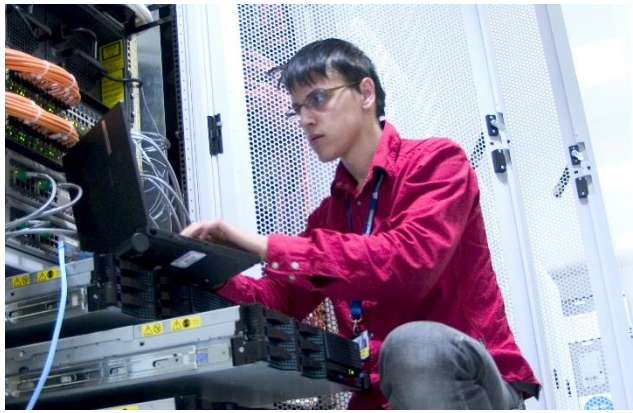
EMIL-VON-BEHRING GYMNASIUM

Graphendurchlauf – Tiefensuche

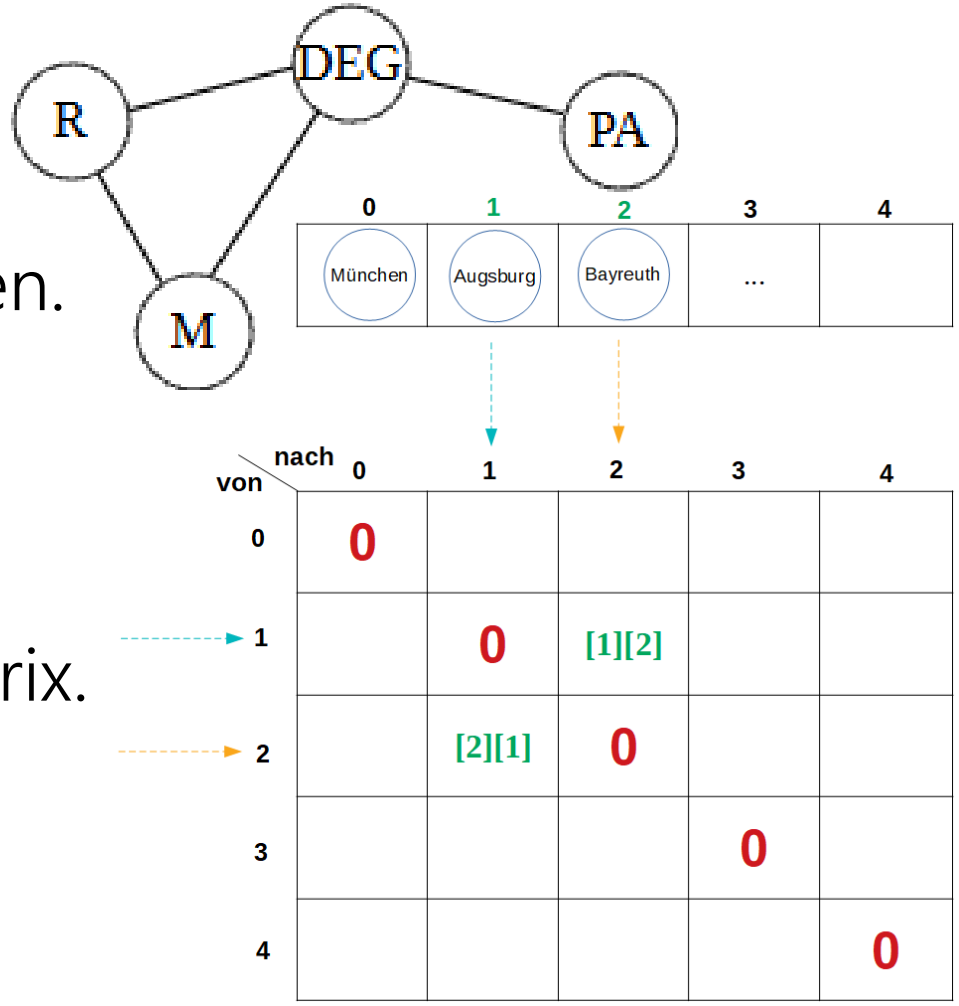
Der Weg ist das Ziel

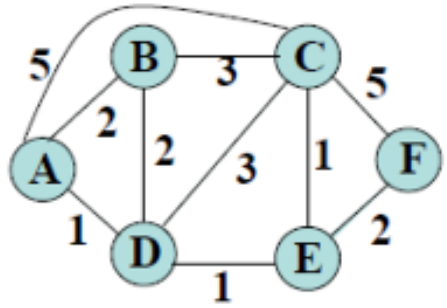


Grafiken © shutterstock.com

Wiederholung Datenstruktur Graph

- Ein Graph besteht aus Knoten und Kanten. Die Knoten haben einen Inhalt z. B. eine Stadt.
- Die Knoten werden in einem Array gespeichert und die Kanten in einer Matrix.
- Ein Durchlauf soll einen Graphen systematisch durchlaufen und sich dabei nicht in den Zyklen „verheddern“.

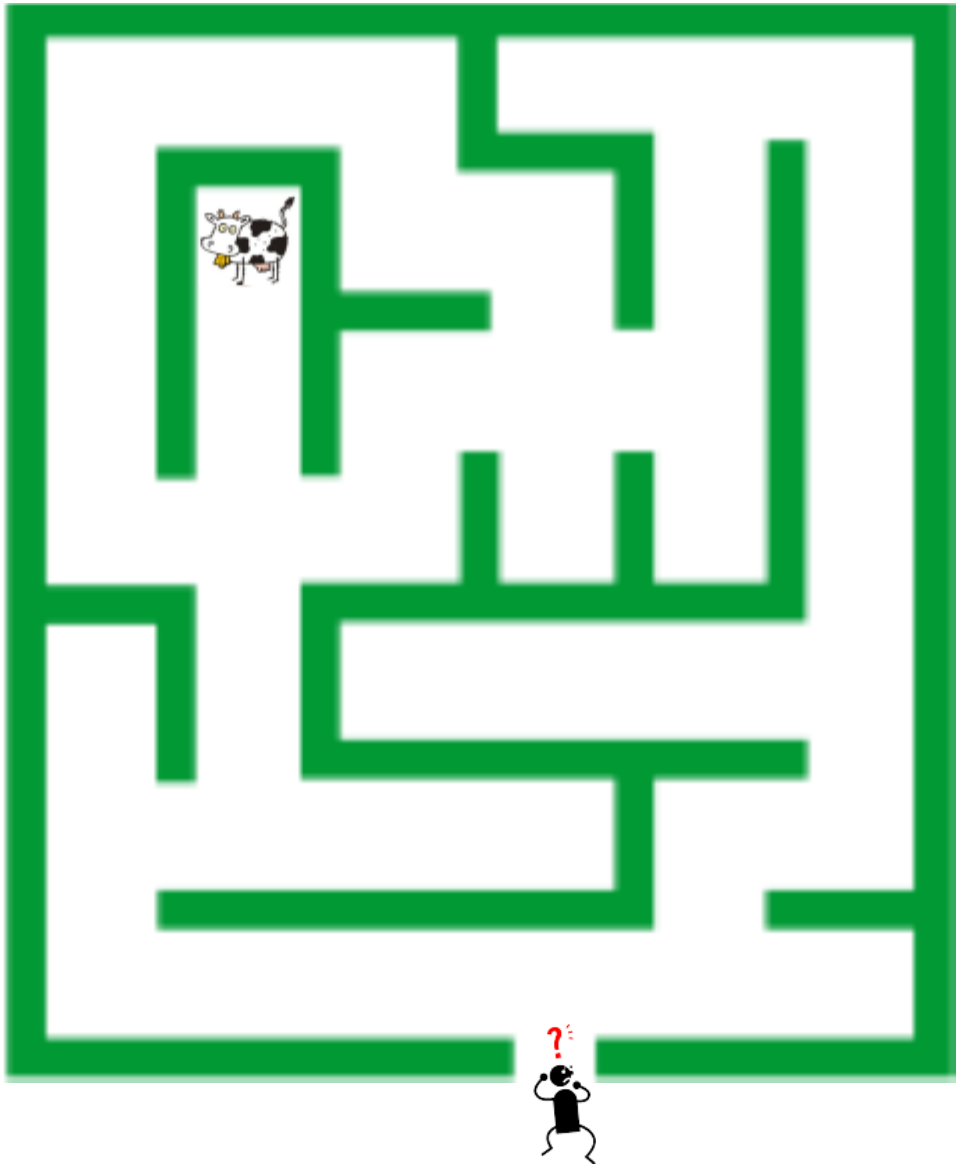




Theseus und der Minotaurus

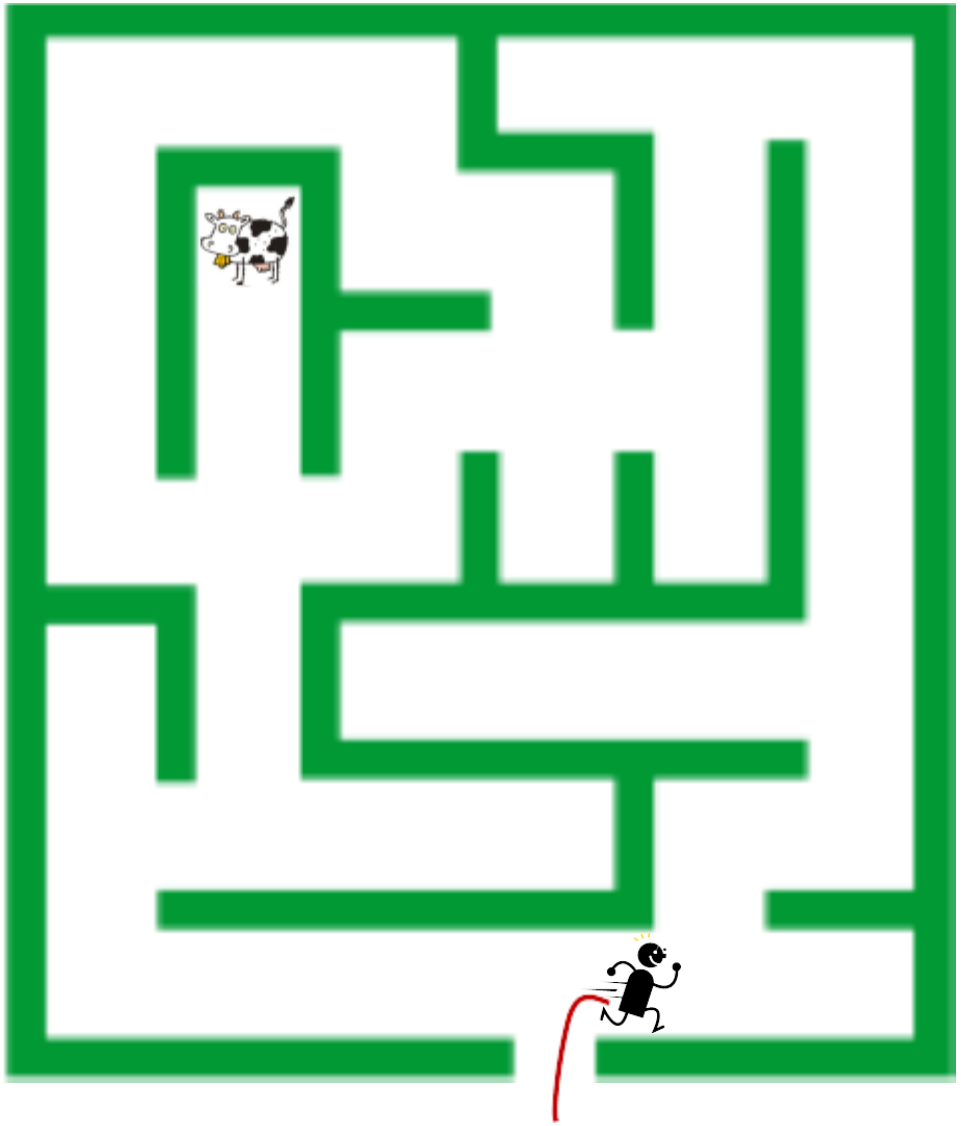
- Der griechischen Mythologie zufolge fand Theseus mit Hilfe des Ariadnefadens den Weg durch das Labyrinth, in dem sich der Minotaurus befand.

