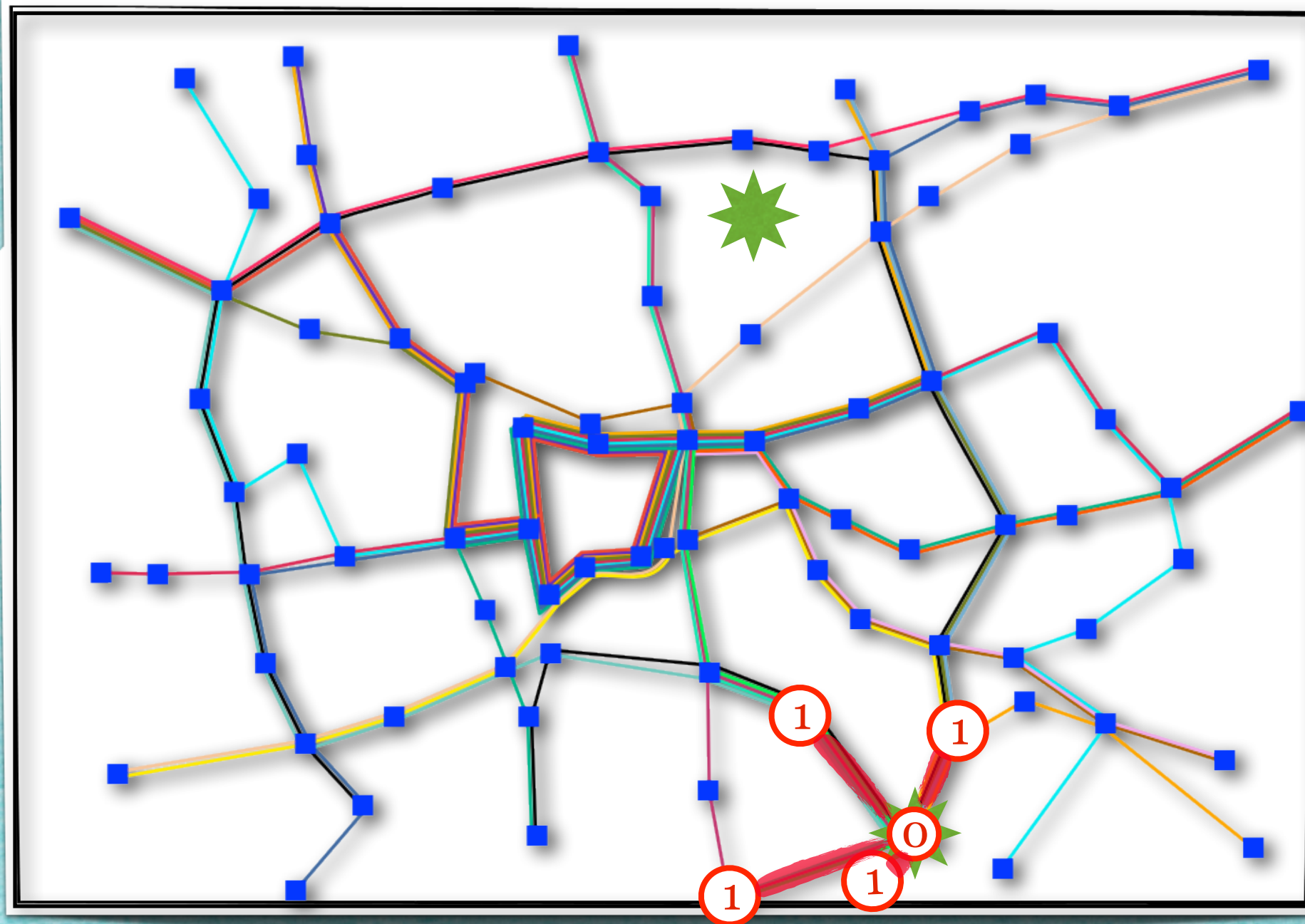
- Auf gemerkten Nachbarn zeigen



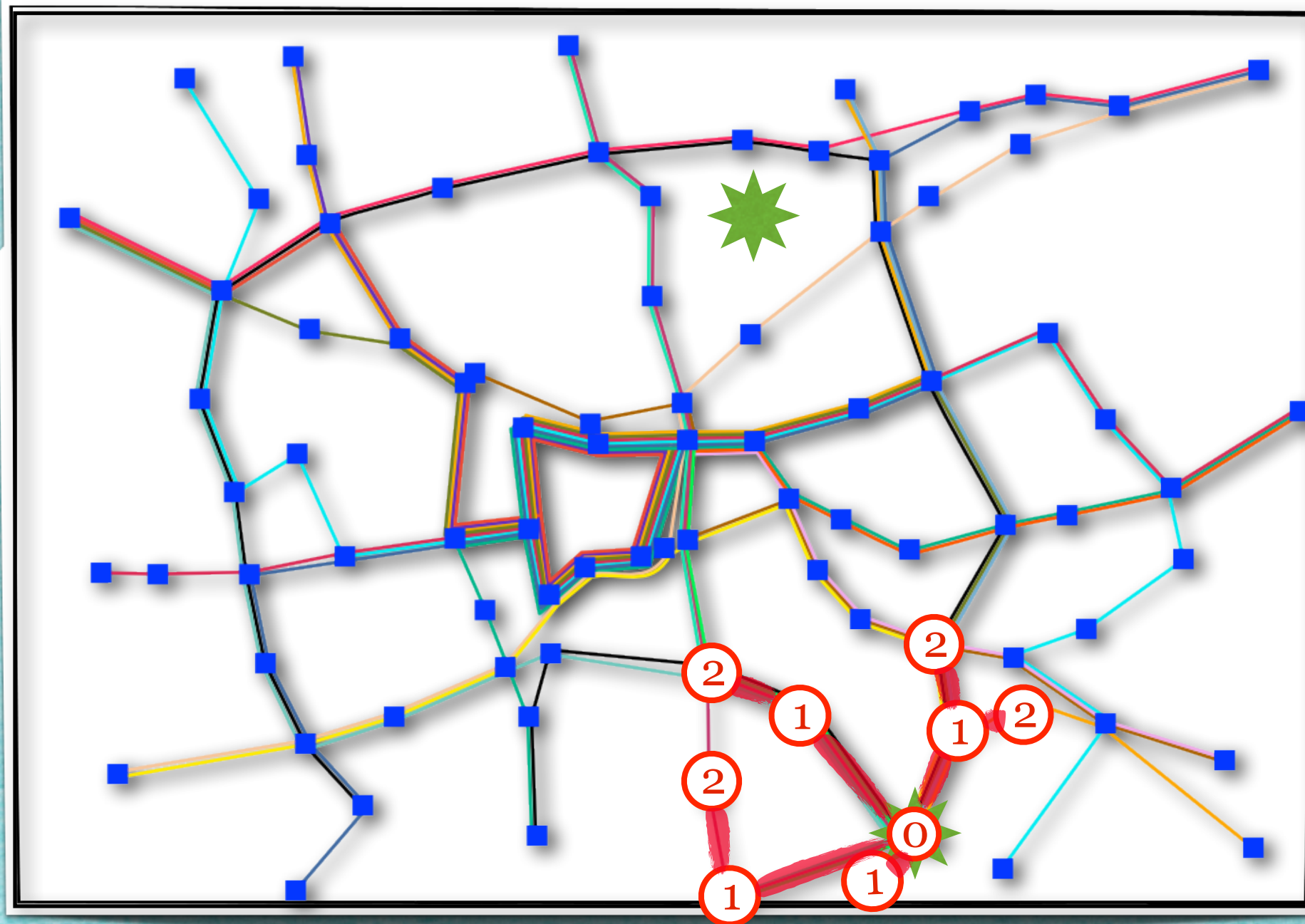
Wellenreiten in Graphen



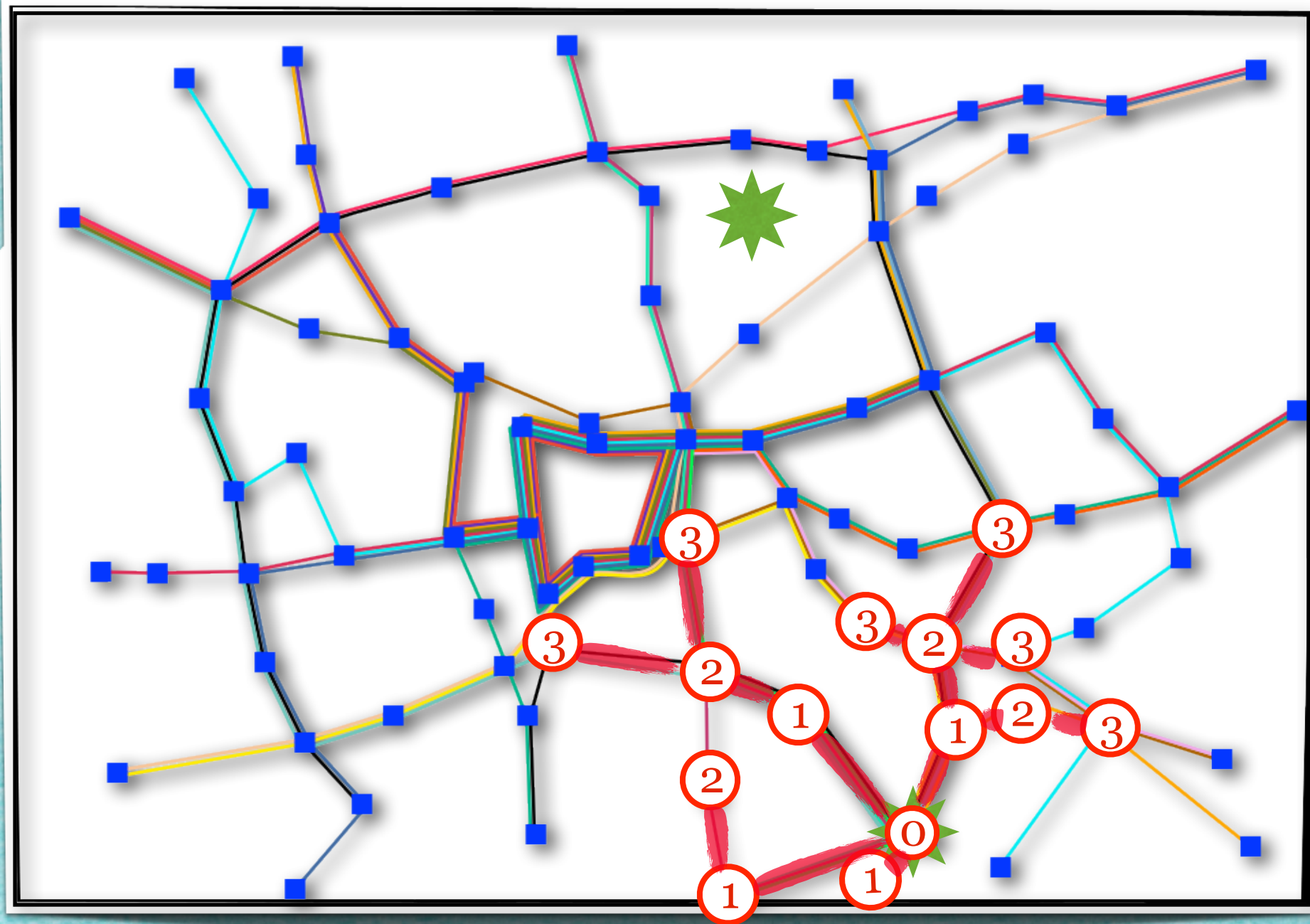
Wellenreiten in Graphen



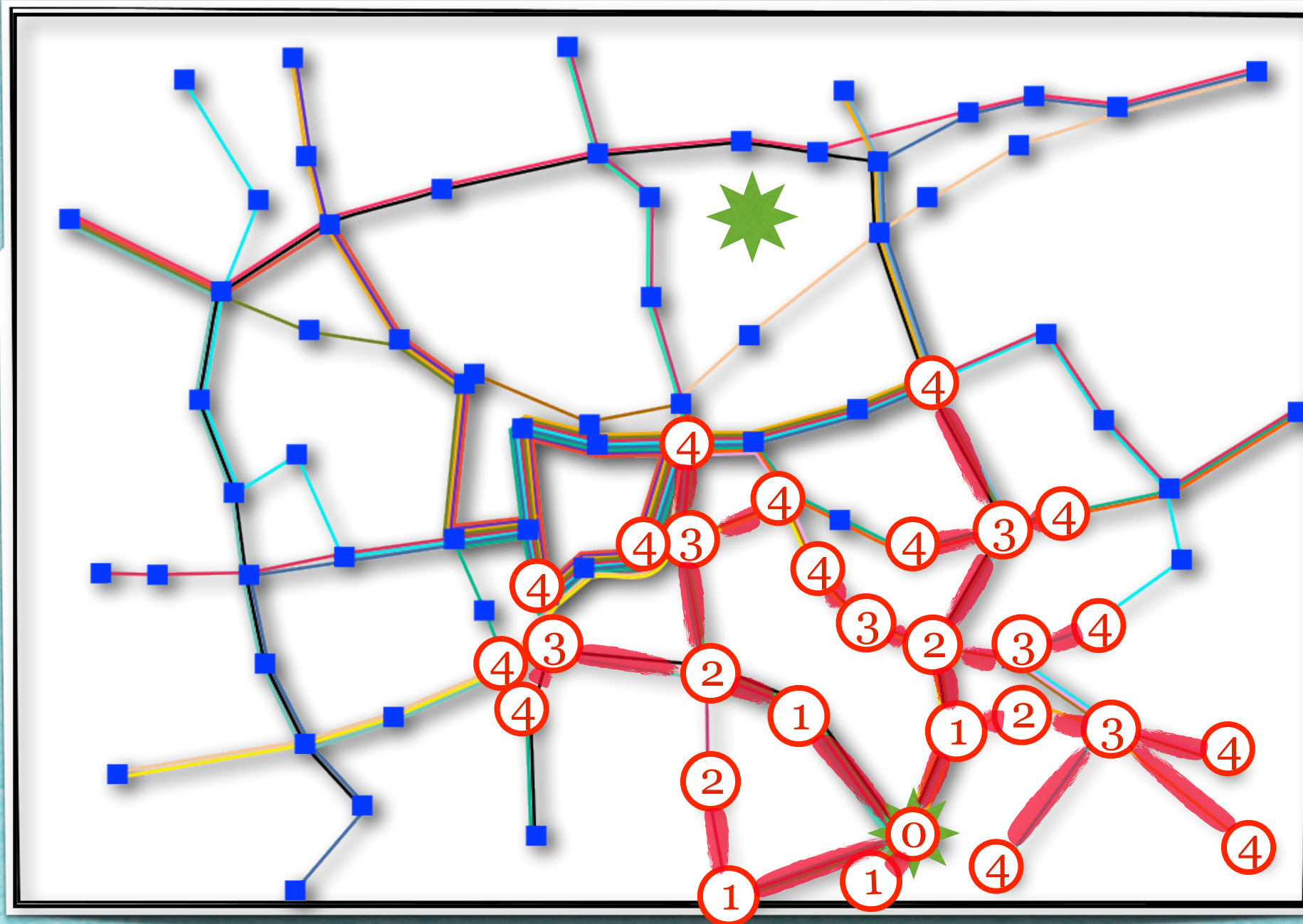
Wellenreiten in Graphen



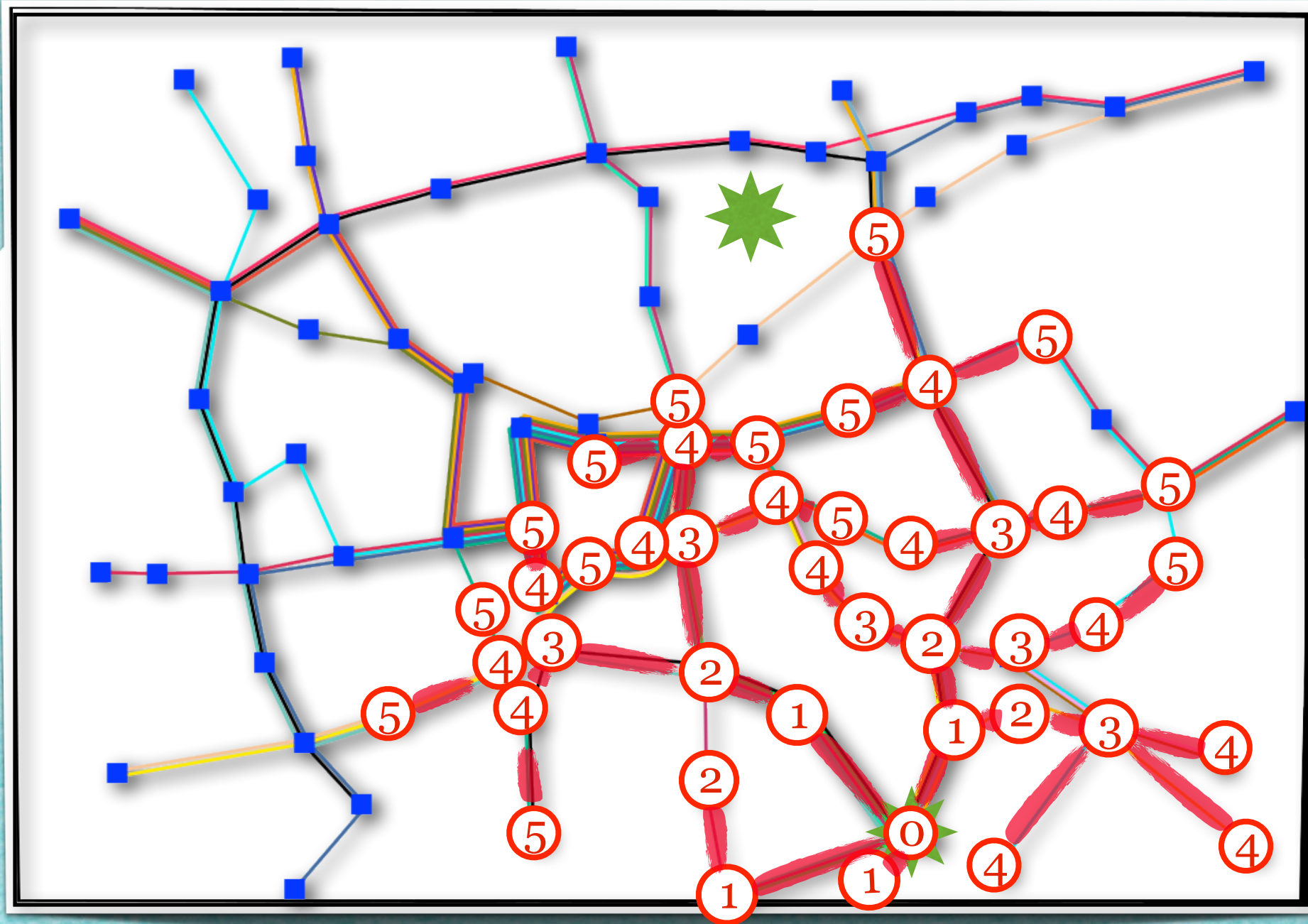
Wellenreiten in Graphen



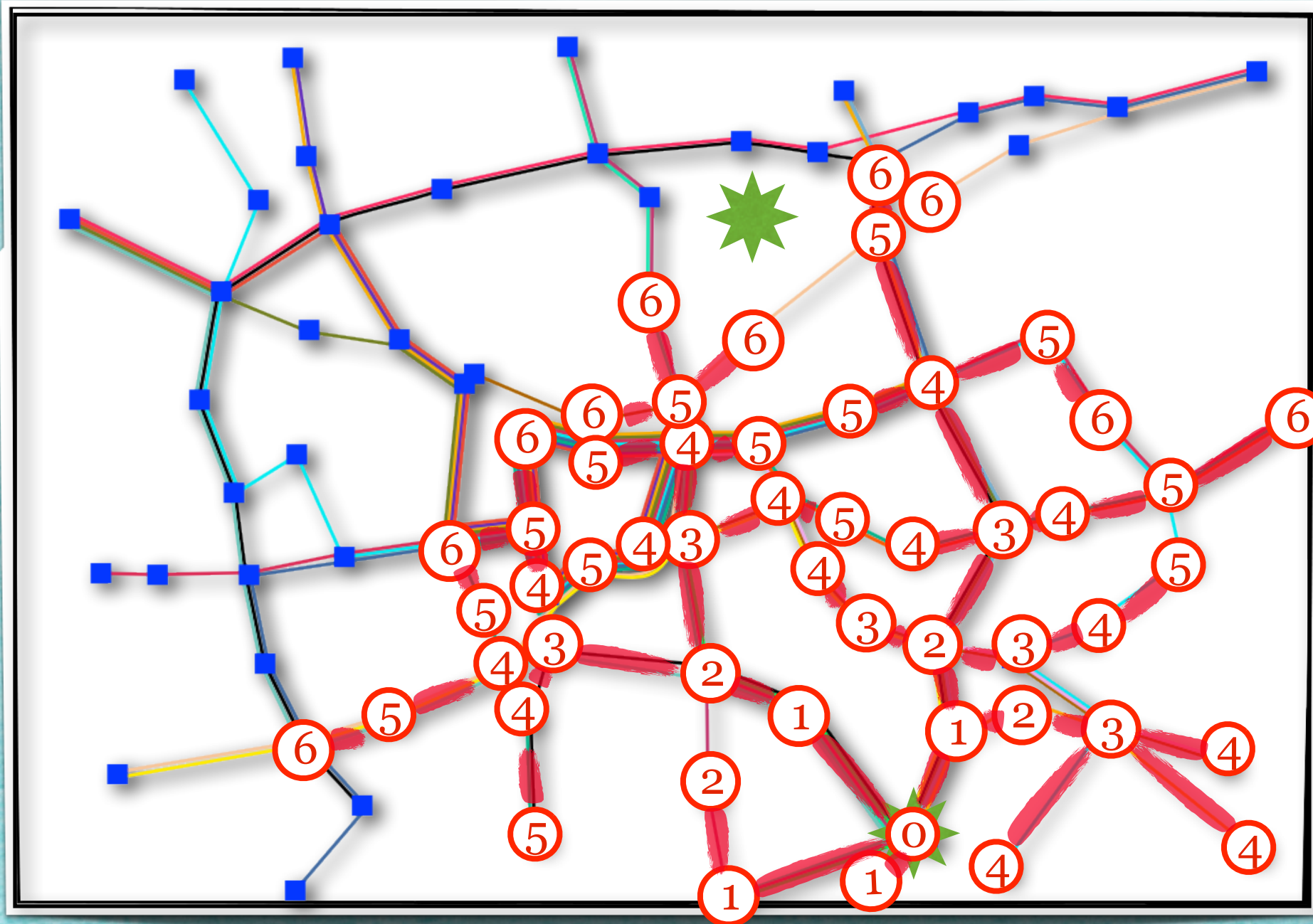