Labyrinth des Minotaurus

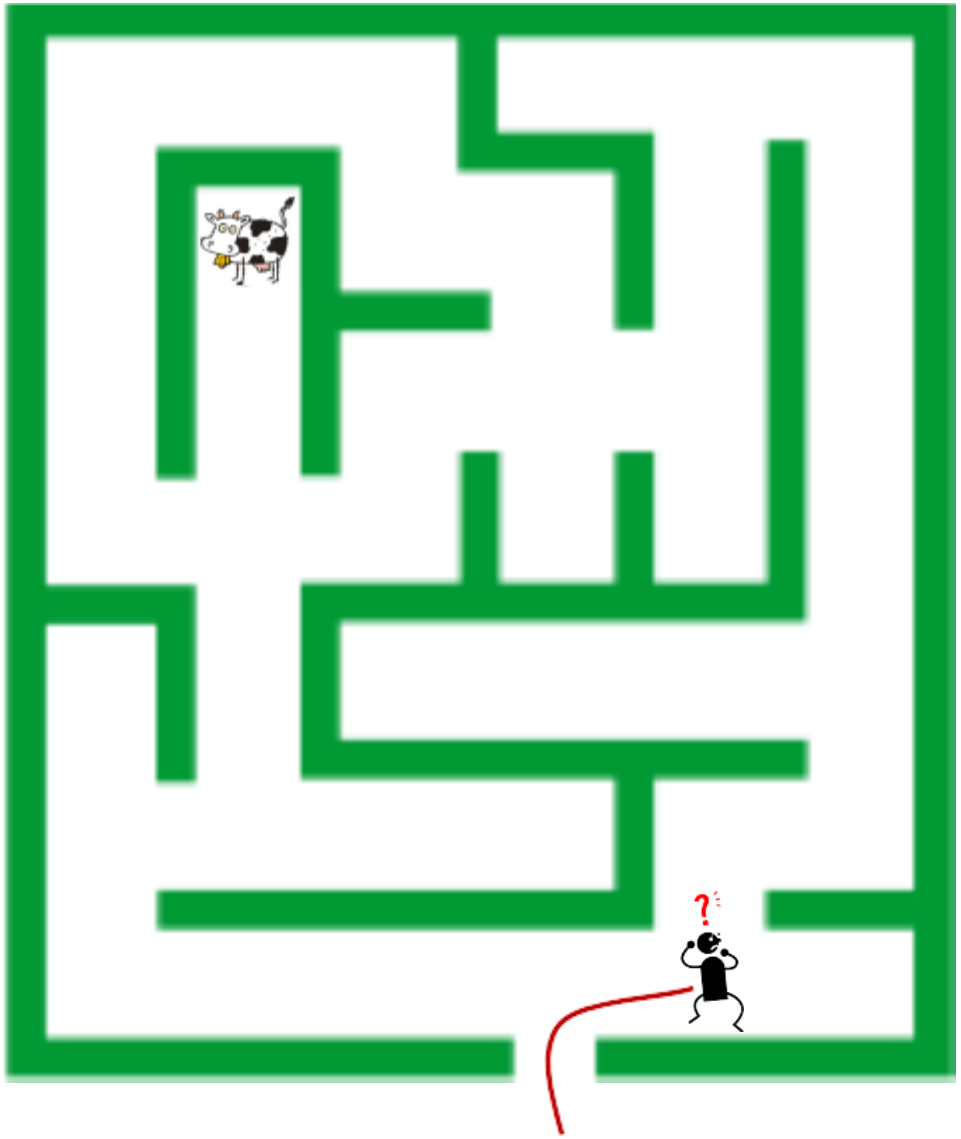


Labyrinth des Minotaurus

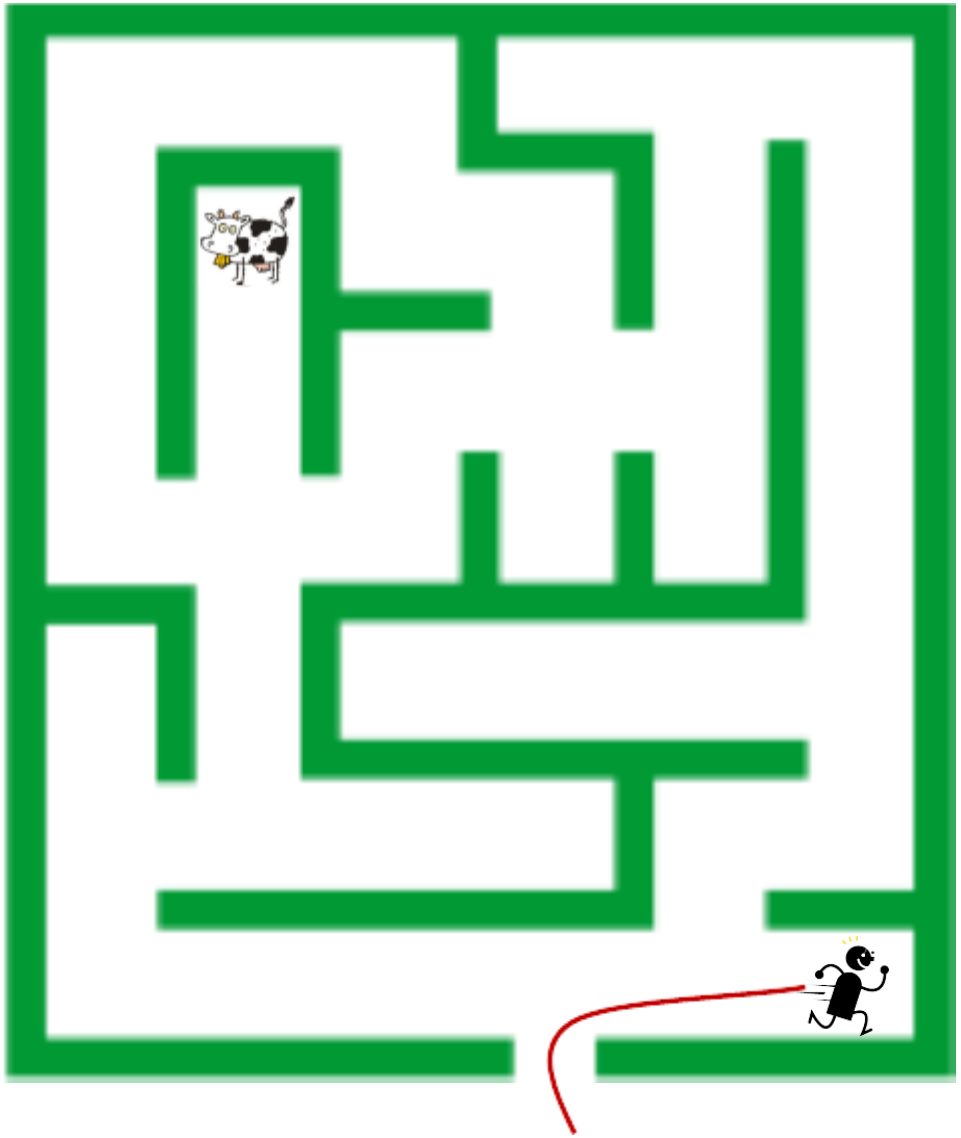
Labyrinth des Minotaurus



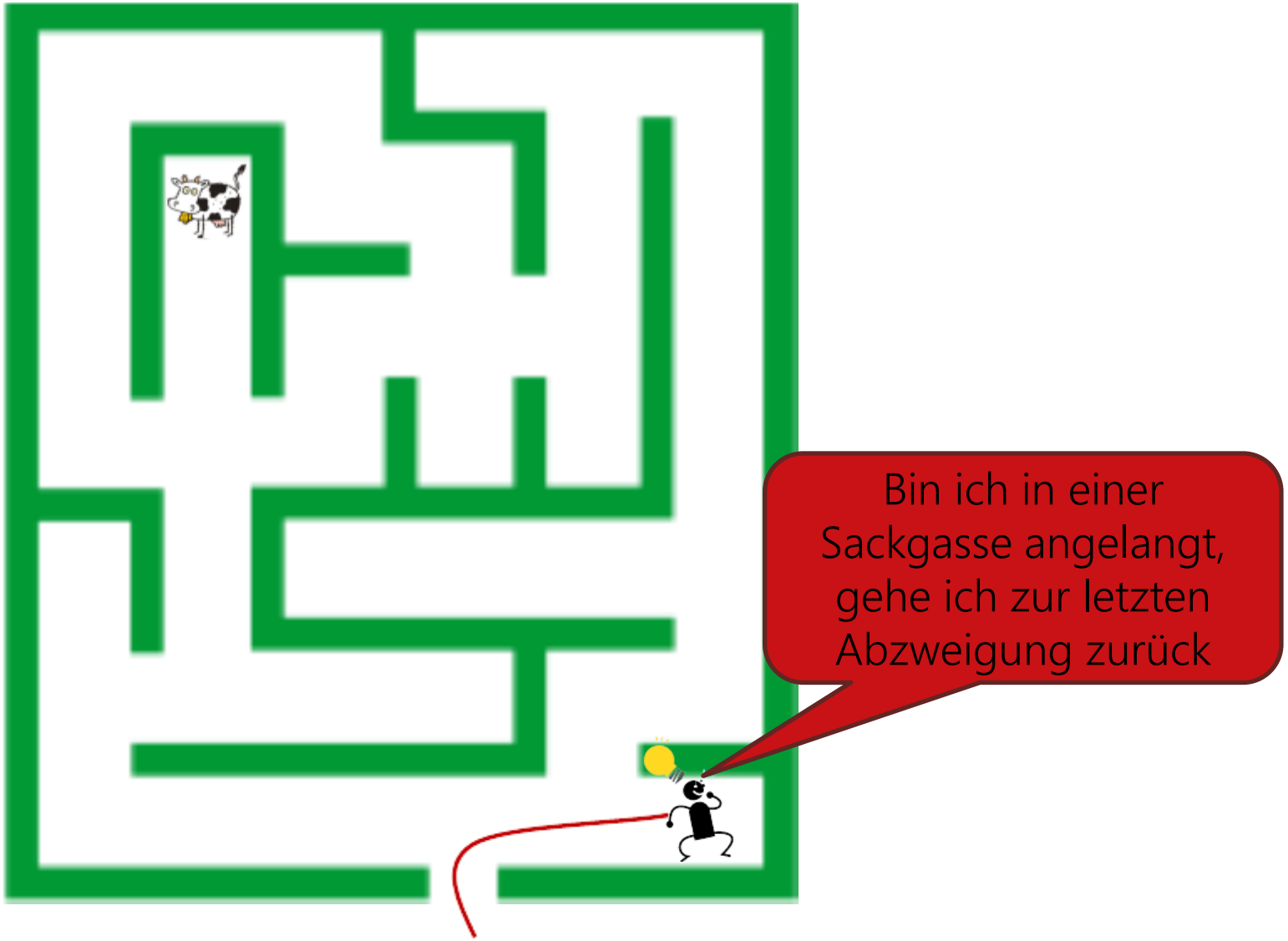
Labyrinth des Minotaurus



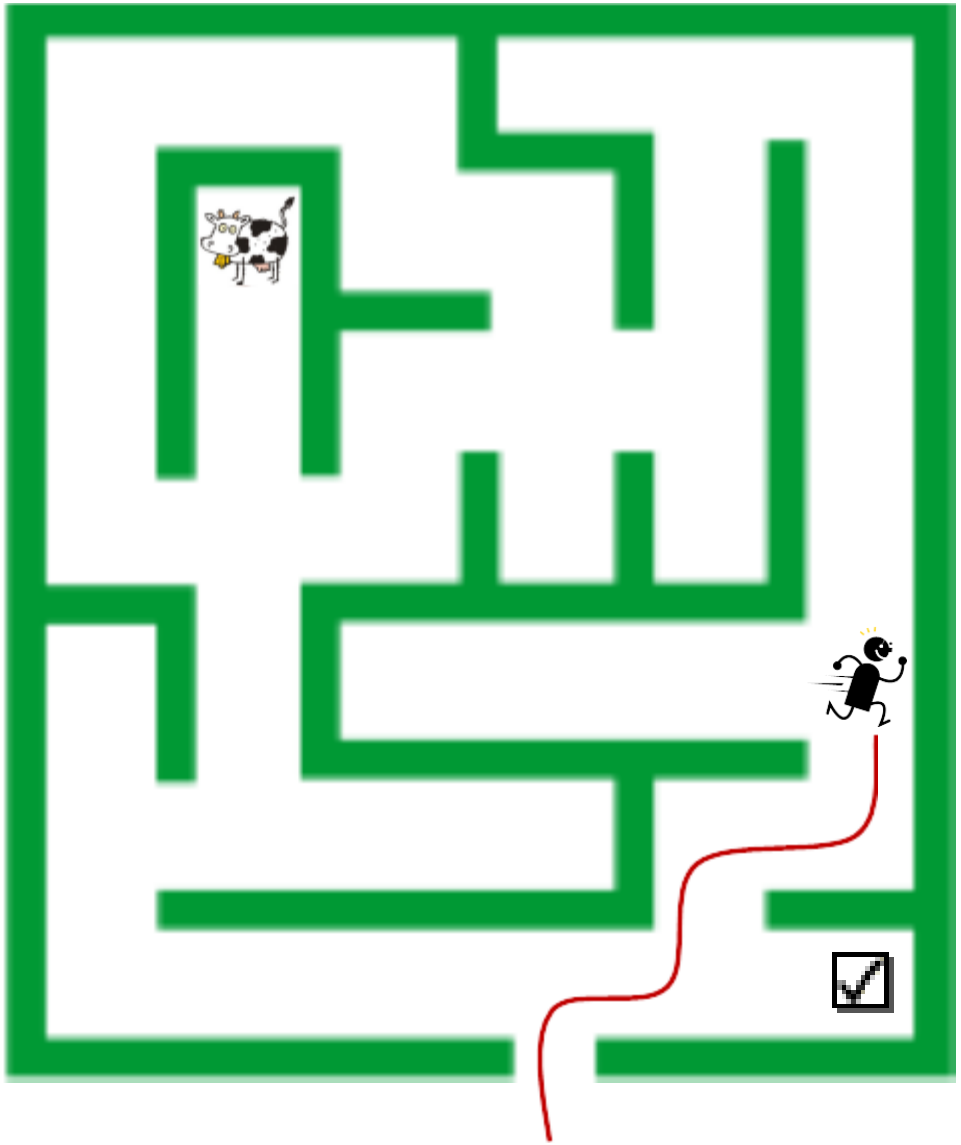
Labyrinth des Minotaurus



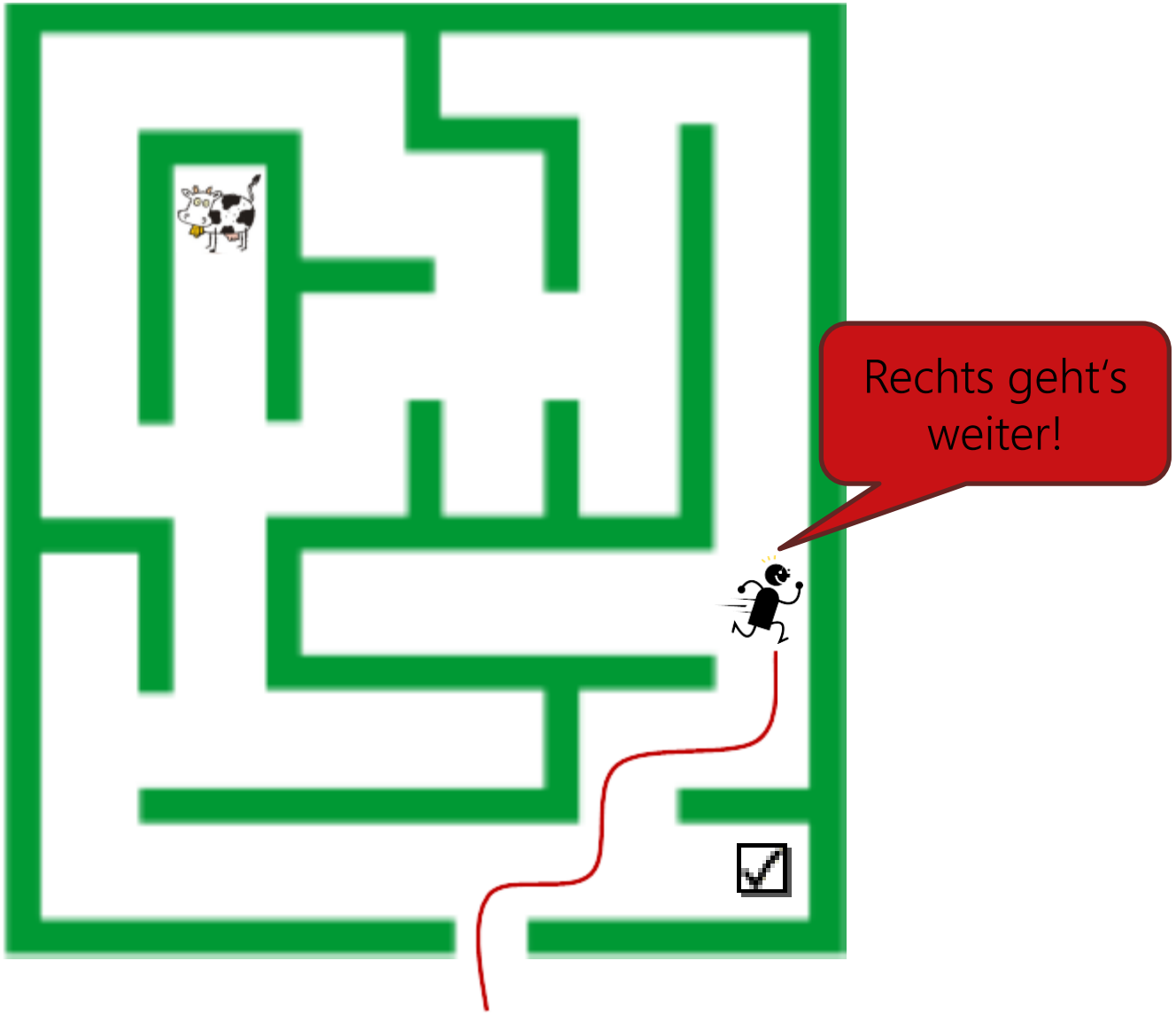
Labyrinth des Minotaurus



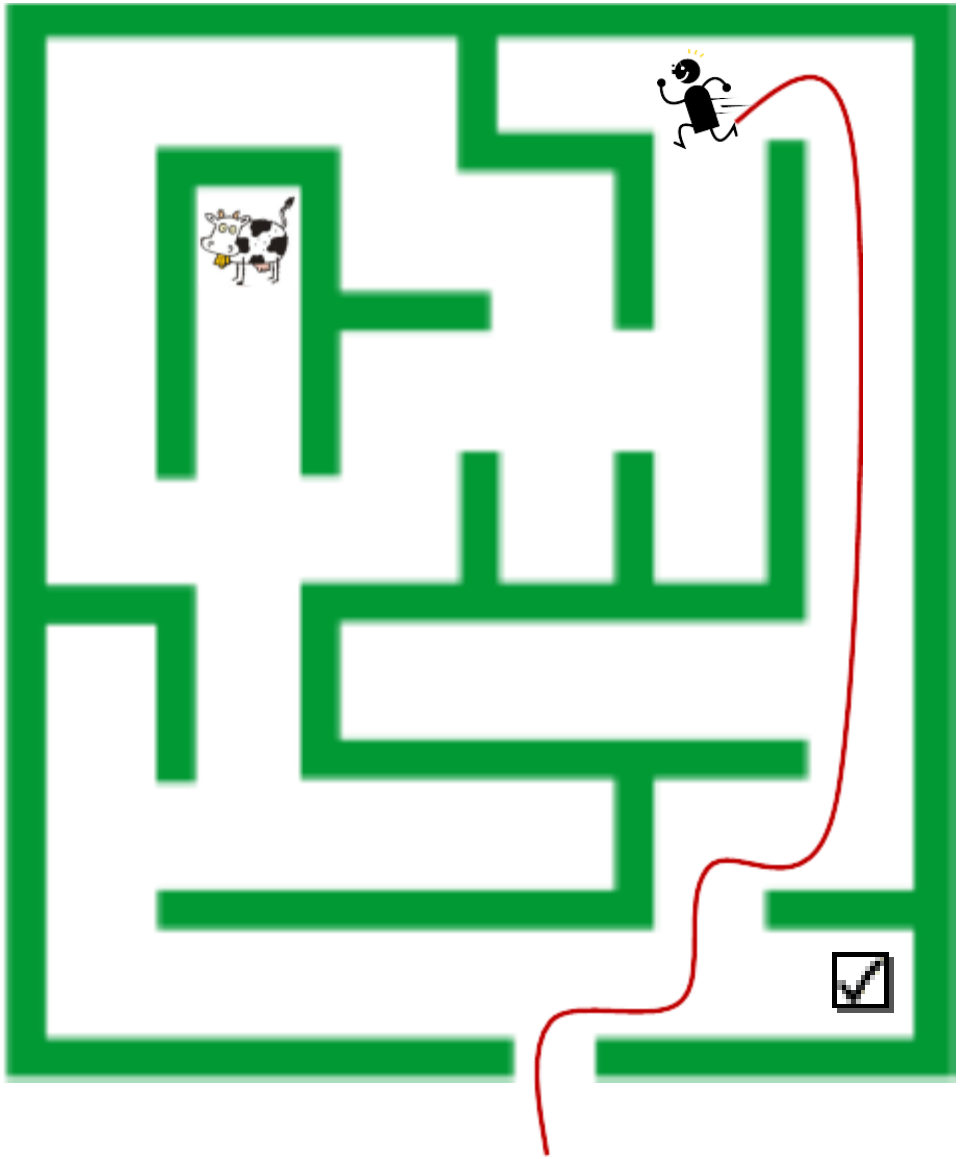
Labyrinth des Minotaurus



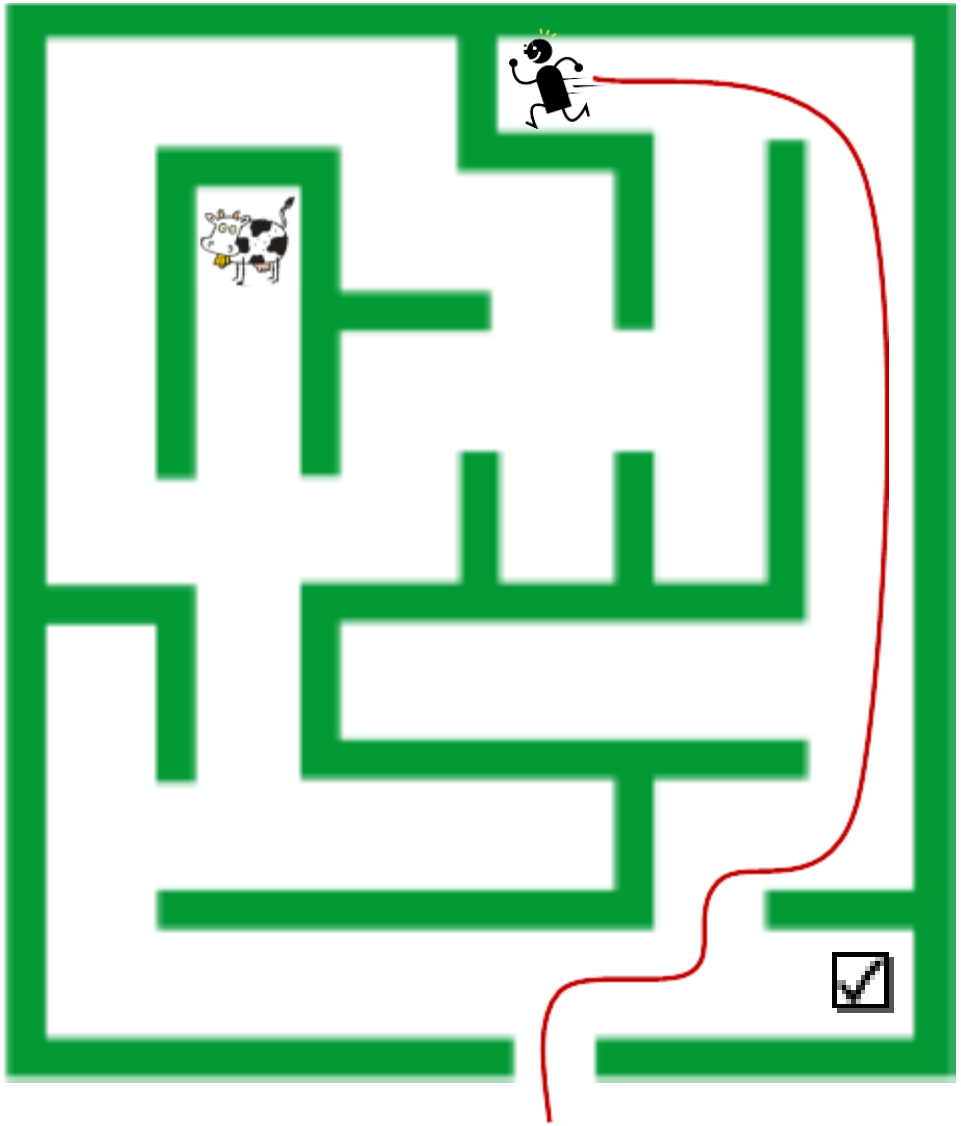
Labyrinth des Minotaurus



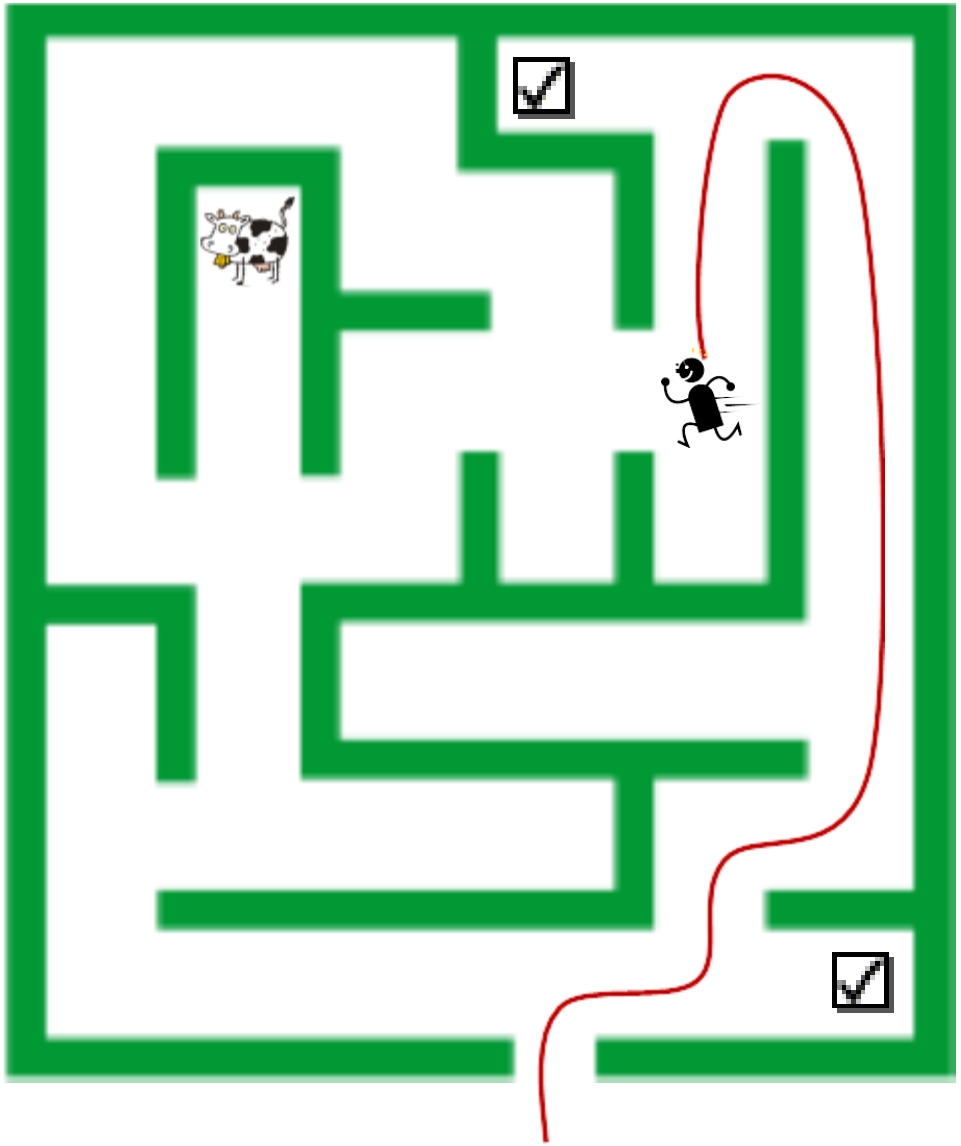
Labyrinth des Minotaurus