Wellenreiten in Graphen



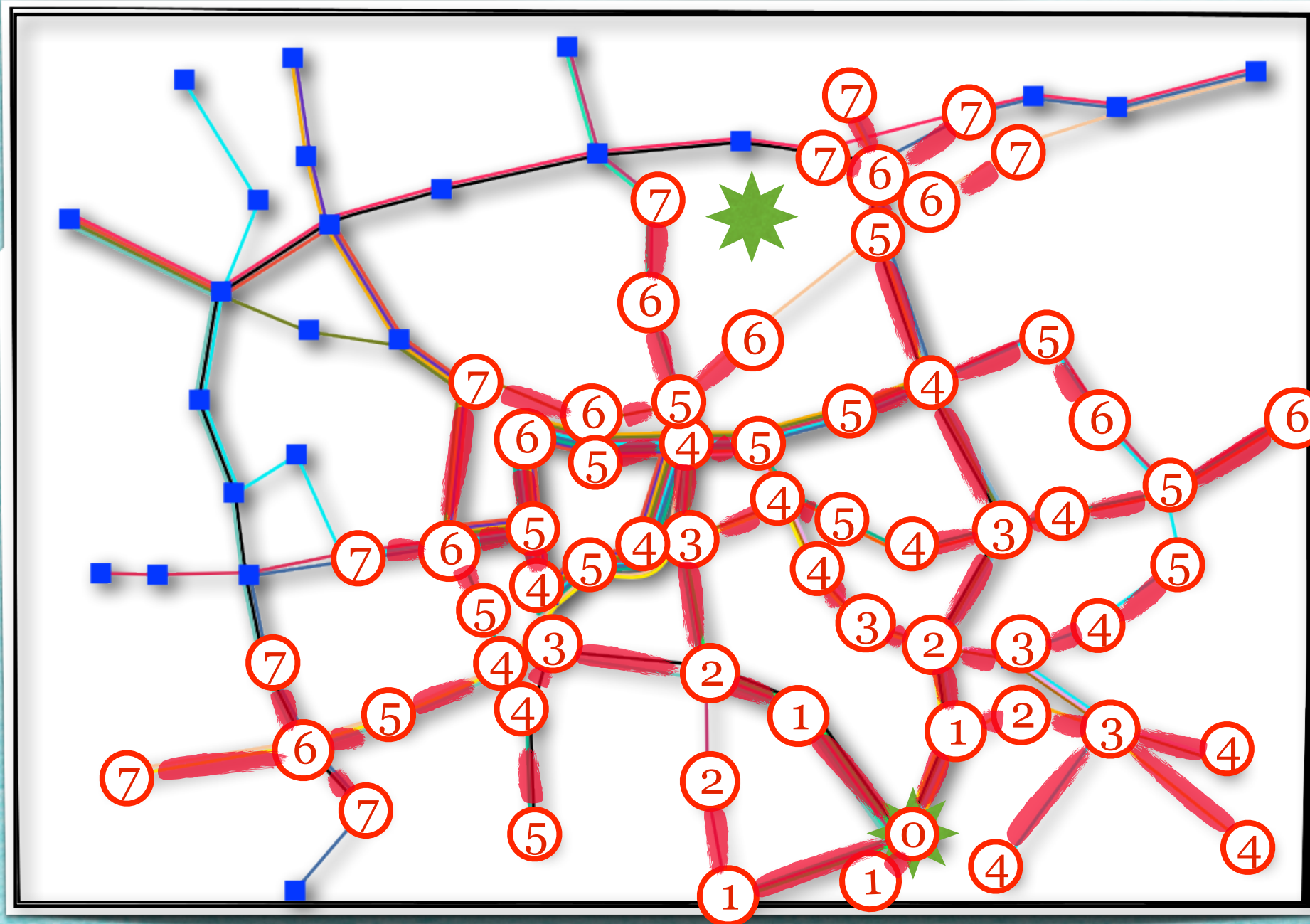
Wellenreiten in Graphen



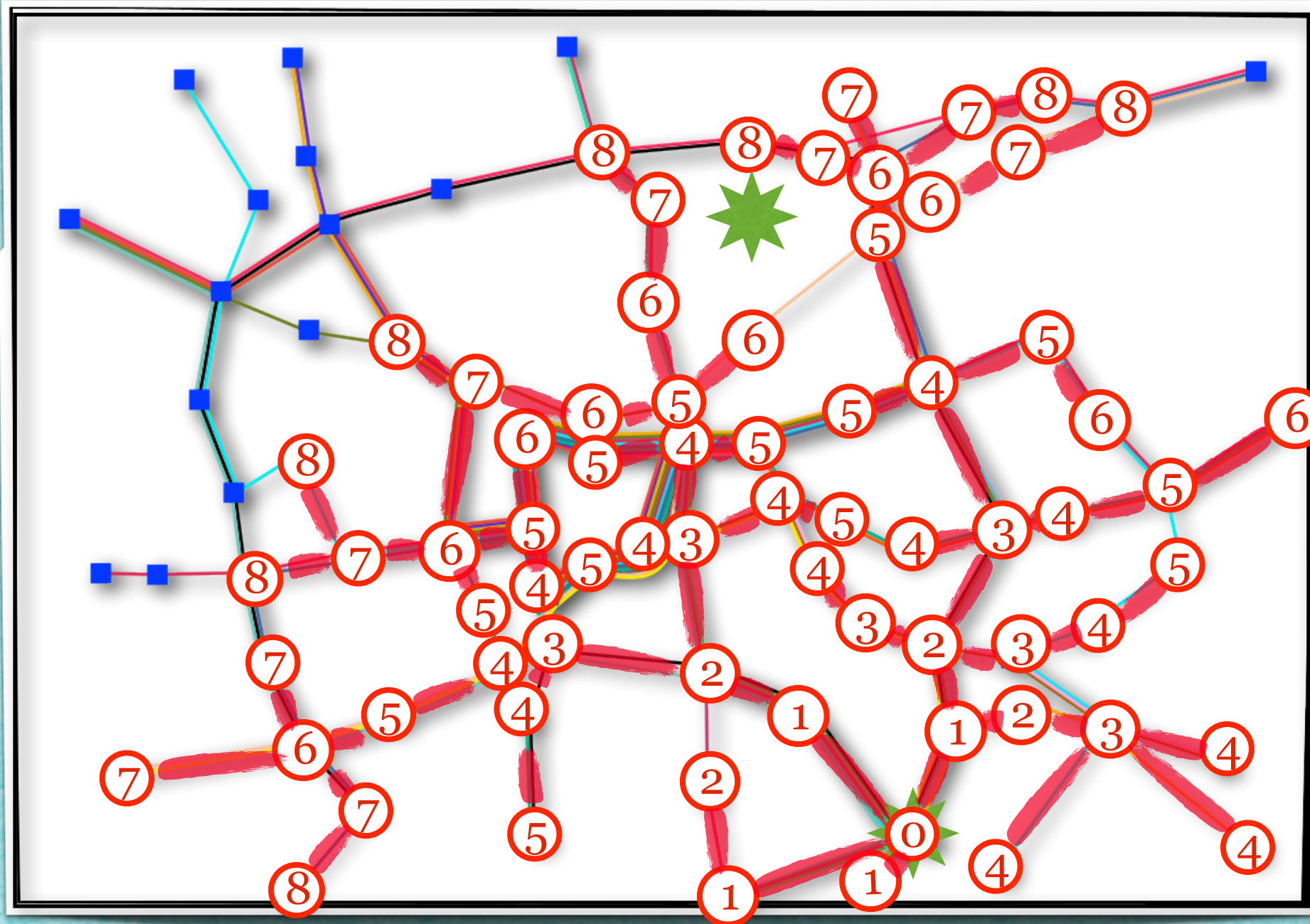
Wellenreiten in Graphen



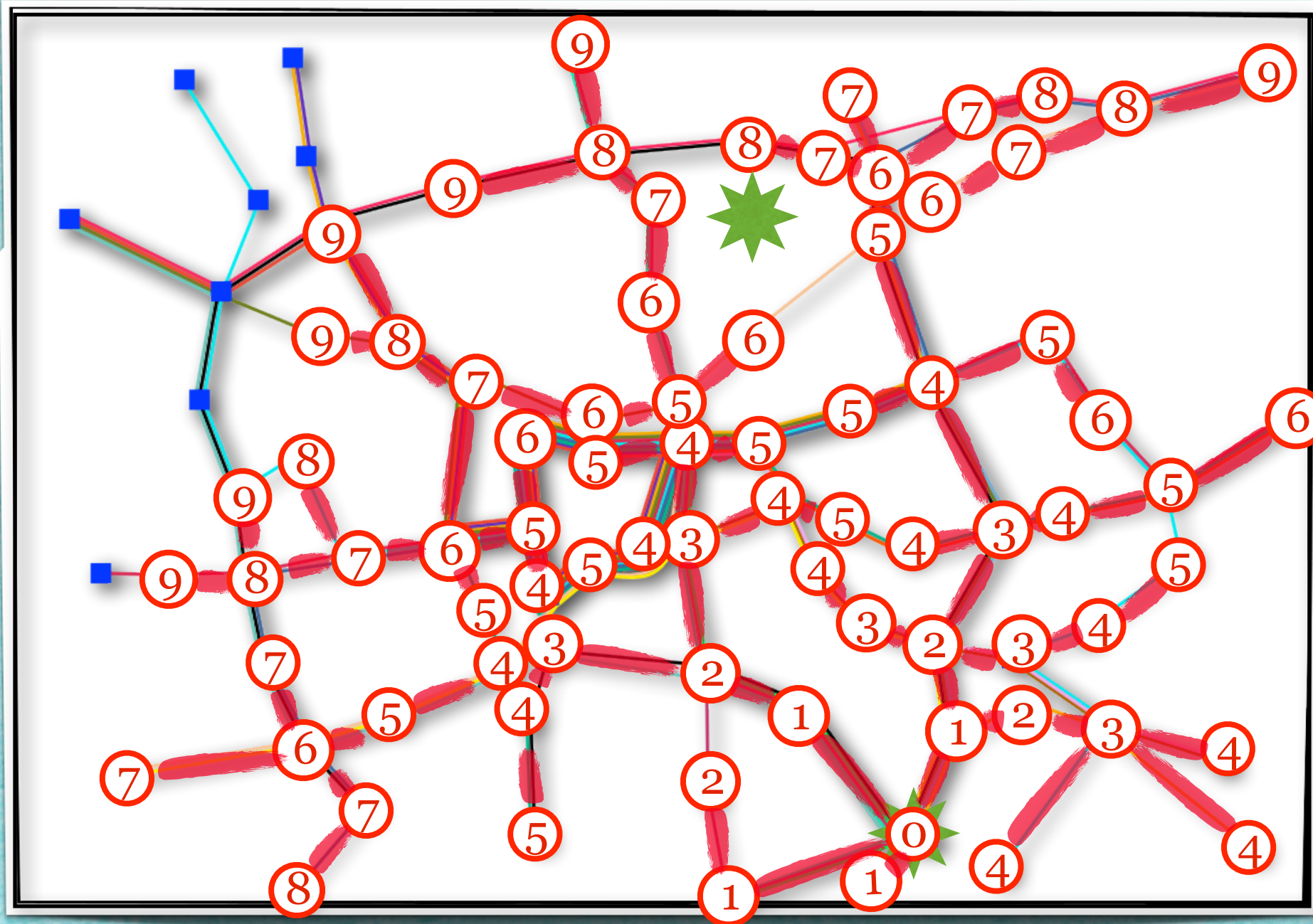
Wellenreiten in Graphen



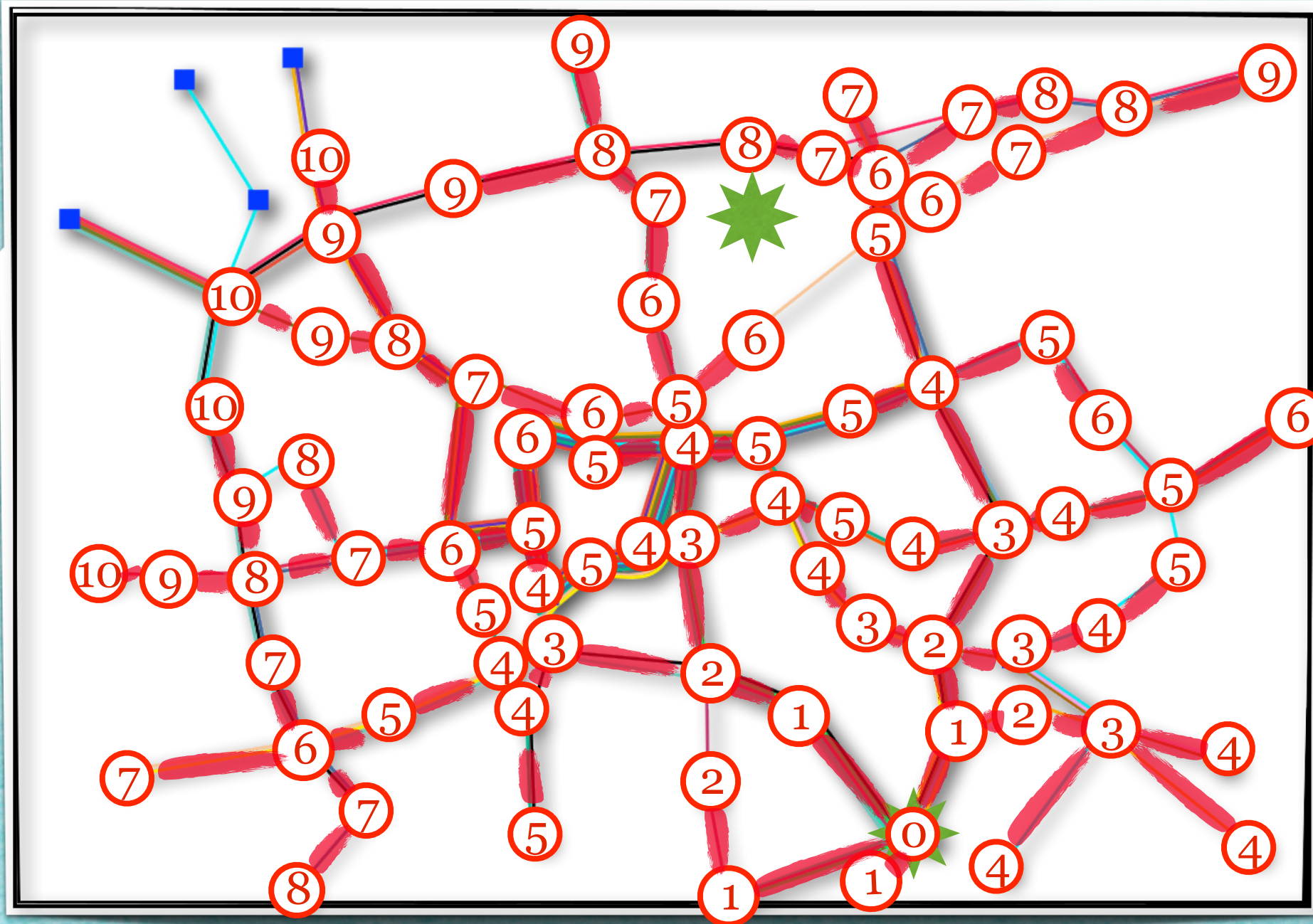
Wellenreiten in Graphen



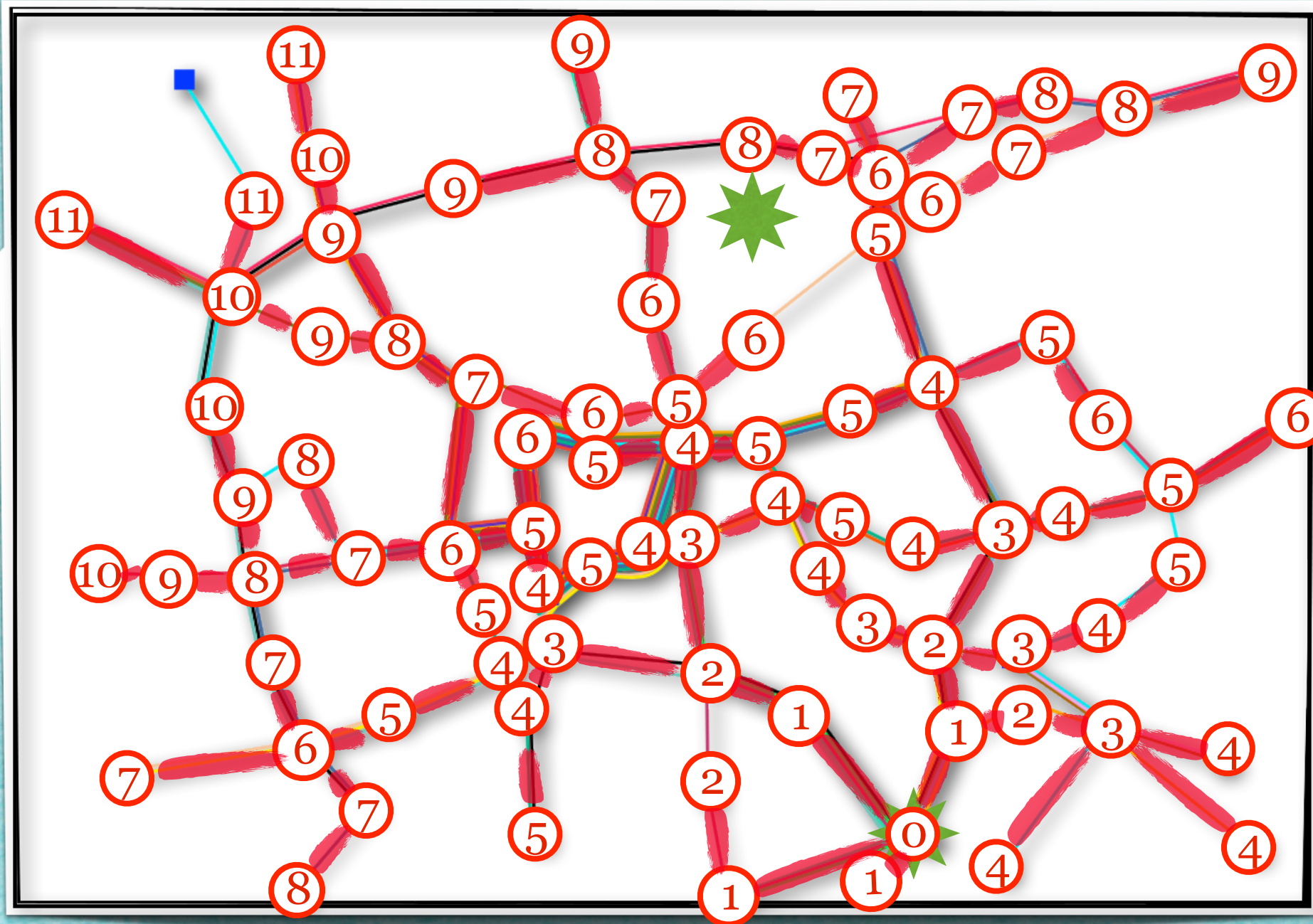
Wellenreiten in Graphen



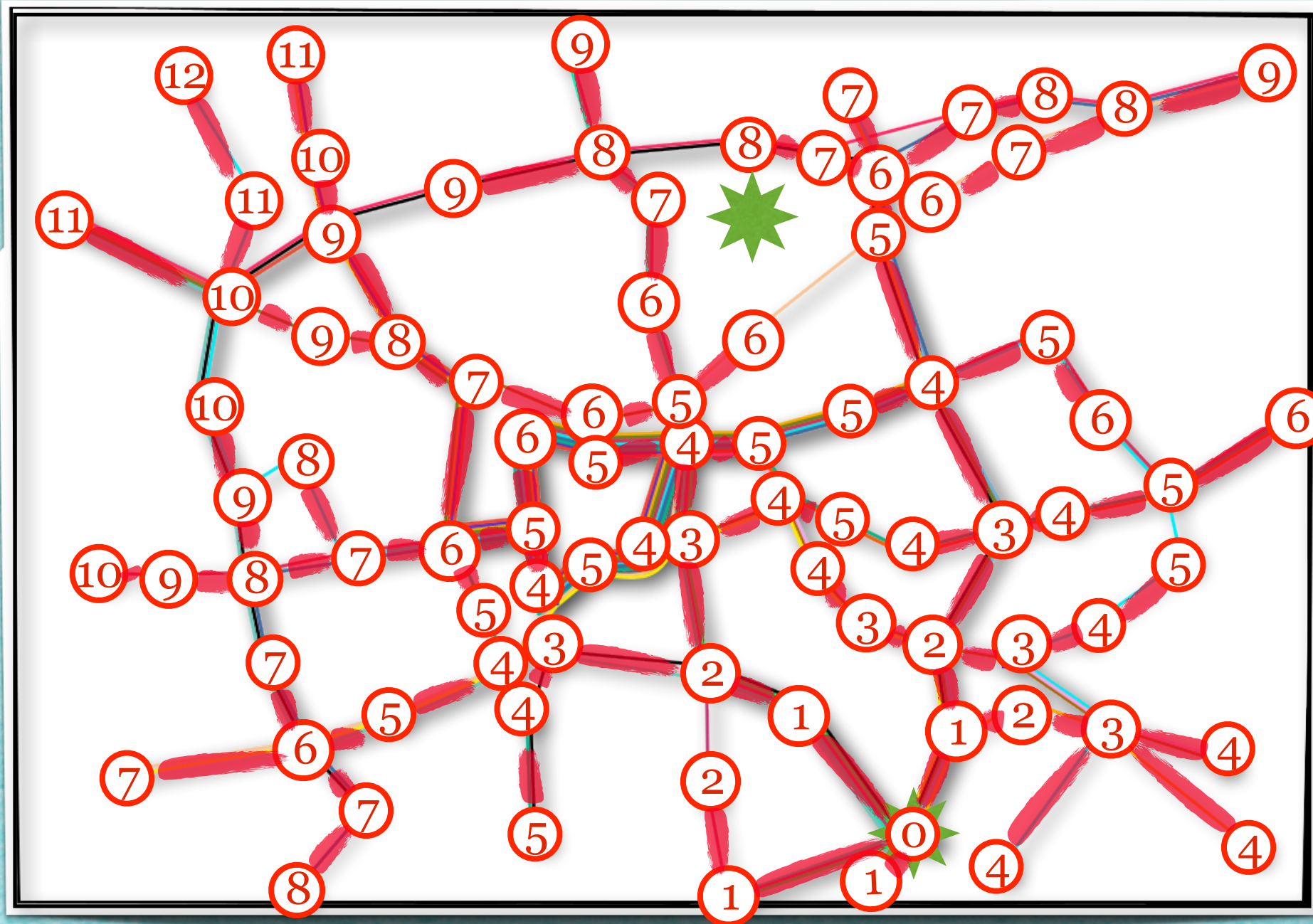