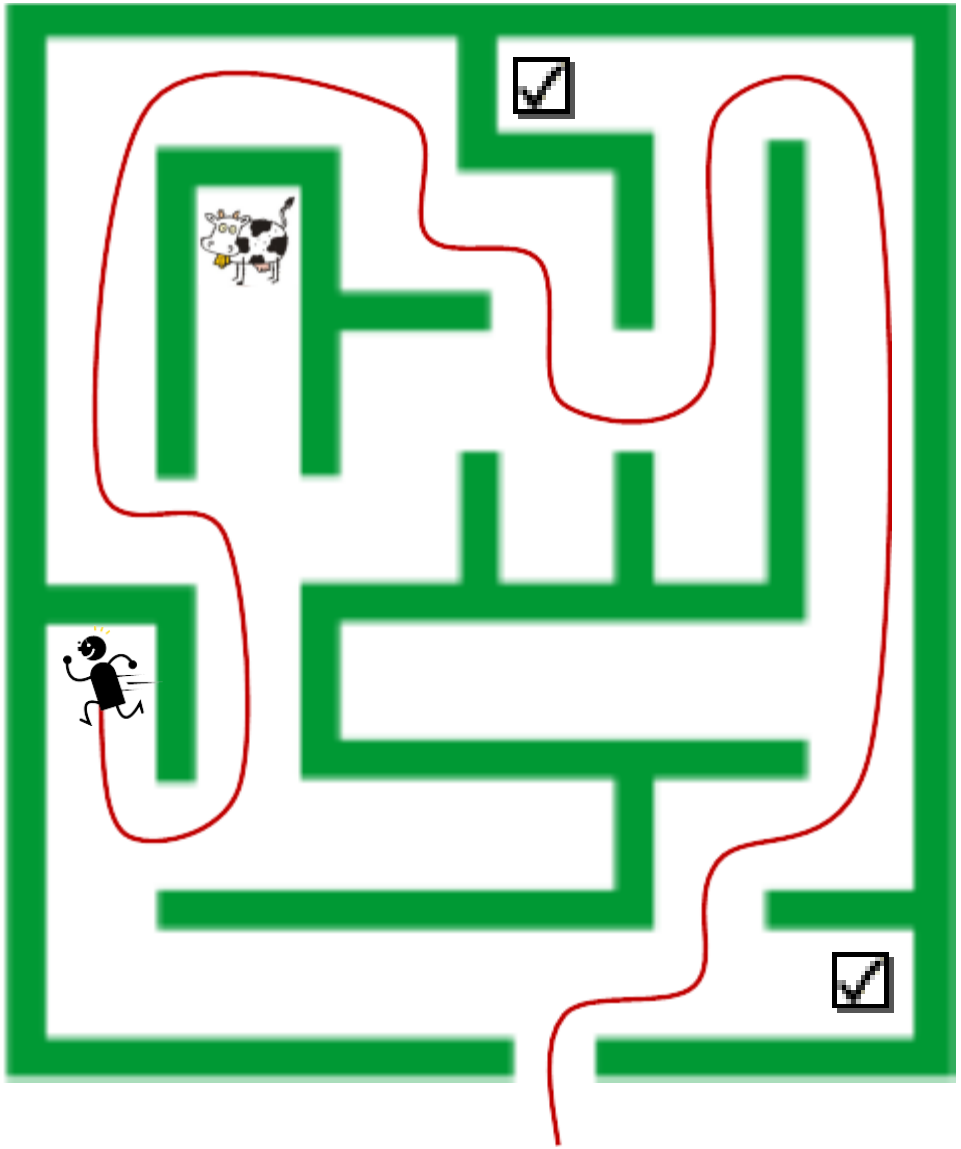
Labyrinth des Minotaurus



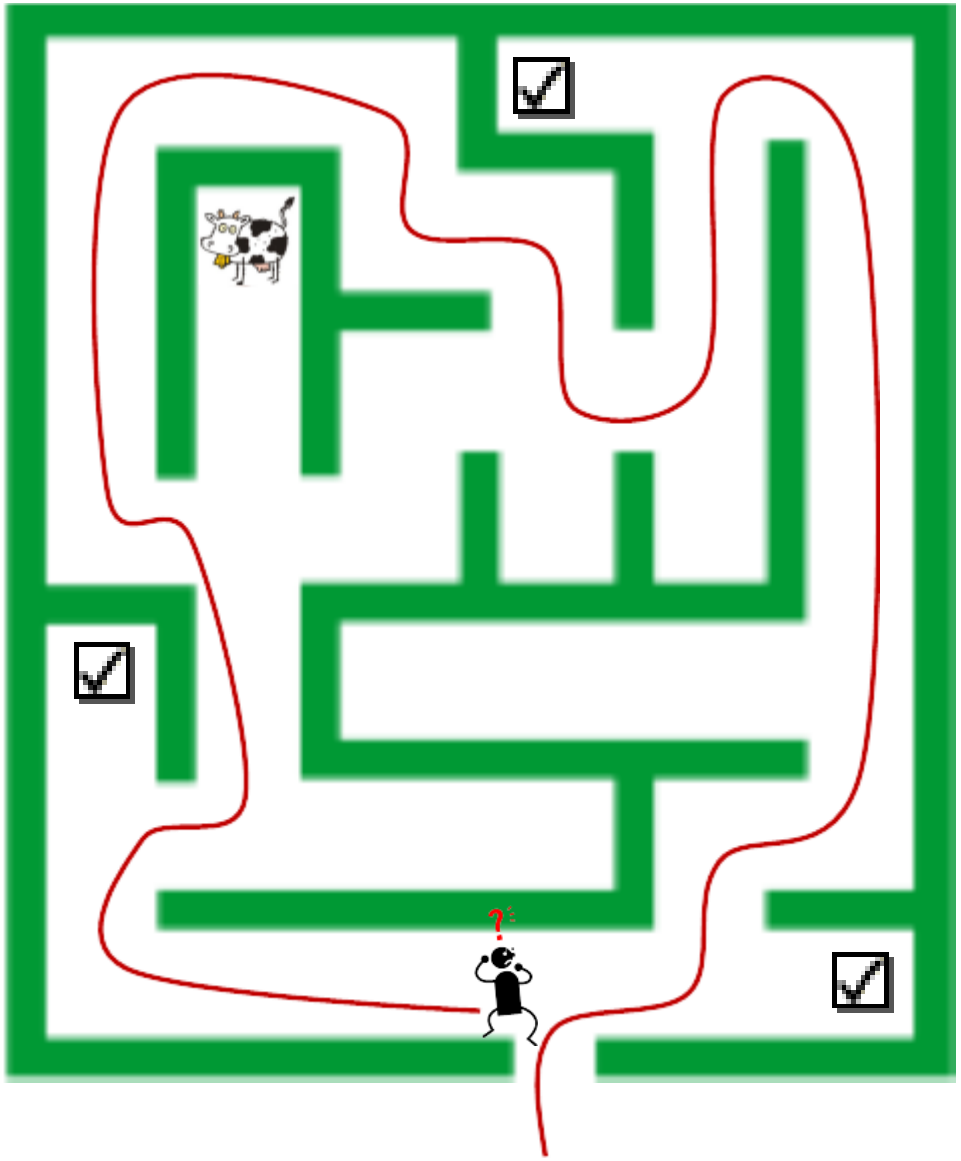
Labyrinth des Minotaurus



Labyrinth des Minotaurus

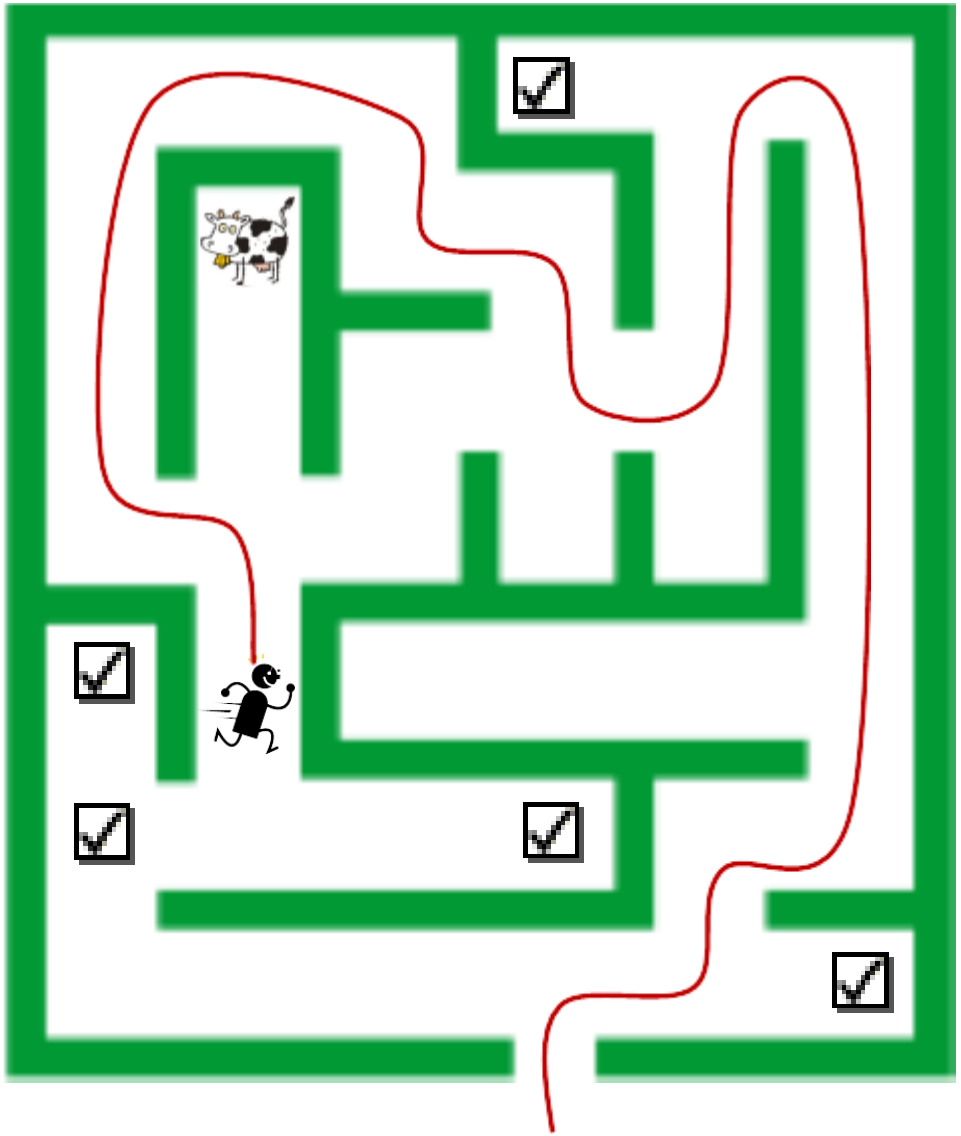


Labyrinth des Minotaurus

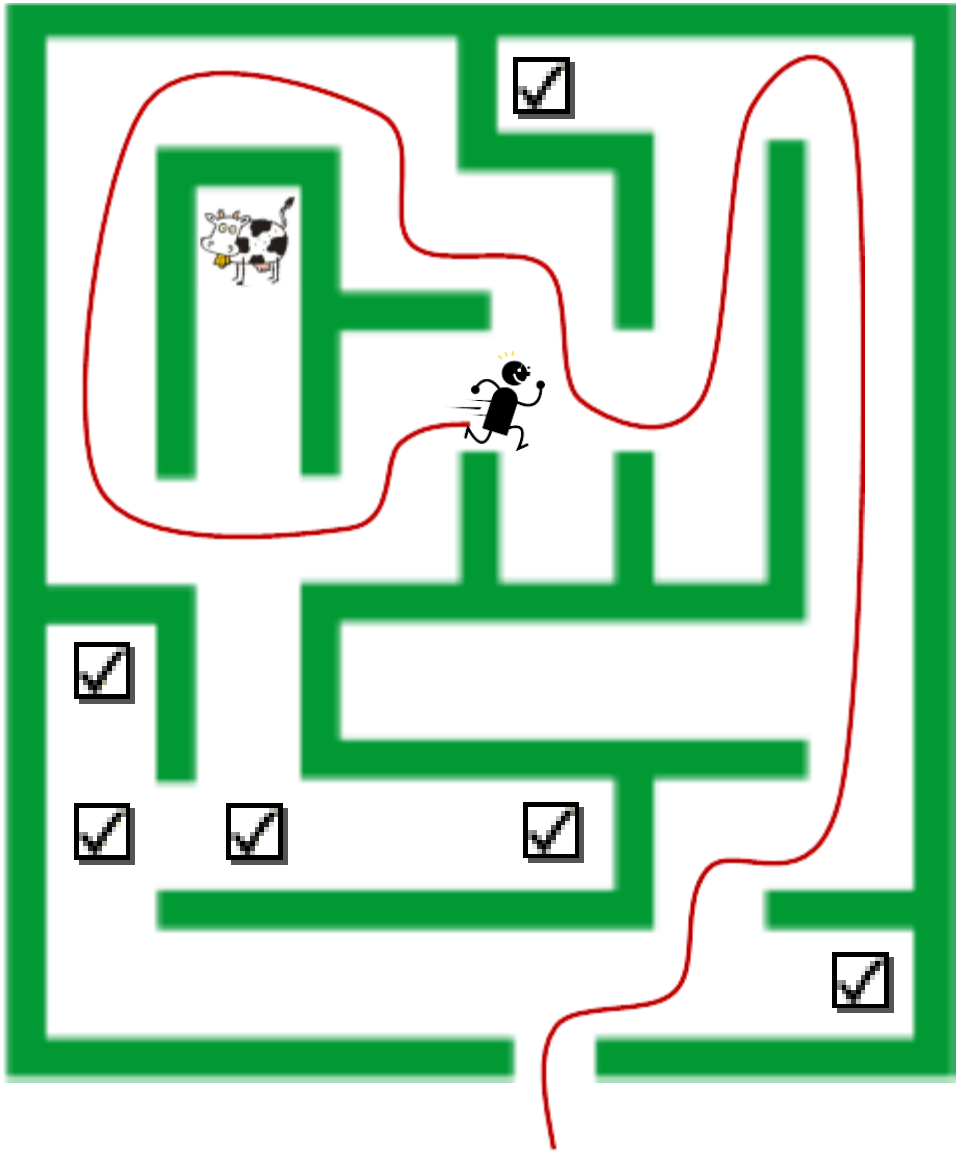


Labyrinth des Minotaurus

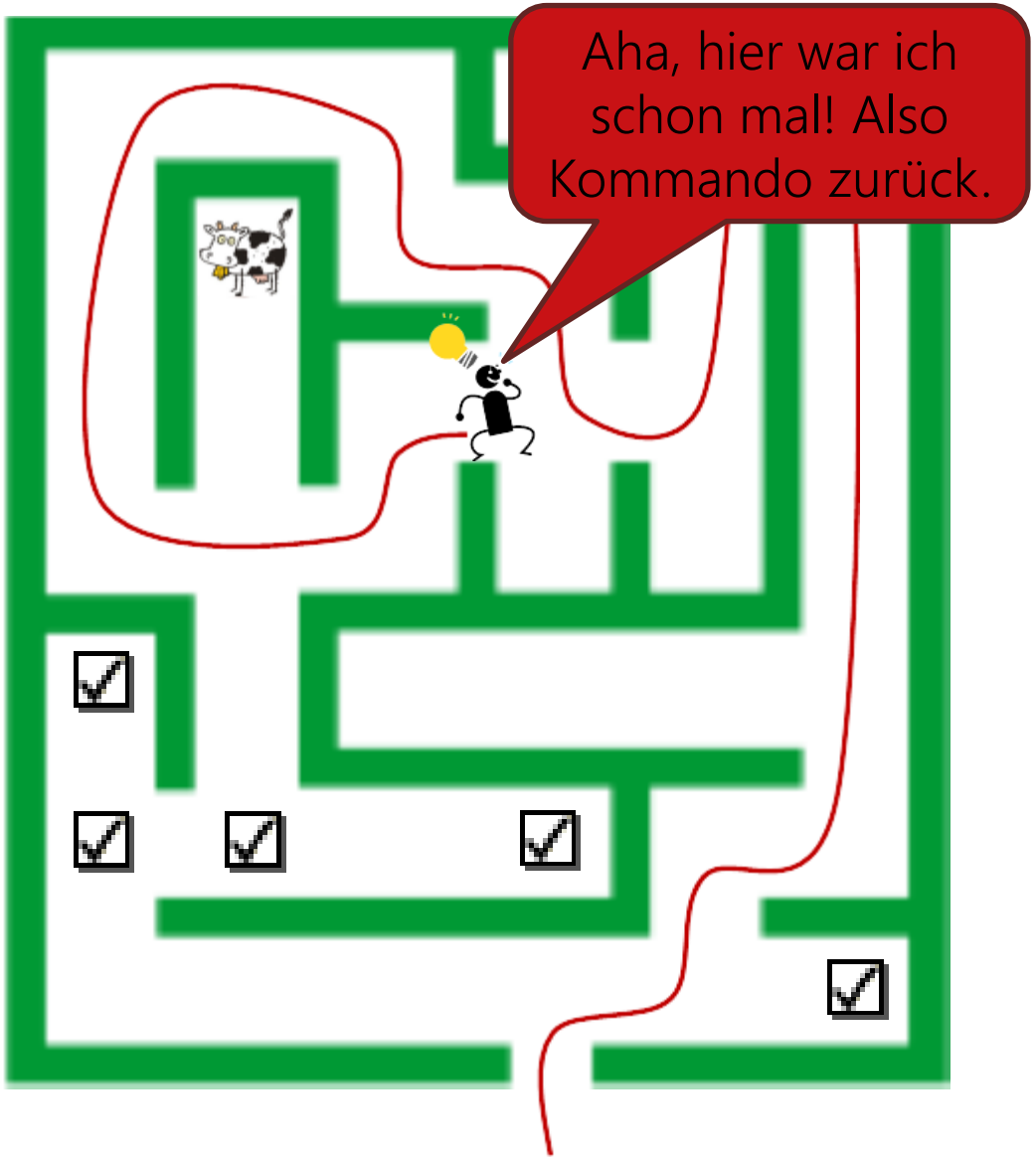
Labyrinth des Minotaurus



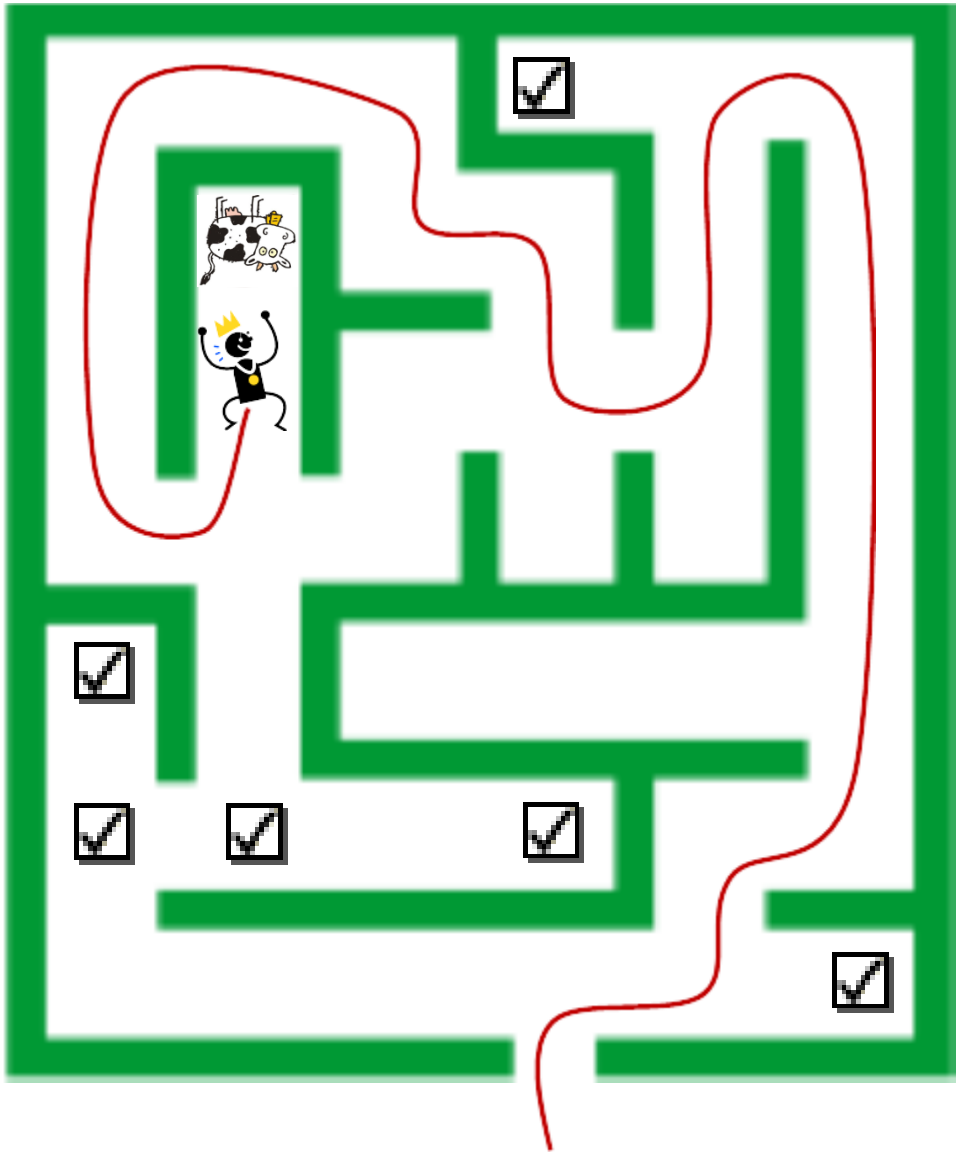
Labyrinth des Minotaurus



Labyrinth des Minotaurus



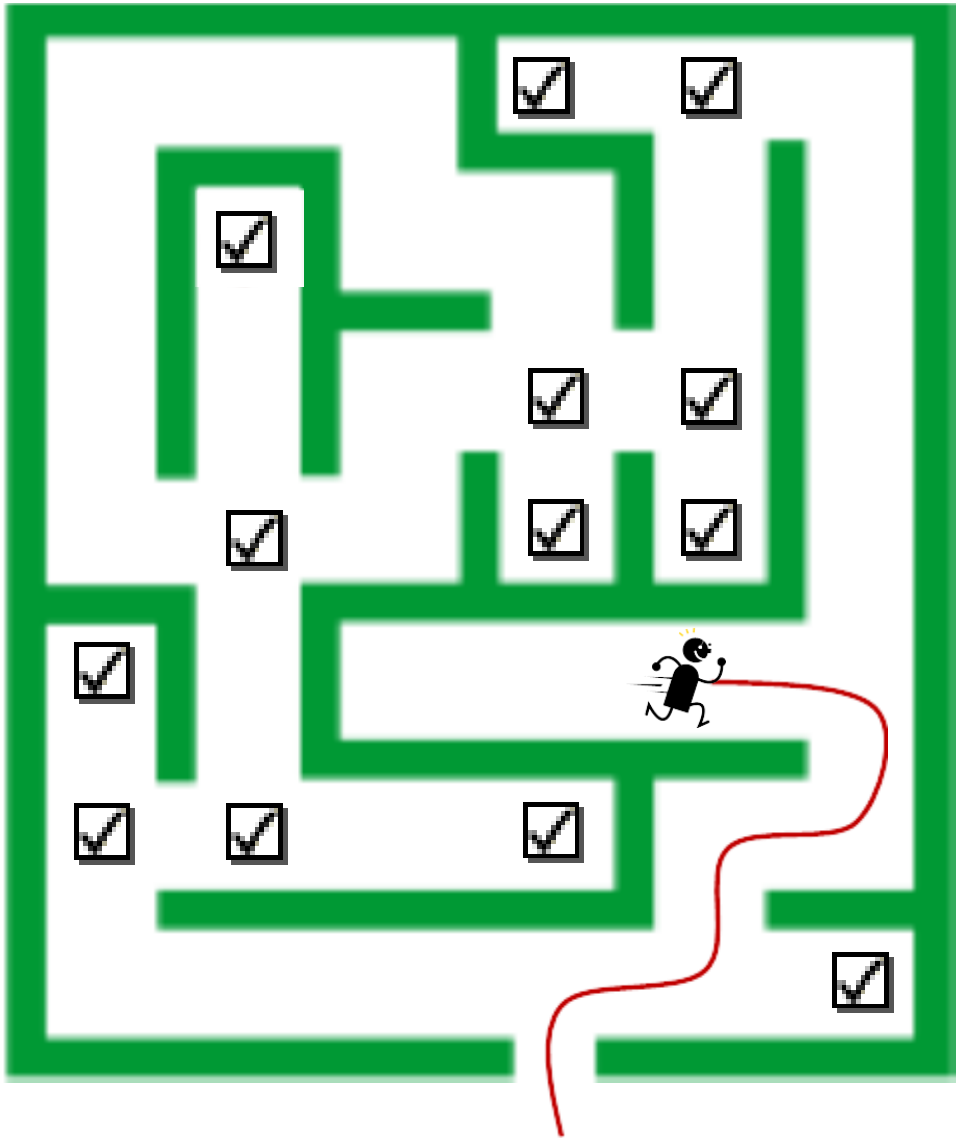
Labyrinth des Minotaurus