Wellenreiten in Graphen



Wellenreiten in Graphen

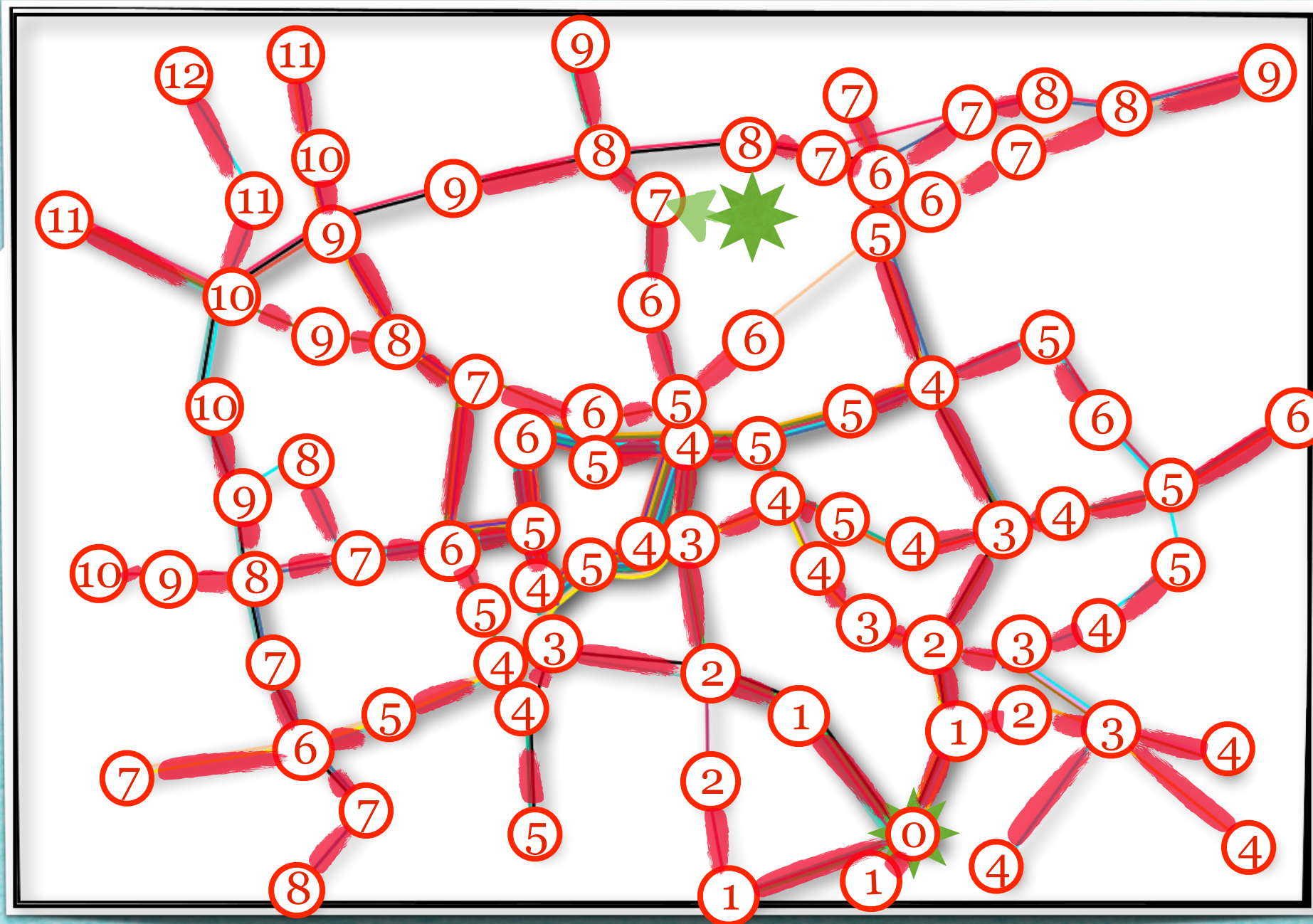


Wellenreiten in Graphen



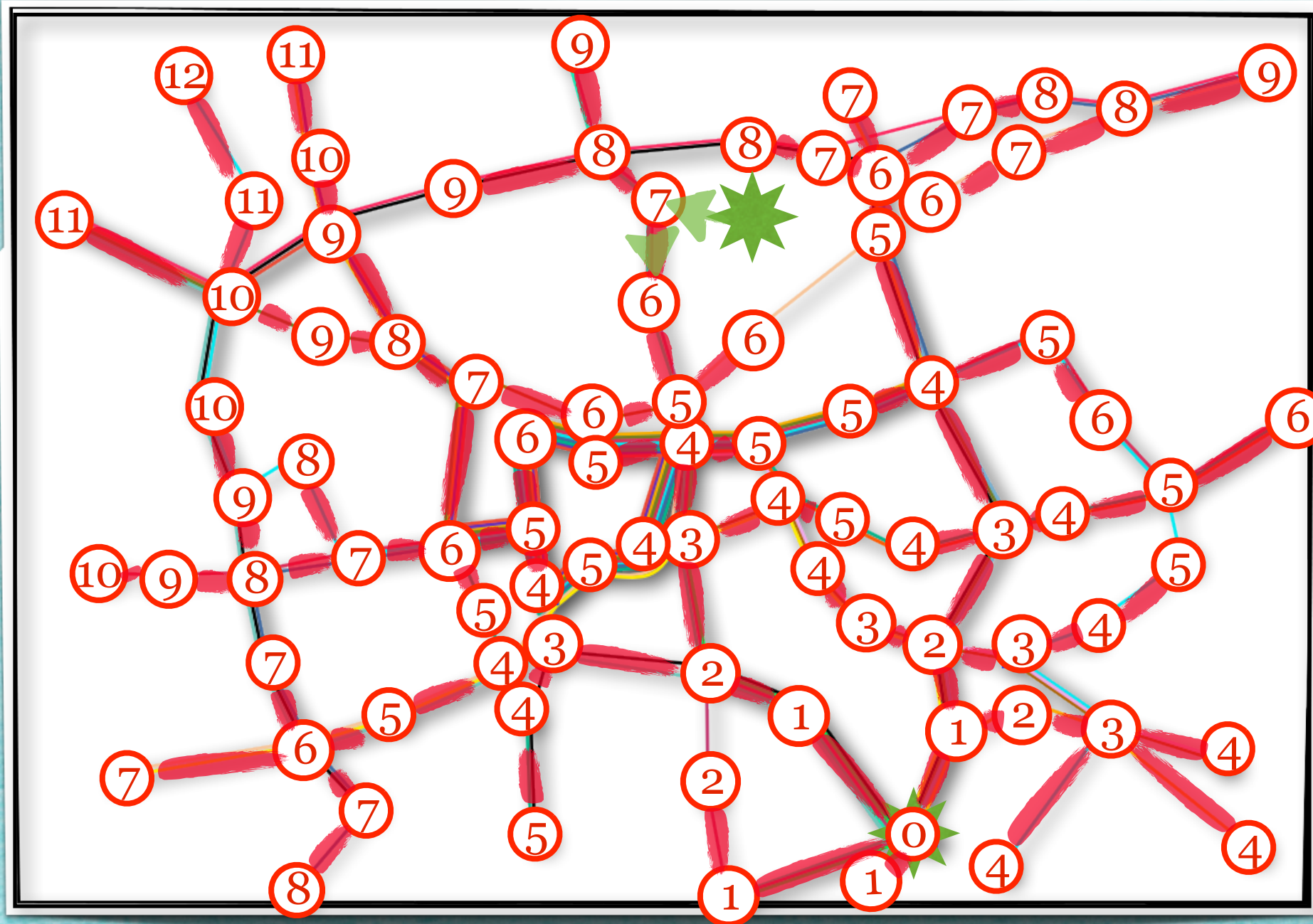
Breitensuche

Wellenreiten in Graphen