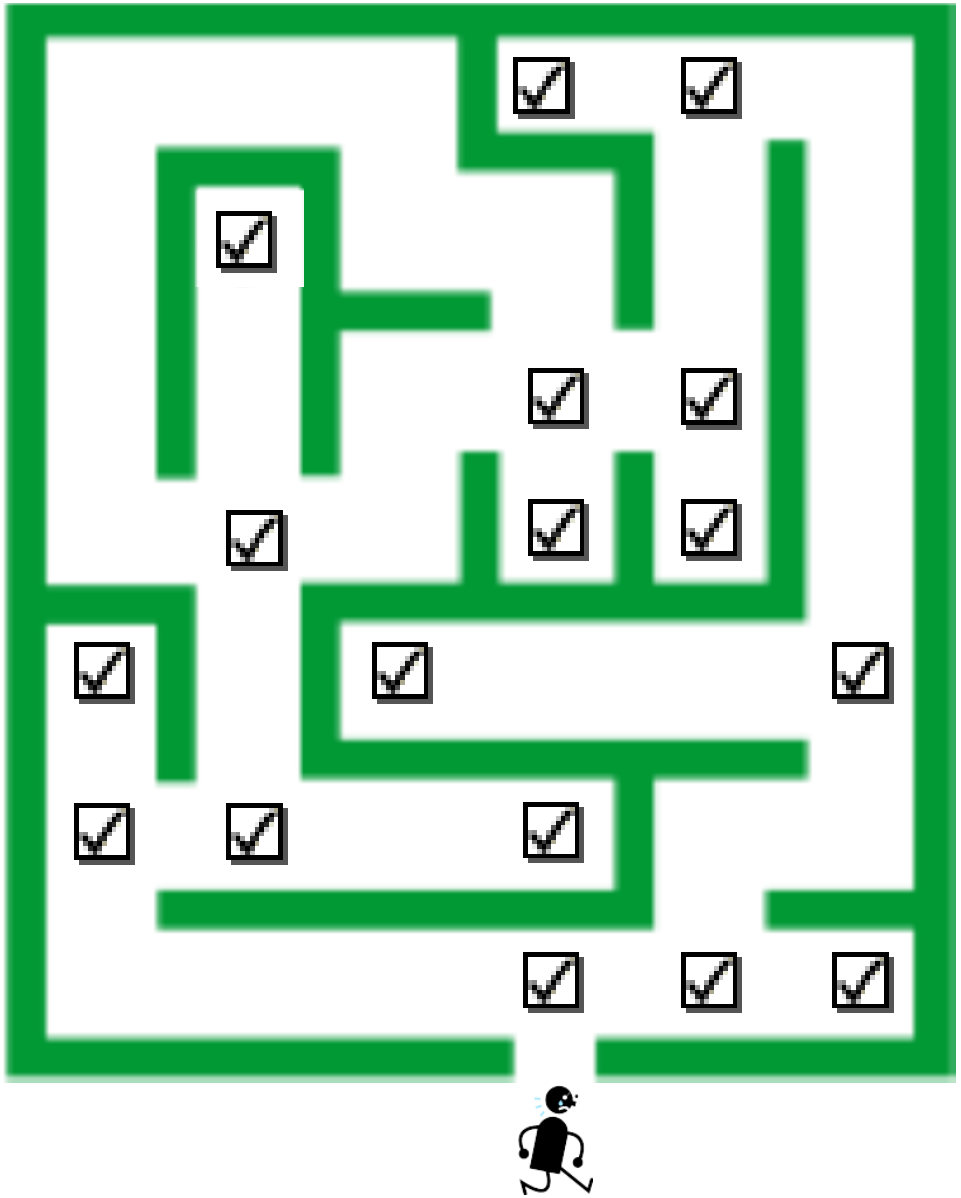
Theseus und der Minotaurus

- Annahme:
  - Der Minotaurus war gerade nicht zu Hause, sondern am Strand.

Labyrinth des Minotaurus



Labyrinth des Minotaurus



Theseus Algorithmus

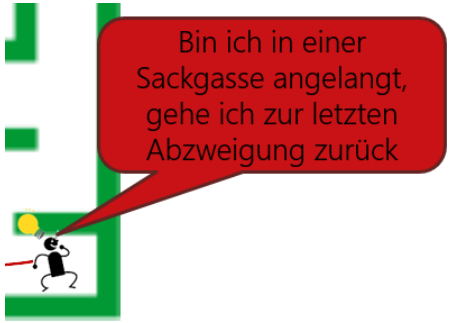
Theseus Algorithmus

- Bei der kommenden Kreuzung immer rechts abbiegen

Theseus Algorithmus

- Bei der kommenden Kreuzung immer rechts abbiegen

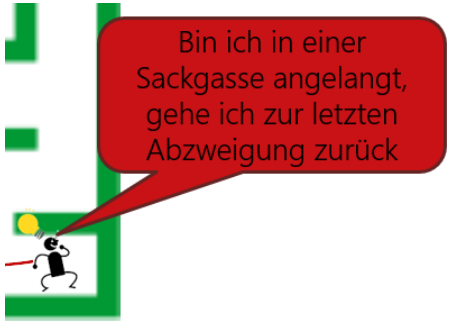
- Bei einer Sackgasse zur letzten Kreuzung zurückgehen



Theseus Algorithmus

- Bei der kommenden Kreuzung immer rechts abbiegen