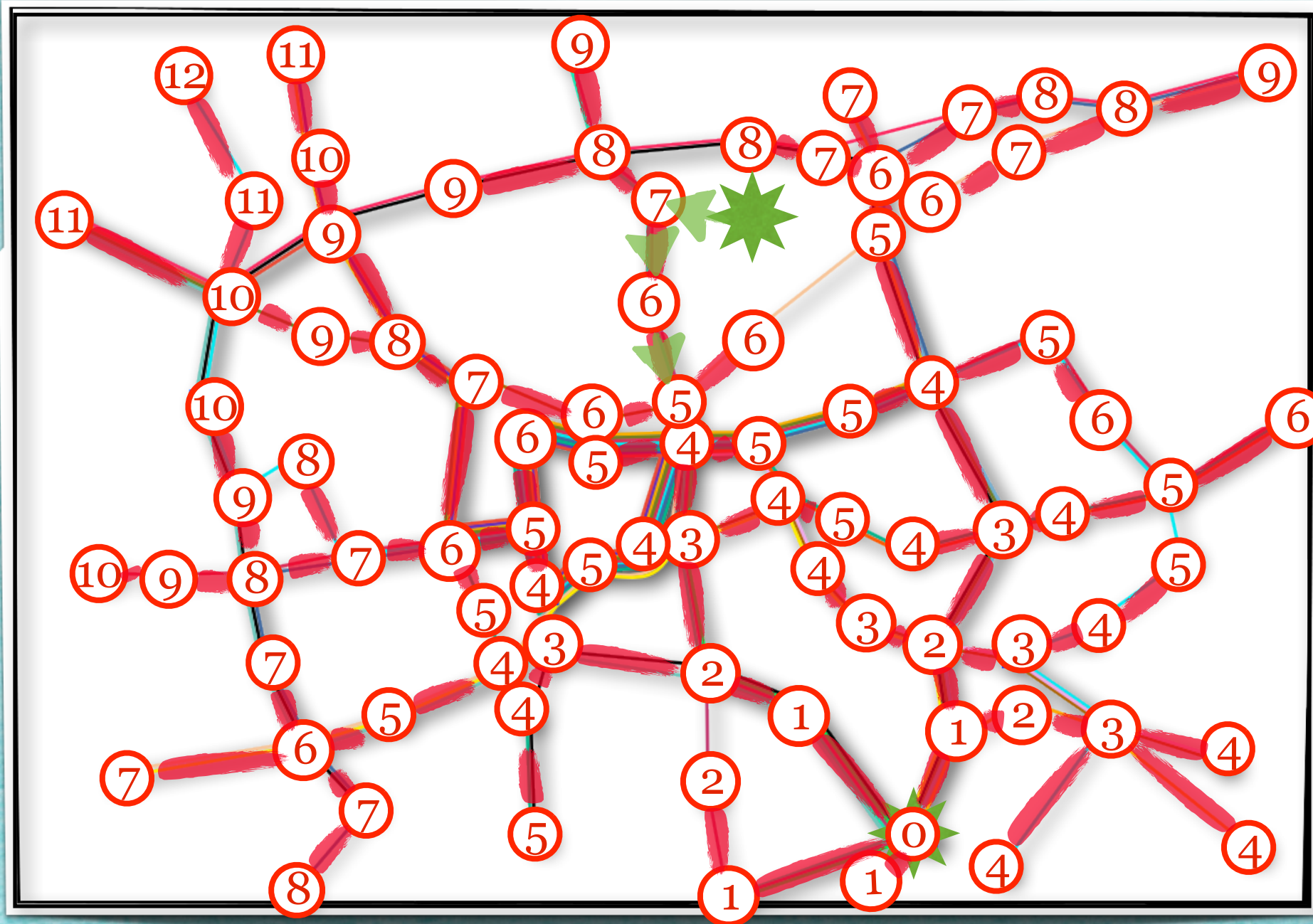
Breitensuche

Wellenreiten in Graphen



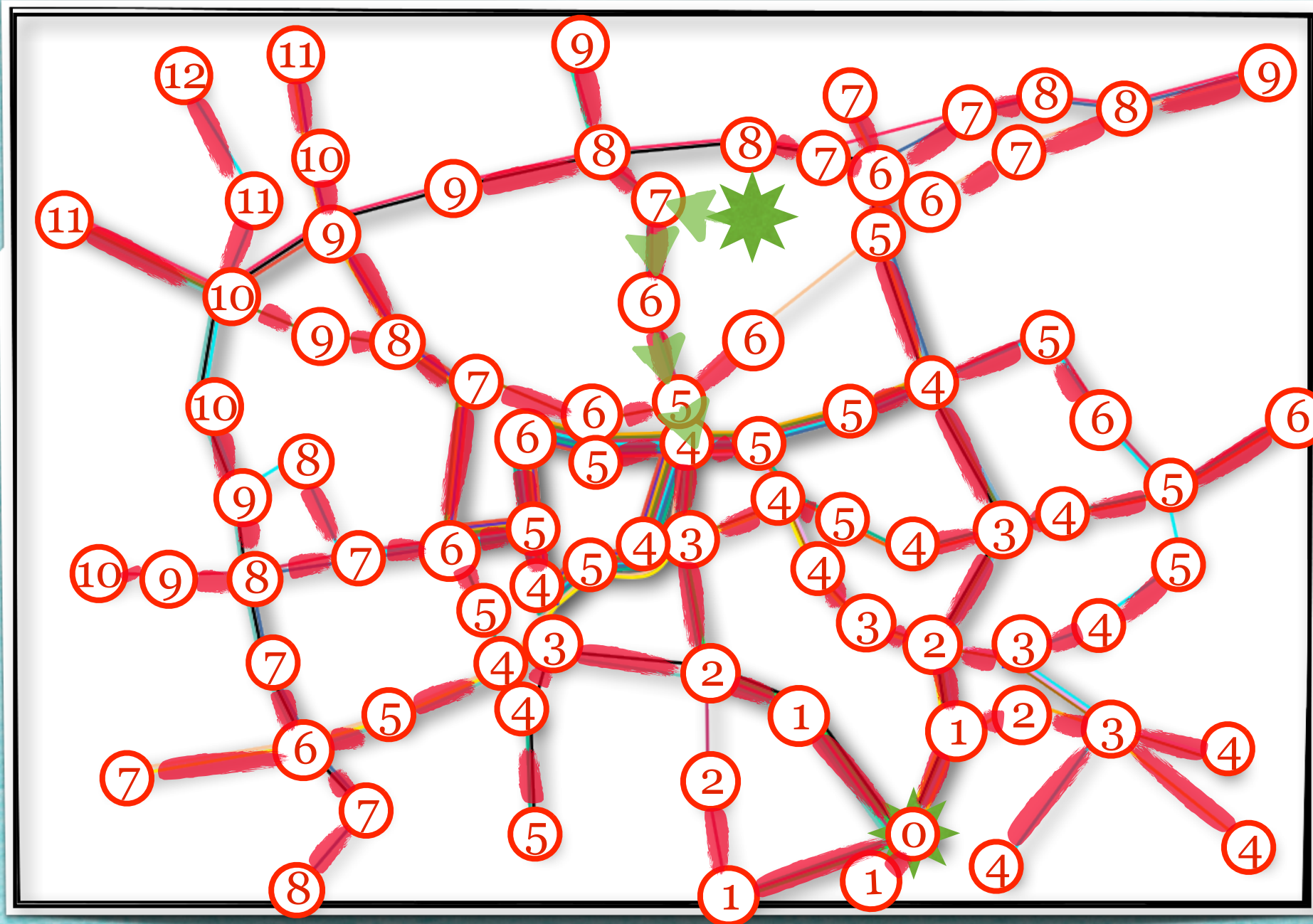
Breitensuche

Wellenreiten in Graphen



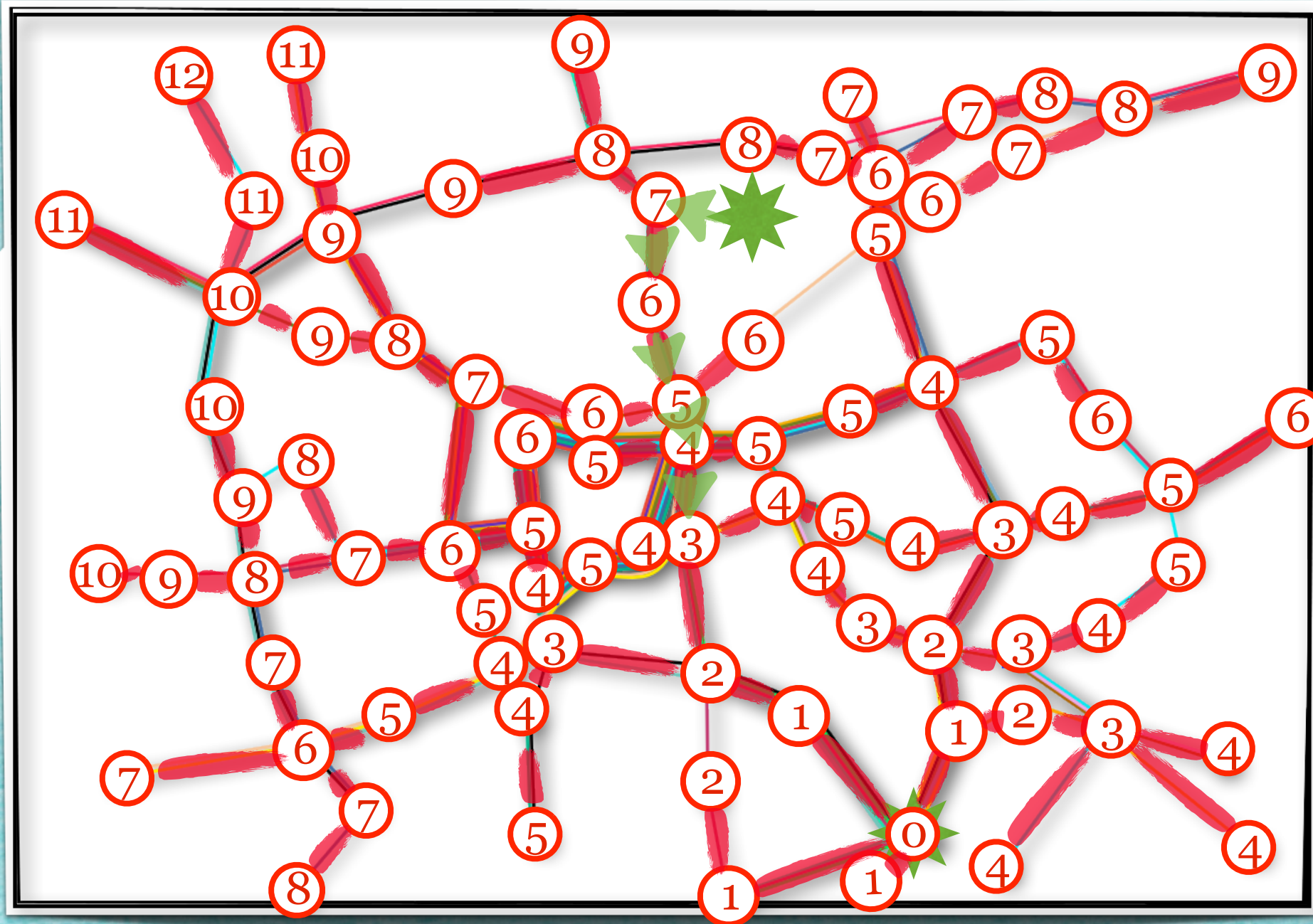
Breitensuche

Wellenreiten in Graphen



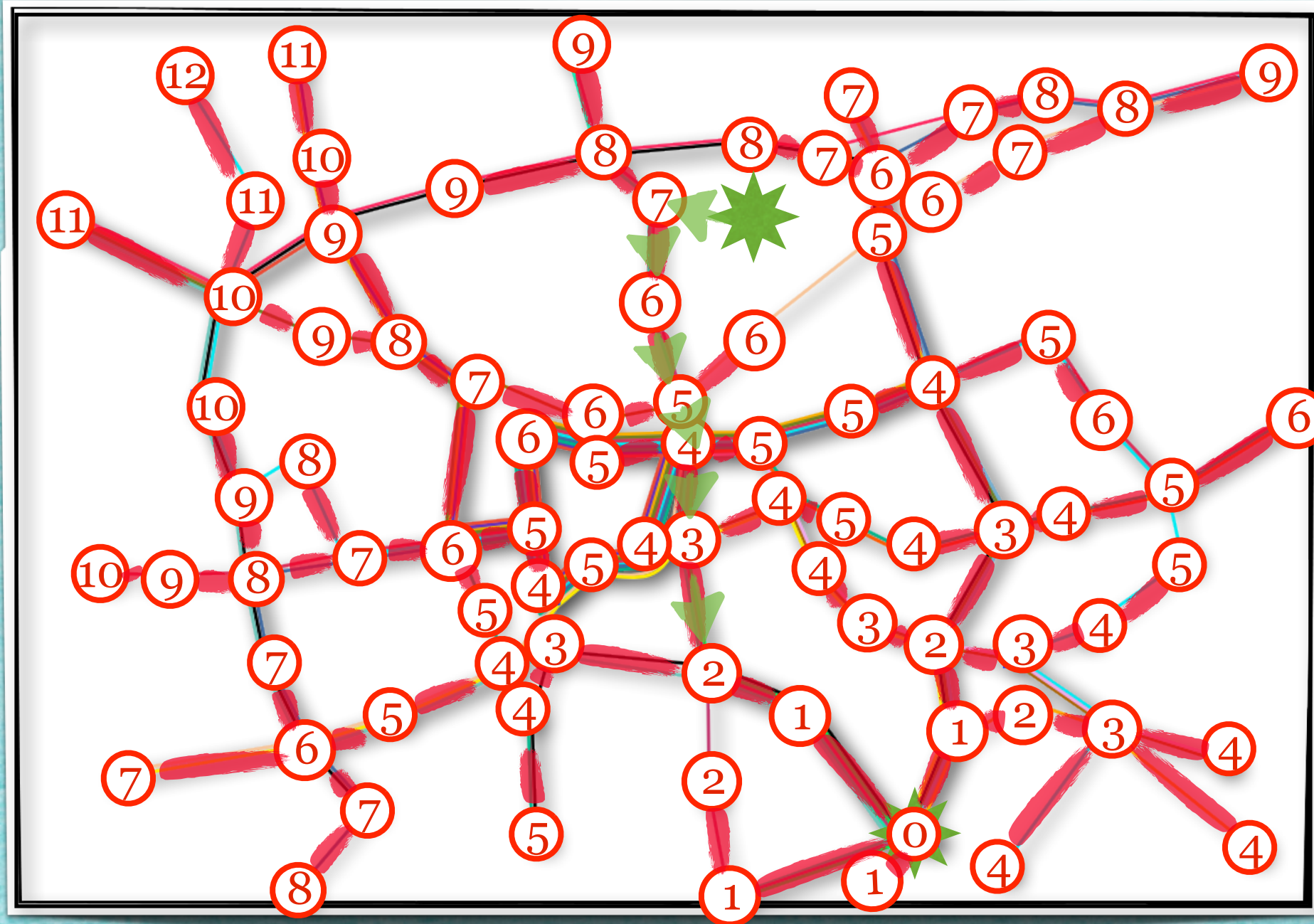
Breitensuche

Wellenreiten in Graphen



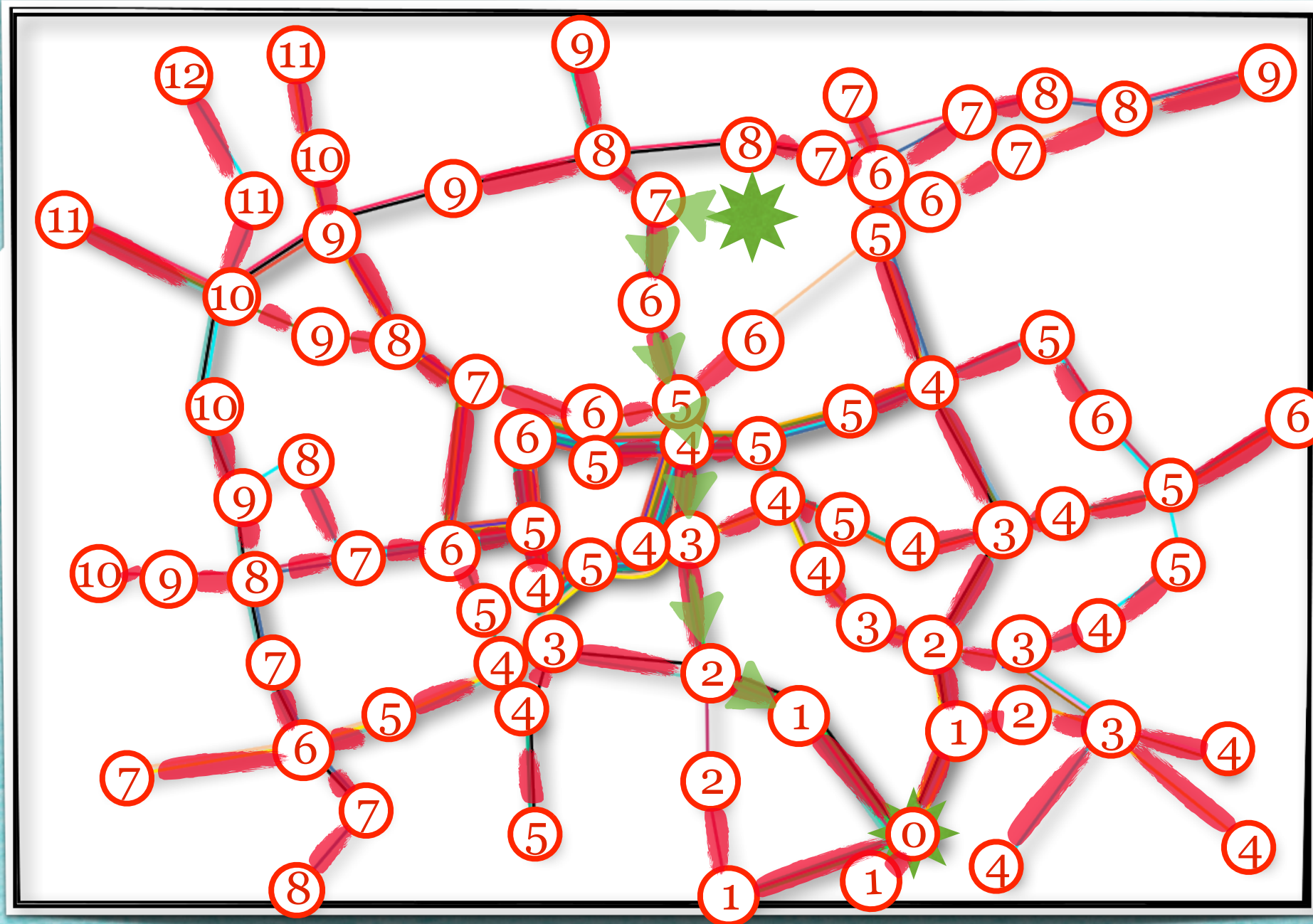
Breitensuche