- Bei einer Sackgasse zur letzten Kreuzung zurückgehen



- Beim Auftreffen auf den eigenen (roten) Faden zur letzten Kreuzung zurückgehen.

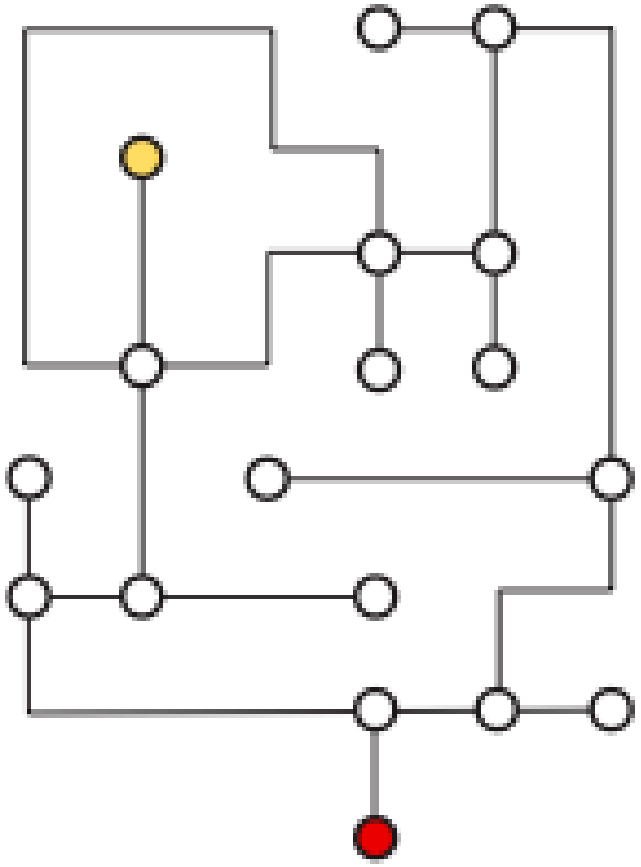
Labyrinth als Graph

- Wir überlegen uns folgendes:
  a) Wie kann das Labyrinth als Graph umgesetzt werden?

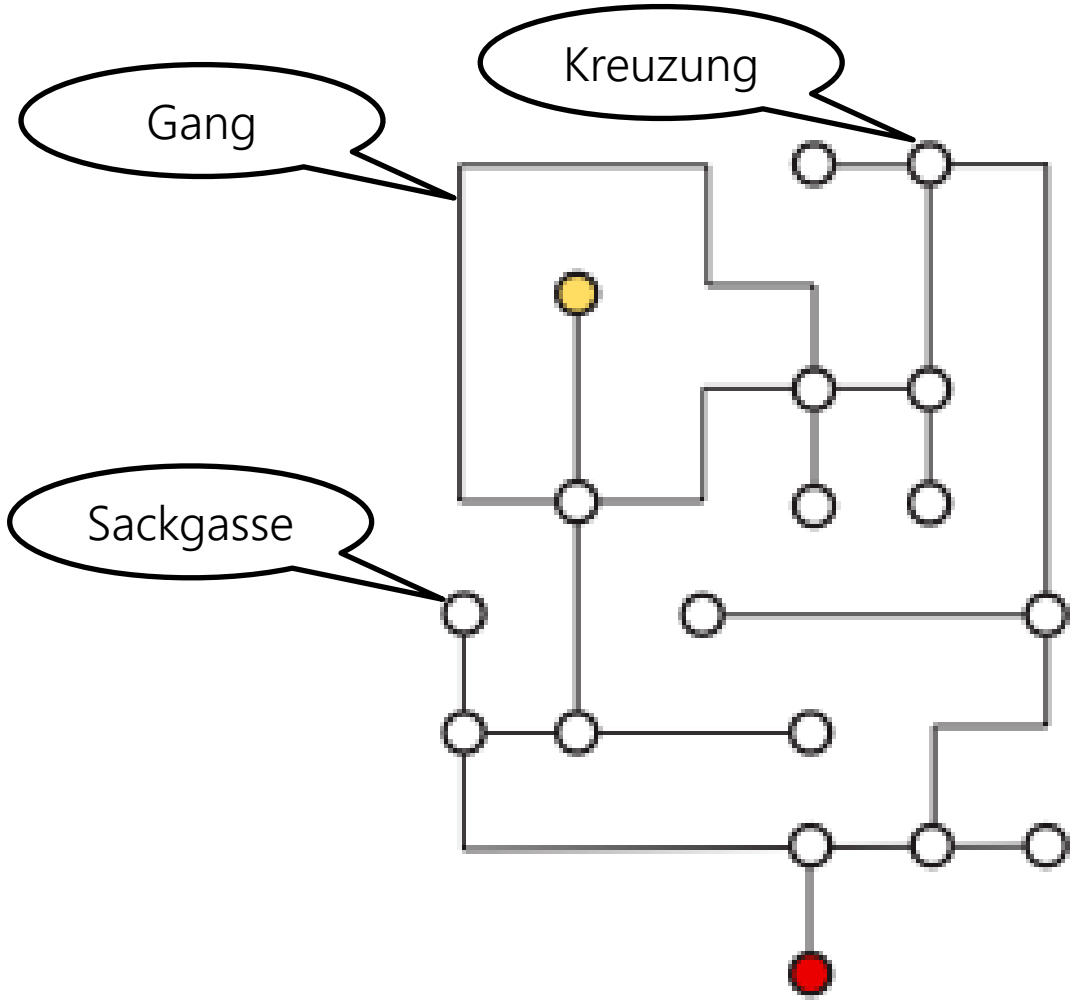
  b) Im Graphen gibt es kein rechts oder links! Wie könnte die Entscheidung über den Folgeknoten getroffen werden?

  c) Wie kann im Graph umgesetzt werden, dass Theseus auf eine Sackgasse bzw. auf seinen roten Faden getroffen ist?

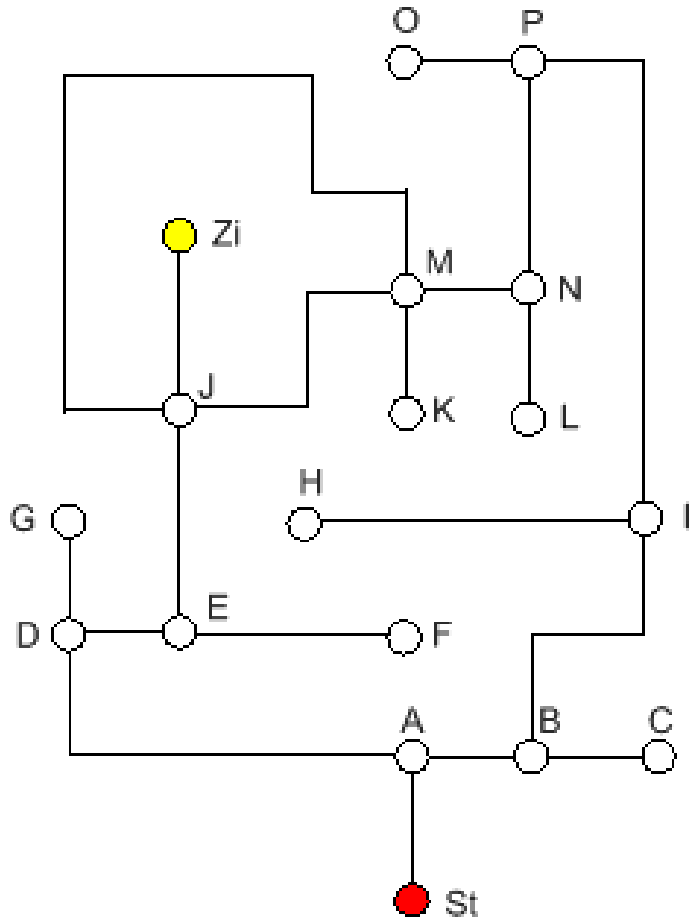
a) Umsetzung als Graph



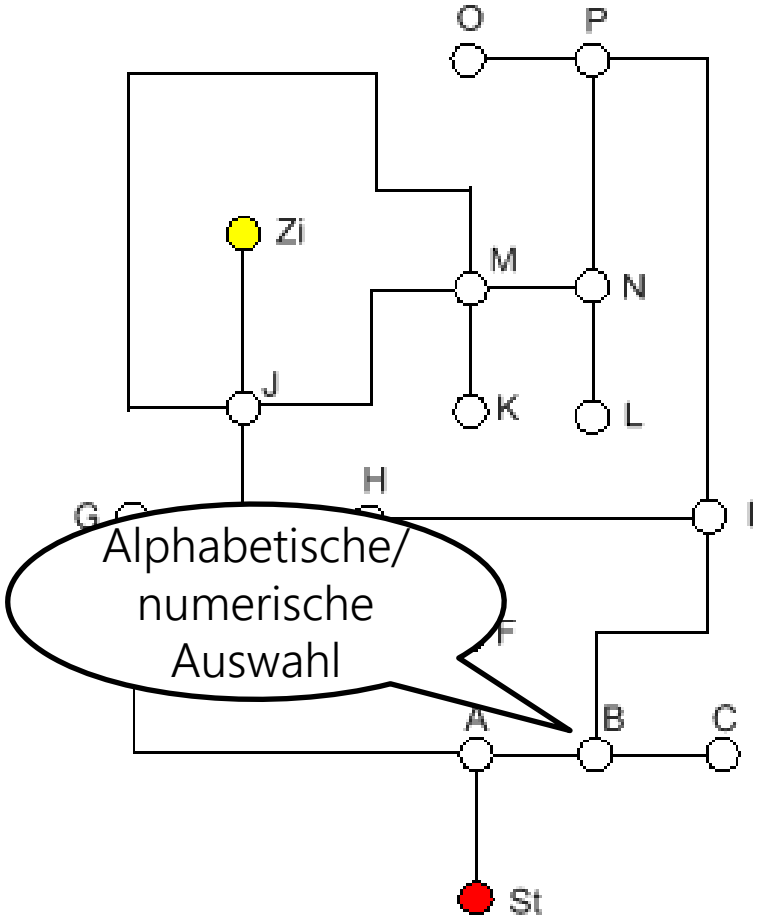
a) Umsetzung als Graph



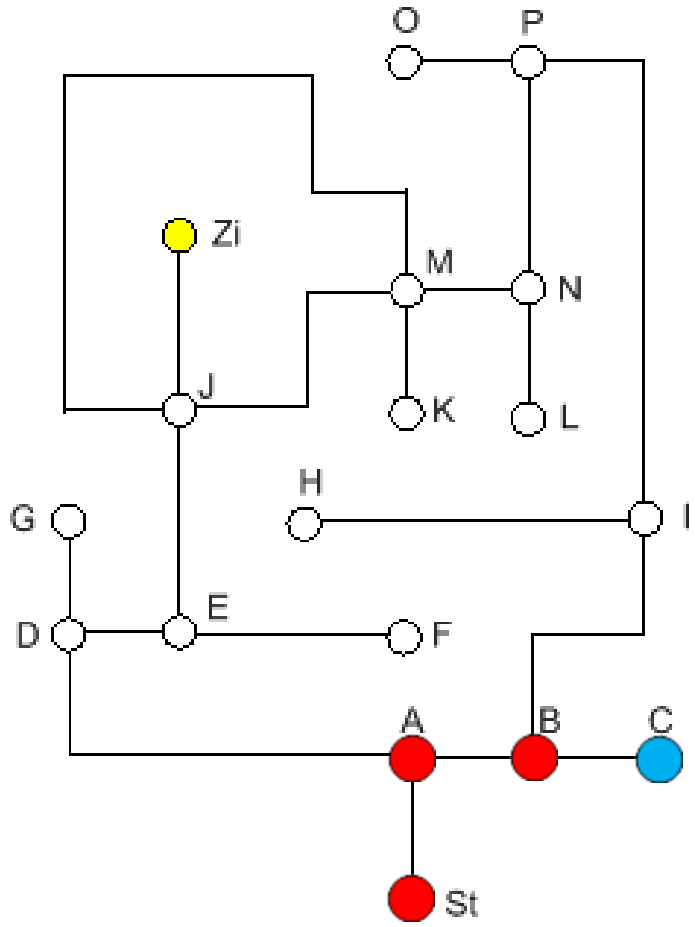
b) Entscheidung über Folgeknoten



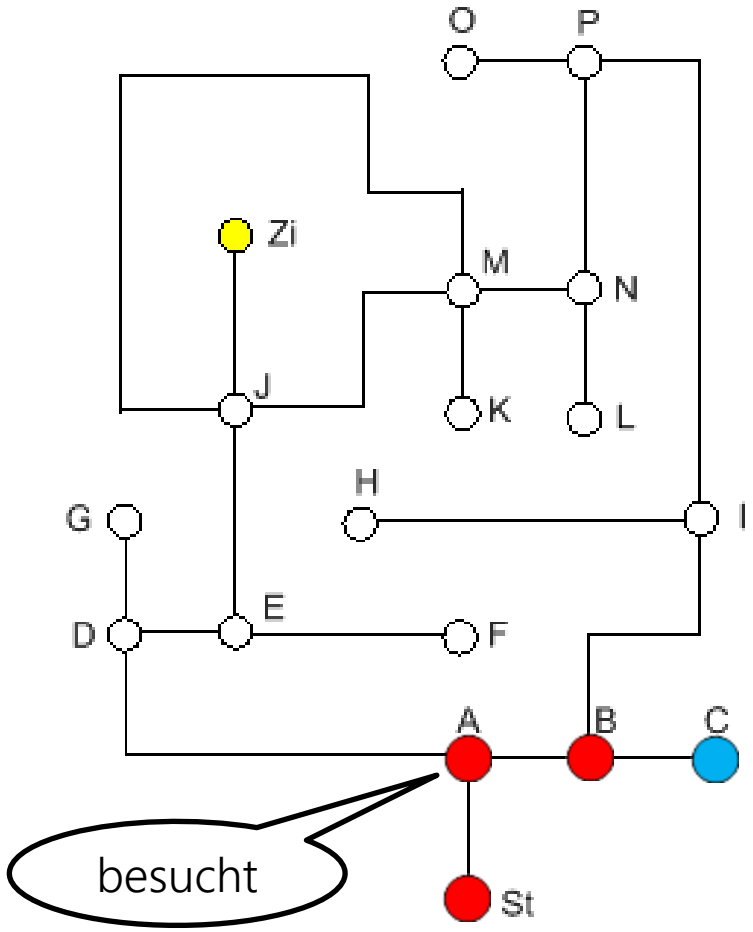
b) Entscheidung über Folgeknoten



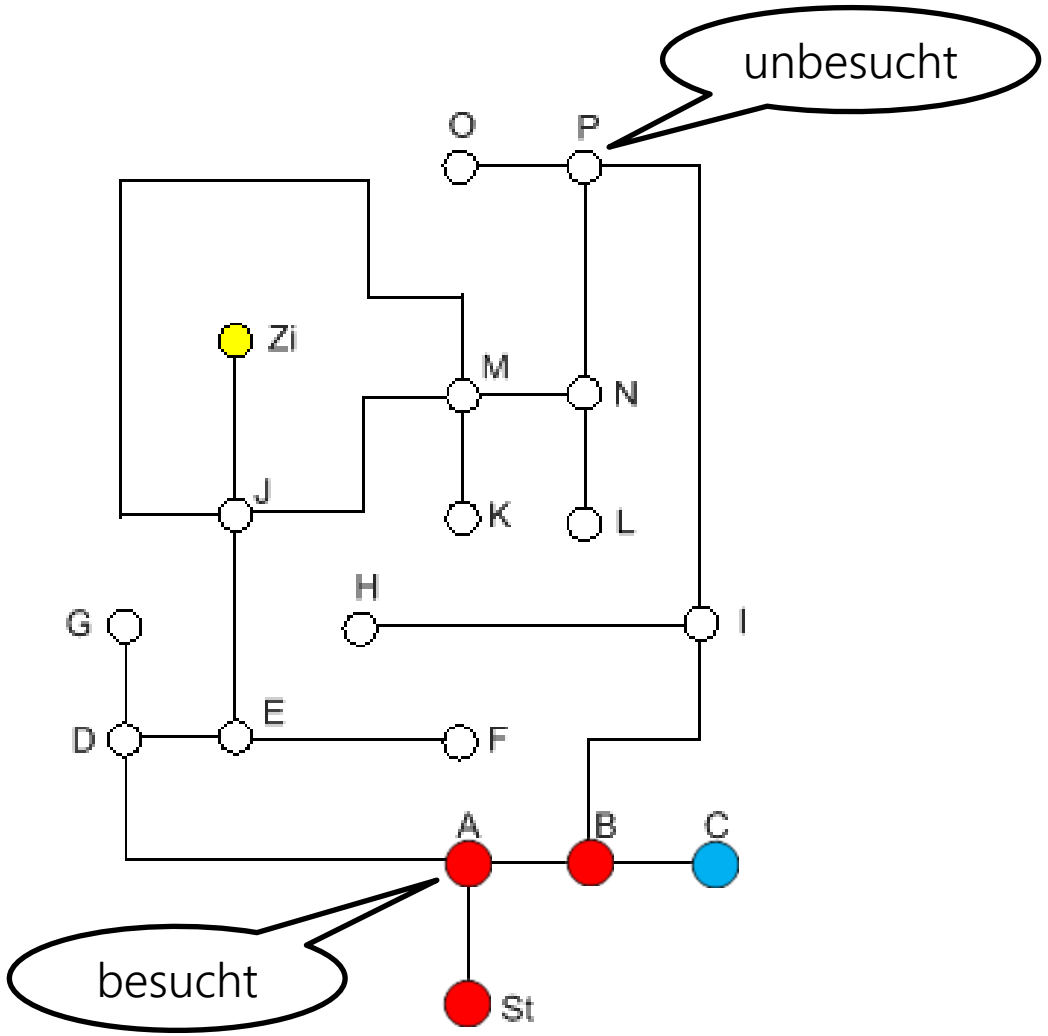
c) Trifft auf Sackgasse bzw. roten Faden



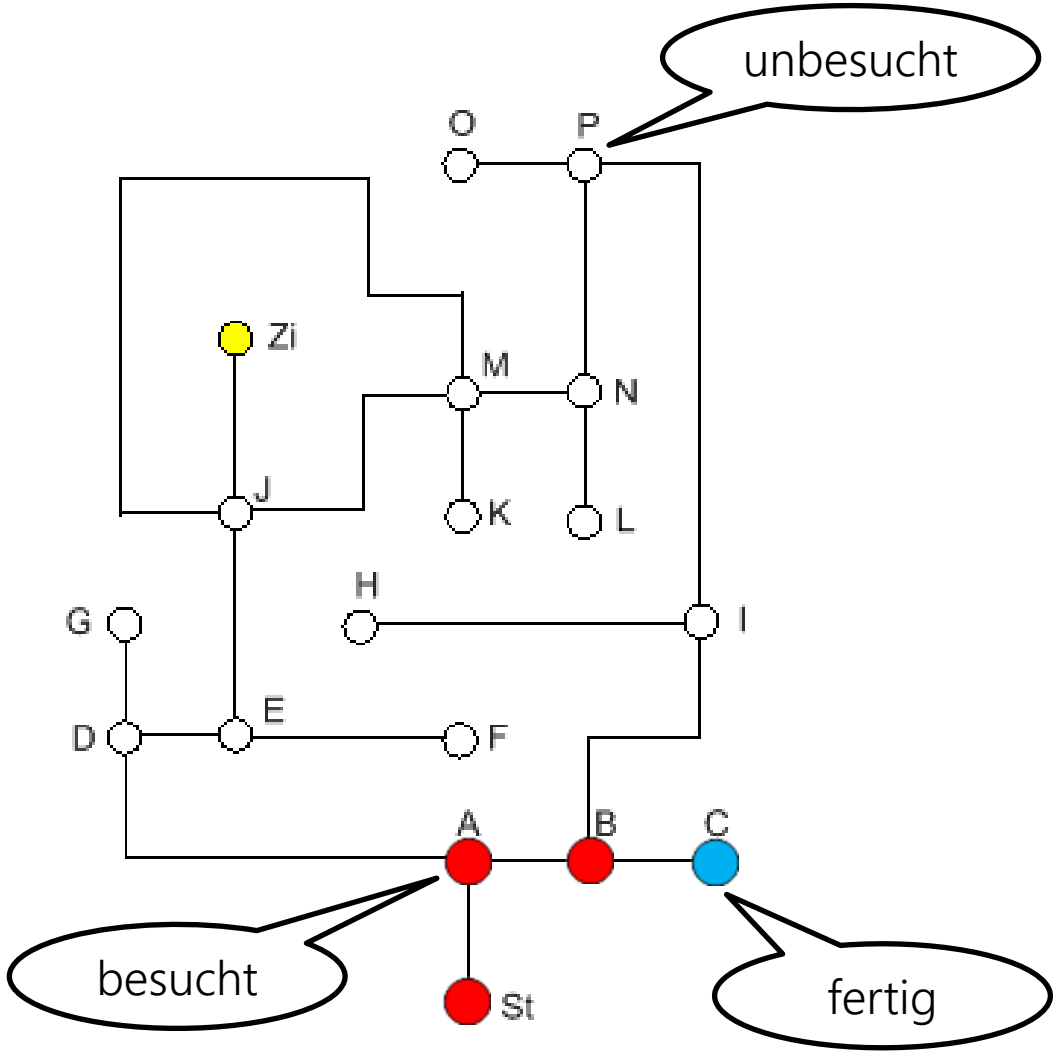
c) Knotenstatus



c) Knotenstatus



c) Knotenstatus

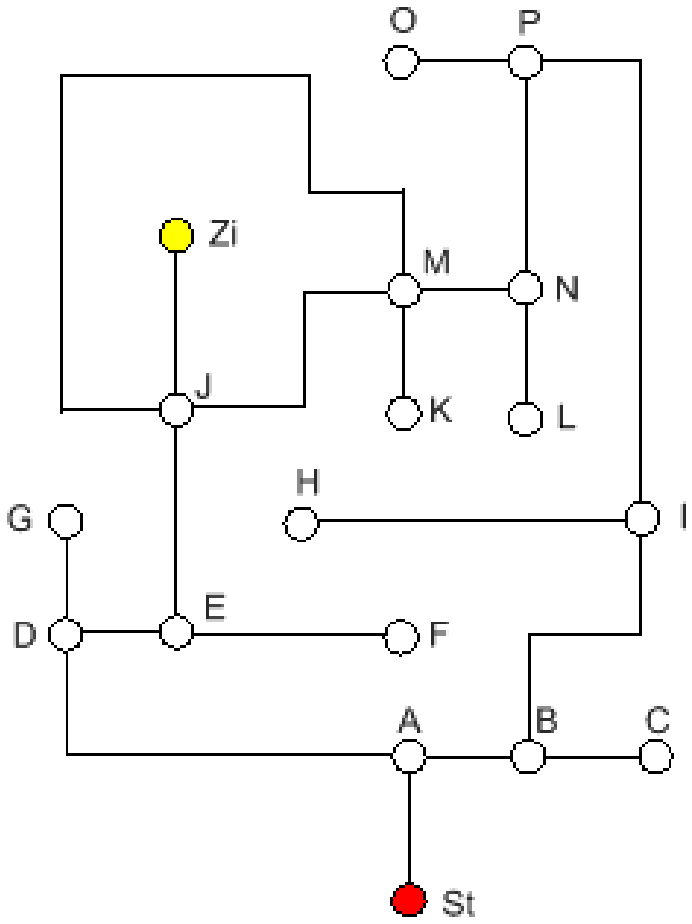


Tiefensuche - Algorithmus

(1) Wiederhole bis alle Knoten auf den Zustand fertig gesetzt sind.

(2) Setze den aktuellen Knoten auf besucht und wähle als Folgeknoten den numerisch bzw. alphabetisch niedrigsten unbesuchten Nachbarknoten.

(3) Hat ein Knoten keine unbesuchten Nachbarknoten setze ihn auf fertig. Es wird zum Vorgängerknoten zurückgekehrt.