Wellenreiten in Graphen



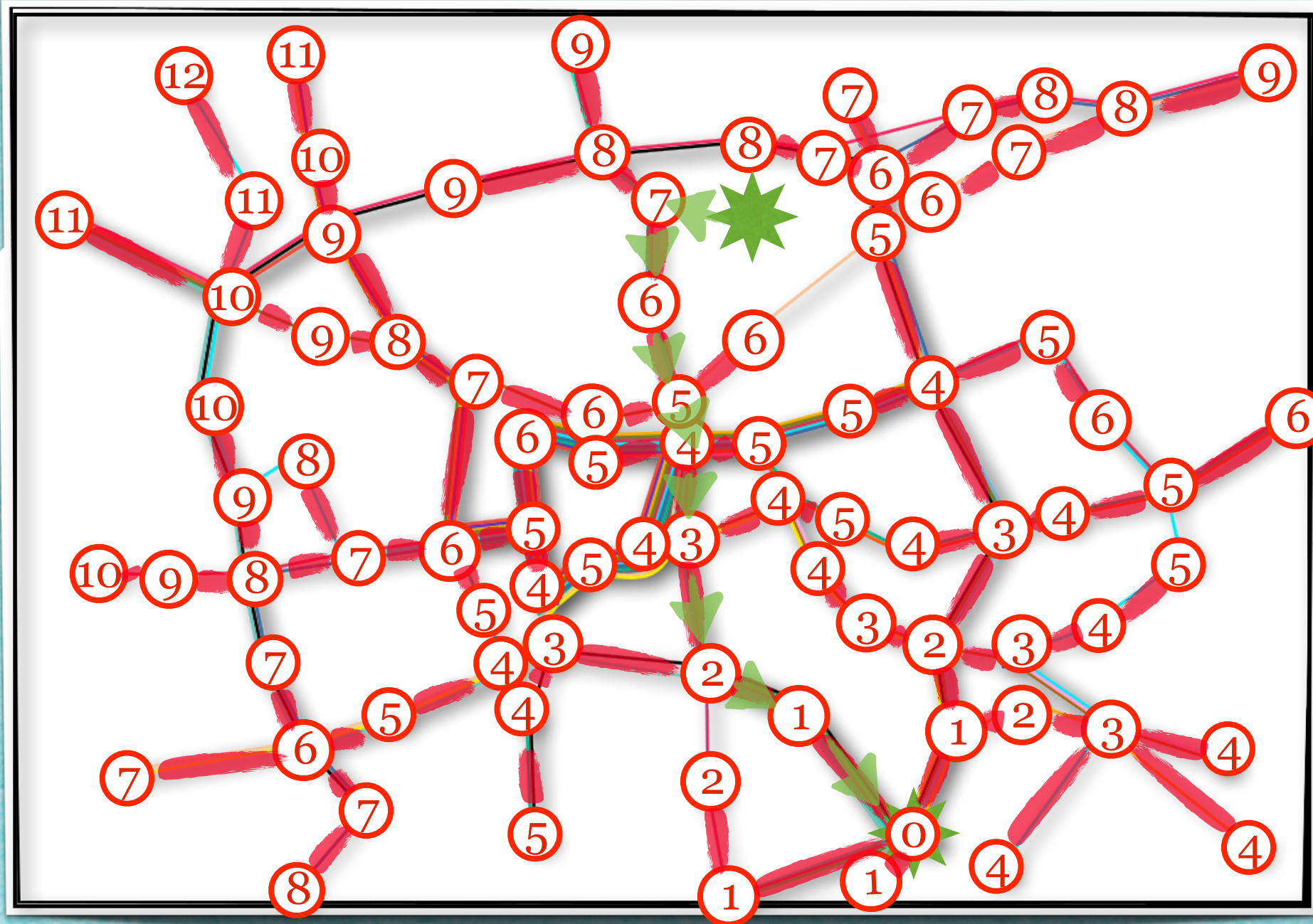
Breitensuche

Wellenreiten in Graphen



Breitensuche

Wellenreiten in Graphen



Breitensuche

Mehr demnächst!

s.fekete@tu-bs.de