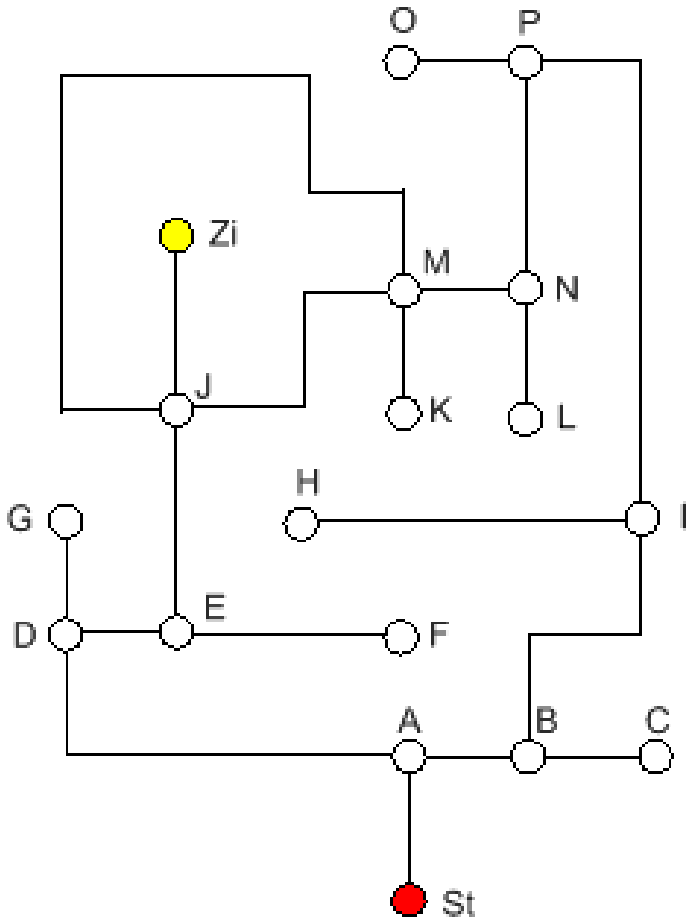
Tiefensuche - Algorithmus

- Tiefensuche Lösung:

  St, A, B, C, B, I, H, I, P, N, L, N, M, J, Zi

  (Ziel in Sichtweite)

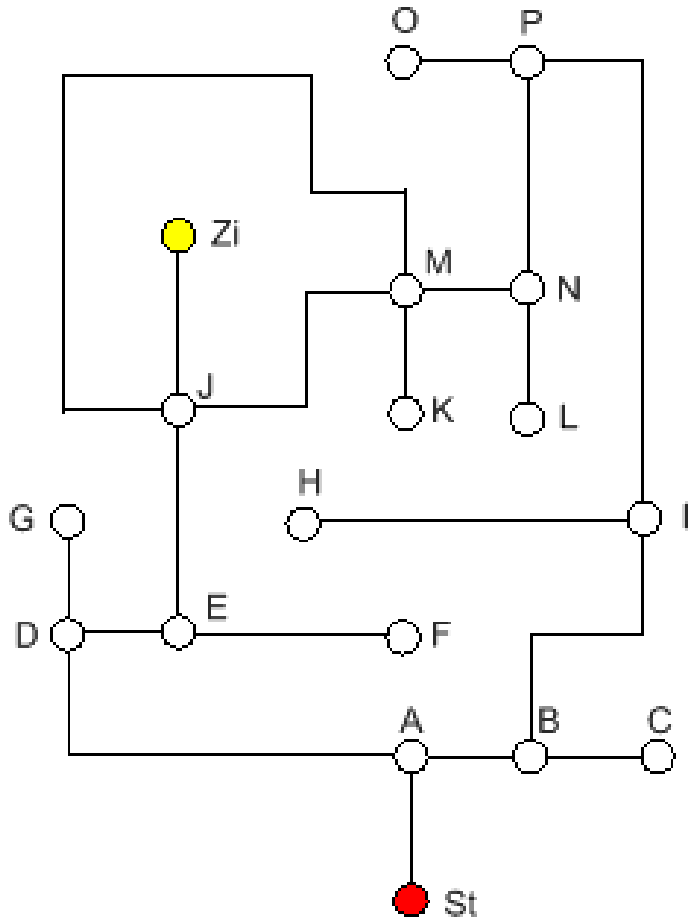
  St, A, B, C, B, I, H, I, P, N, L, N, M, J, E, D, G, D, E, F, E, J, Zi



Tiefensuche - Algorithmus

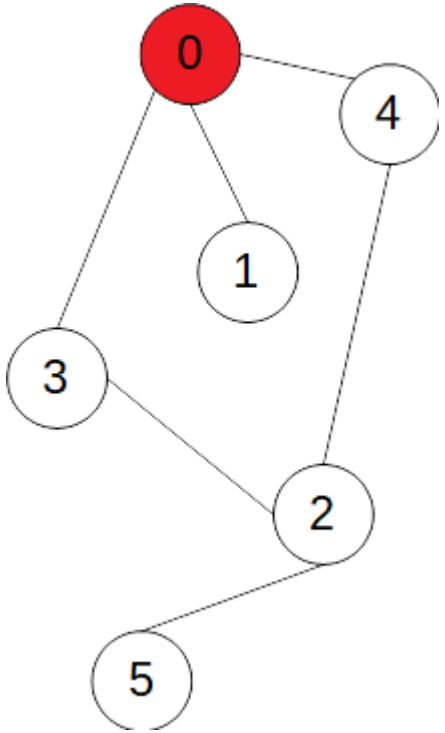
- Tiefendurchlauf Lösung:

St, A, B, C, B, I, H, I, P, N, L, N, M, J, E,
D, G, D, E, F, E, J, Zi, J, M, K, M, N, P,
O, P, I, B, A, St



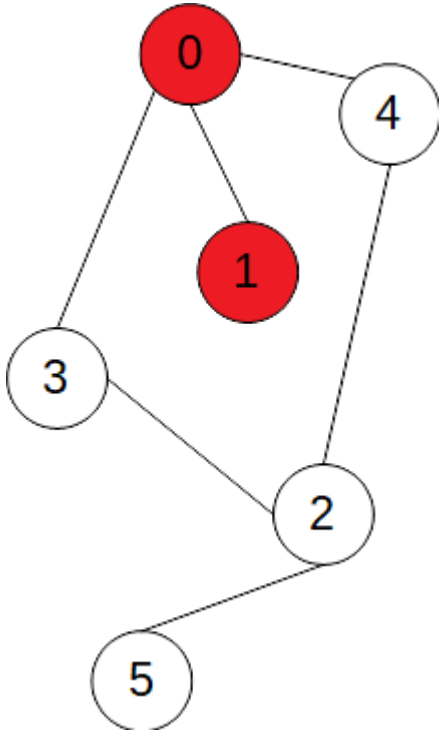
Übung Tiefendurchlauf

Schritte	Ablauf	Zustandsänderung
0		0 besucht setzen
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		



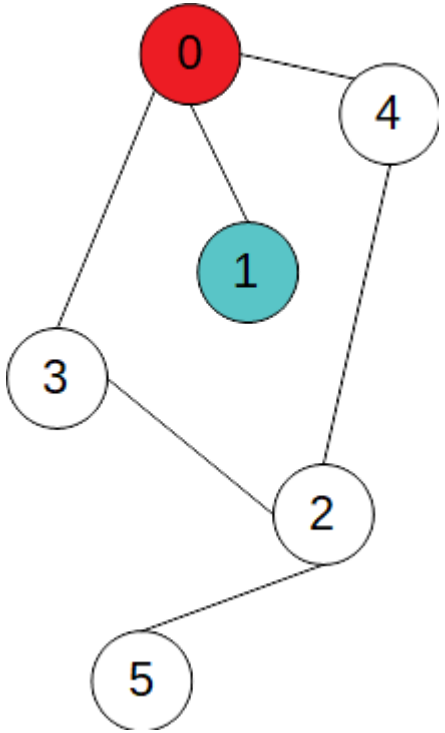
Übung Tiefendurchlauf

Schritte	Ablauf	Zustandsänderung
0		0 besucht setzen
1	1 besuchen	1 besucht setzen
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		



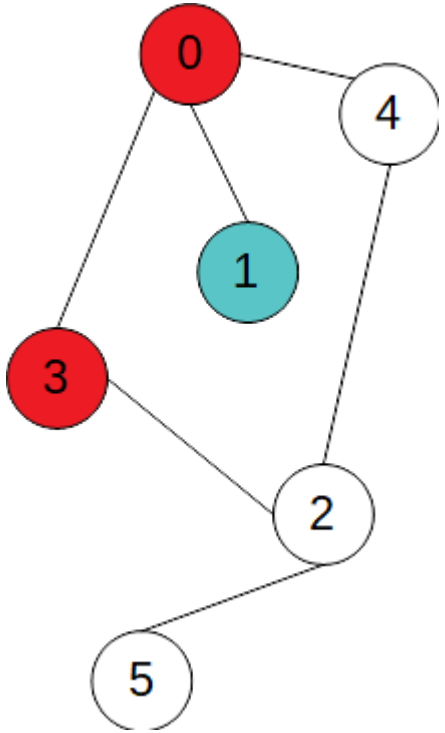
Übung Tiefendurchlauf

Schritte	Ablauf	Zustandsänderung
0		0 besucht setzen
1	1 besuchen	1 besucht setzen
2	0 besuchen	1 fertig setzen
3		
4		
5		
6		
7		
8		
9		
10		
11		



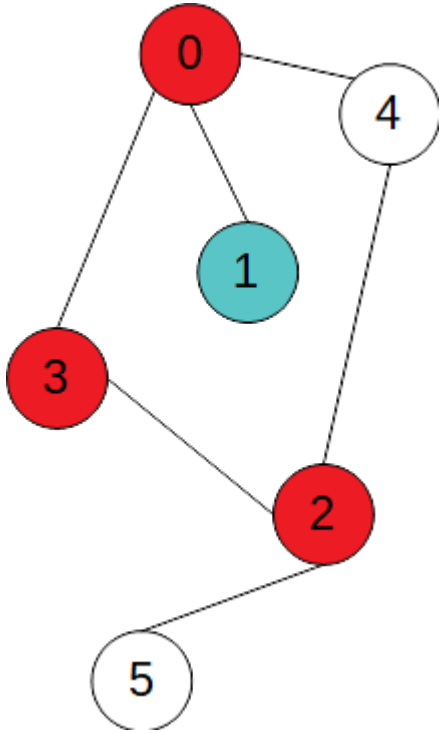
Übung Tiefendurchlauf

Schritte	Ablauf	Zustandsänderung
0		0 besucht setzen
1	1 besuchen	1 besucht setzen
2	0 besuchen	1 fertig setzen
3	3 besuchen	3 besucht setzen
4		
5		
6		
7		
8		
9		
10		
11		



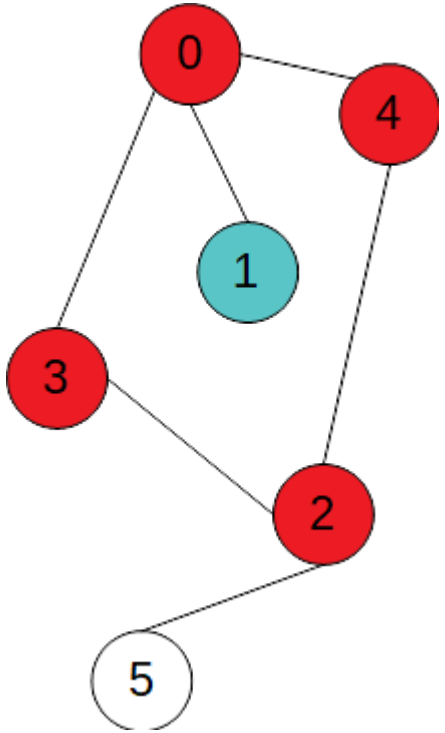
Übung Tiefendurchlauf

Schritte	Ablauf	Zustandsänderung
0		0 besucht setzen
1	1 besuchen	1 besucht setzen
2	0 besuchen	1 fertig setzen
3	3 besuchen	3 besucht setzen
4	2 besuchen	2 besucht setzen
5		
6		
7		
8		
9		
10		
11		



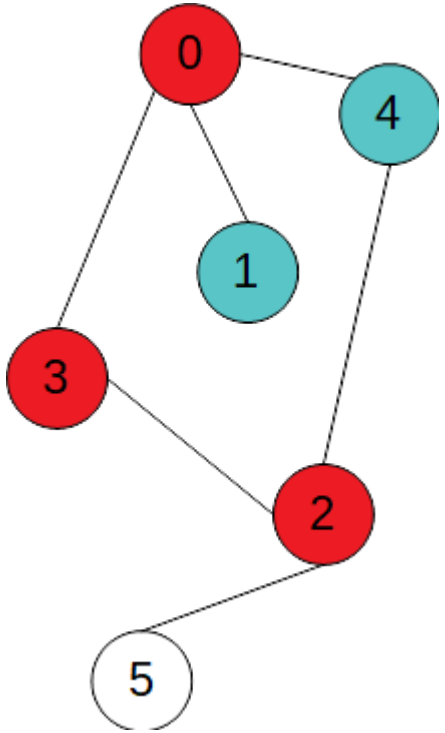
Übung Tiefendurchlauf

Schritte	Ablauf	Zustandsänderung
0		0 besucht setzen
1	1 besuchen	1 besucht setzen
2	0 besuchen	1 fertig setzen
3	3 besuchen	3 besucht setzen
4	2 besuchen	2 besucht setzen
5	4 besuchen	4 besucht setzen
6		
7		
8		
9		
10		
11		



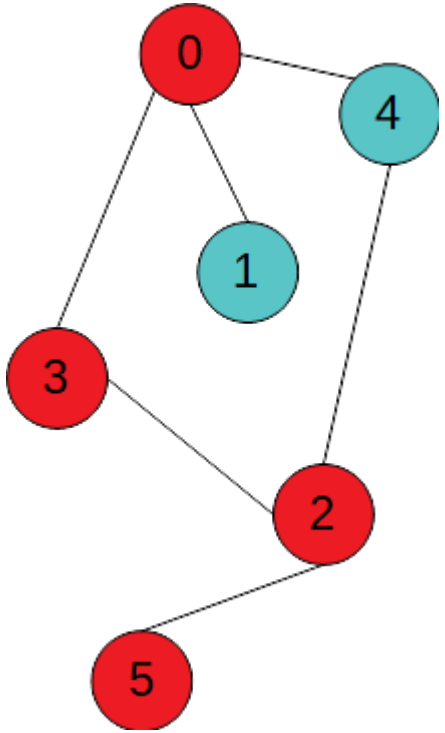
Übung Tiefendurchlauf

Schritte	Ablauf	Zustandsänderung
0		0 besucht setzen
1	1 besuchen	1 besucht setzen
2	0 besuchen	1 fertig setzen
3	3 besuchen	3 besucht setzen
4	2 besuchen	2 besucht setzen
5	4 besuchen	4 besucht setzen
6	2 besuchen	4 fertig setzen
7		
8		
9		
10		
11		



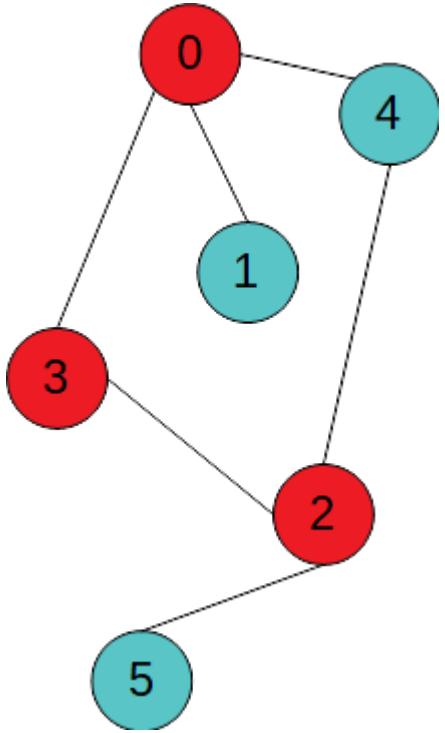
Übung Tiefendurchlauf

Schritte	Ablauf	Zustandsänderung
0		0 besucht setzen
1	1 besuchen	1 besucht setzen
2	0 besuchen	1 fertig setzen
3	3 besuchen	3 besucht setzen
4	2 besuchen	2 besucht setzen
5	4 besuchen	4 besucht setzen
6	2 besuchen	4 fertig setzen
7	5 besuchen	5 besucht setzen
8		
9		
10		
11		



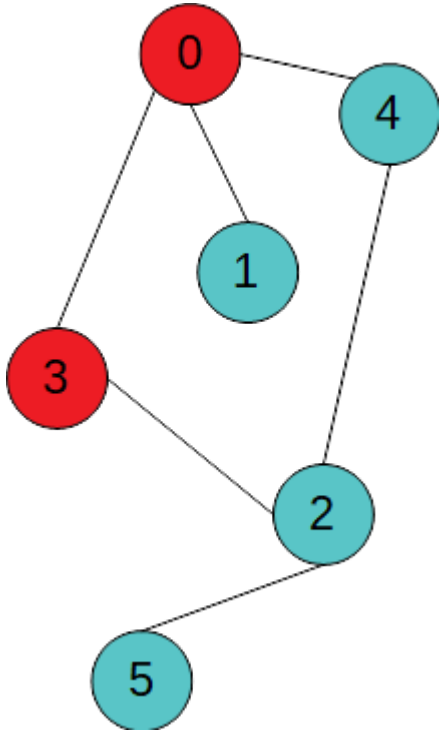
Übung Tiefendurchlauf

Schritte	Ablauf	Zustandsänderung
0		0 besucht setzen
1	1 besuchen	1 besucht setzen
2	0 besuchen	1 fertig setzen
3	3 besuchen	3 besucht setzen
4	2 besuchen	2 besucht setzen
5	4 besuchen	4 besucht setzen
6	2 besuchen	4 fertig setzen
7	5 besuchen	5 besucht setzen
8	2 besuchen	5 fertig setzen
9		
10		
11		



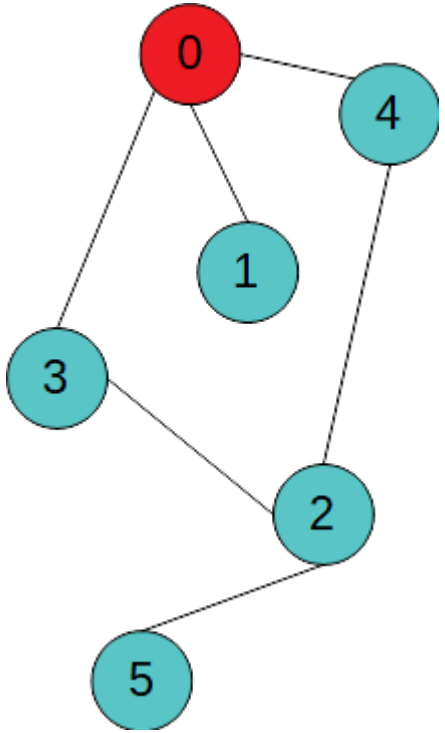
Übung Tiefendurchlauf

Schritte	Ablauf	Zustandsänderung
0		0 besucht setzen
1	1 besuchen	1 besucht setzen
2	0 besuchen	1 fertig setzen
3	3 besuchen	3 besucht setzen
4	2 besuchen	2 besucht setzen
5	4 besuchen	4 besucht setzen
6	2 besuchen	4 fertig setzen
7	5 besuchen	5 besucht setzen
8	2 besuchen	5 fertig setzen
9	3 besuchen	2 fertig setzen
10		
11		



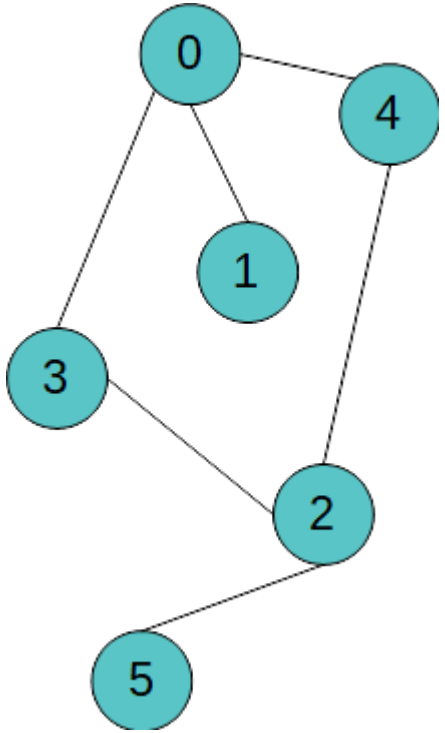
Übung Tiefendurchlauf

Schritte	Ablauf	Zustandsänderung
0		0 besucht setzen
1	1 besuchen	1 besucht setzen
2	0 besuchen	1 fertig setzen
3	3 besuchen	3 besucht setzen
4	2 besuchen	2 besucht setzen
5	4 besuchen	4 besucht setzen
6	2 besuchen	4 fertig setzen
7	5 besuchen	5 besucht setzen
8	2 besuchen	5 fertig setzen
9	3 besuchen	2 fertig setzen
10	0 besuchen	3 fertig setzen
11		

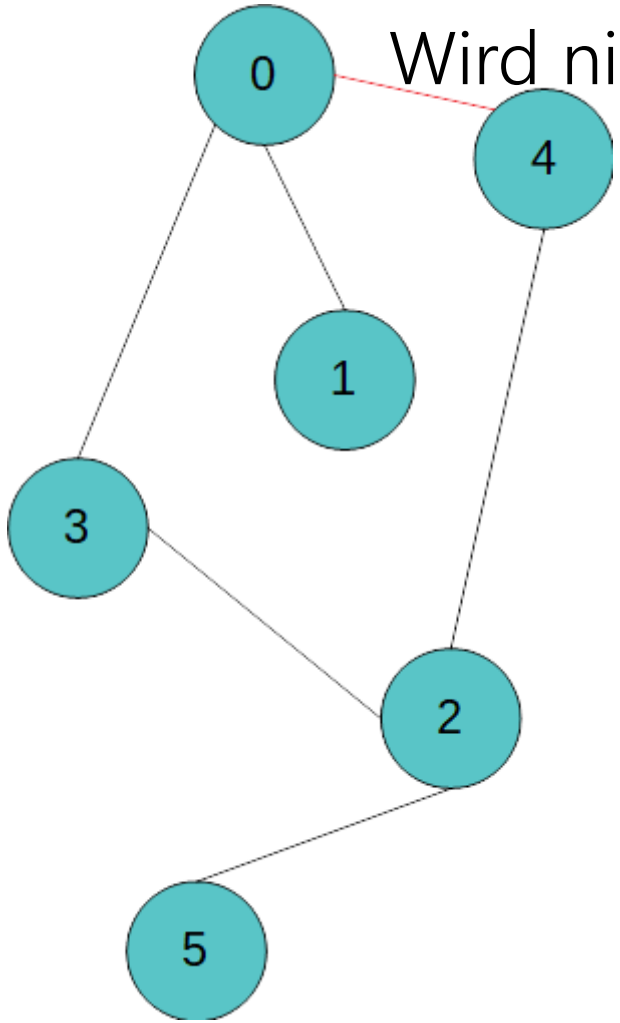


Übung Tiefendurchlauf

Schritte	Ablauf	Zustandsänderung
0		0 besucht setzen
1	1 besuchen	1 besucht setzen
2	0 besuchen	1 fertig setzen
3	3 besuchen	3 besucht setzen
4	2 besuchen	2 besucht setzen
5	4 besuchen	4 besucht setzen
6	2 besuchen	4 fertig setzen
7	5 besuchen	5 besucht setzen
8	2 besuchen	5 fertig setzen
9	3 besuchen	2 fertig setzen
10	0 besuchen	3 fertig setzen
11	-	0 fertig setzen



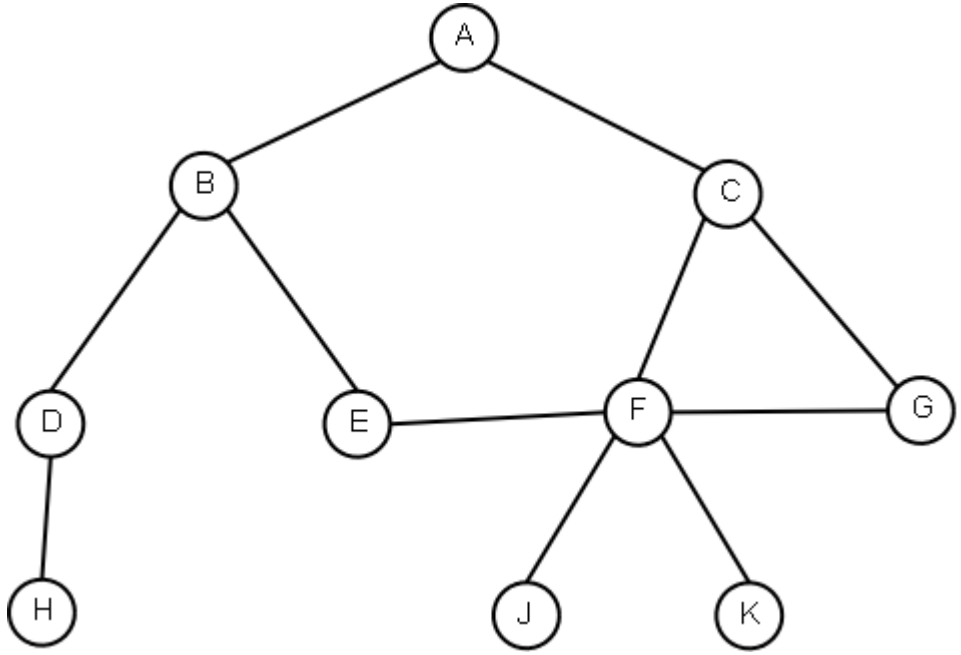
Übung Tiefensuche



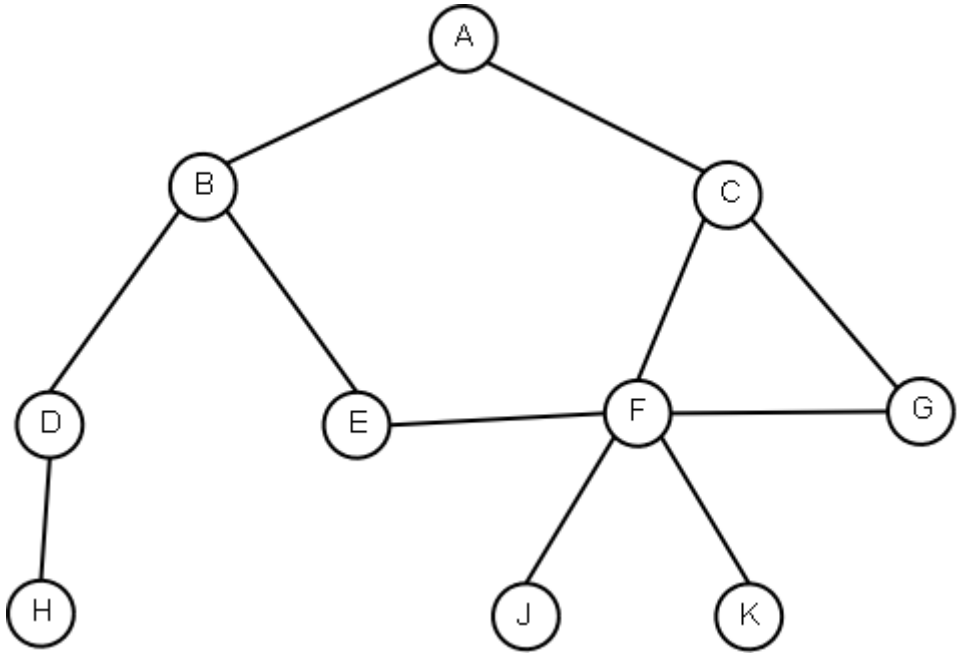
Wird nicht durchlaufen

Übung Tiefensuche / Tiefensuchbaum

- Startknoten A



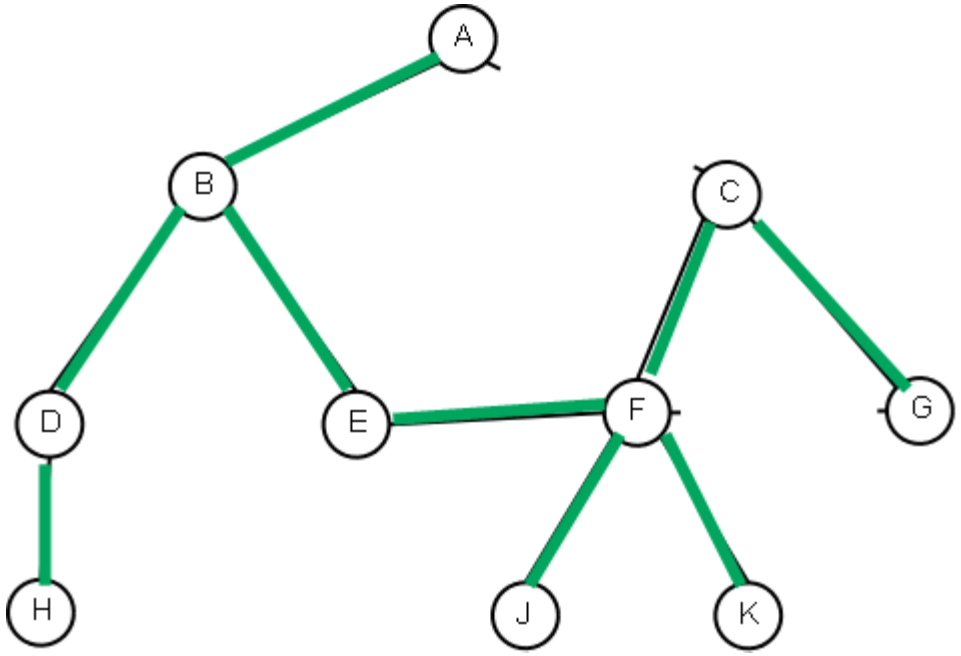
Übung Tiefensuche / Tiefensuchbaum



- Lösung:

A – B – D – H – D – B – E – F – C – G – C – F – J – F – K – F – E – B – A

Welche Kanten wurden genutzt?



- Lösung:

A – B – D – H – D – B – E – F – C – G – C – F – J – F – K – F – E – B – A

Zusammenfassung

1. Wiederhole bis alle Knoten auf den Zustand fertig gesetzt sind:
2. Setze den aktuellen Knoten auf besucht und wähle als nächsten Knoten den numerisch bzw. alphabetisch niedrigsten unbesuchten Knoten. (ausgehend vom aktuellen Knoten)
3. Hat ein Knoten keine unbesuchten Nachbarknoten (mehr), setze ihn auf fertig: Alle abgehenden Zweige des Knotens wurden durchsucht. Es wird zum Vorgängerknoten zurückgekehrt. (Backtracking)

- Für jeden Knoten wird exakt der gleiche Algorithmus ausgeführt. Es handelt sich demnach um einen rekursiven Algorithmus:
- „Gehe zu einem Knoten, an dem du noch nicht warst und führe von dort den Tiefendurchlauf aus.“

Implementierung der Tiefensuche (Vorbereitung)

- **Vorarbeit in der Klasse Knoten:**
  - Jeder Knoten kann besucht und fertig gesetzt werden. Setze dies mit booleschen Attributen in der Klasse KNOTEN um.
- **Vorbereitung in der Klasse Graph → Methode tiefendurchlauf(String startknoten):**
- Existiert der angegebene Startknoten im Graphen? Falls nicht -> Methode abbrechen!
- Alle Knoten müssen auf unbesucht und unfertig gesetzt werden.
- Lagere die eigentliche Tiefensuche in eine Hilfsmethode **besuchenTiefe(int startnummer)** aus.

Implementierung der Tiefensuche (Algorithmus)

- Setze den aktuellen Knoten auf besucht (inkl. Konsolenausgabe)

→ Überlege dir, wie man herausfindet, welche Knoten unbesuchte Nachbarknoten sind!

	IN	M	A	UL
IN	0	1	1	-1
M	1	0	1	-1
A	1	1	0	1
UL	-1	-1	1	0

- Besuche rekursiv (jeden) unbesuchten Nachbarknoten!
- Sind alle Nachbarknoten besucht, so wird der Knoten (inkl. Konsolenausgabe) auf fertig gesetzt.

- Teste deine Methode mit der Klasse Testgraphen.