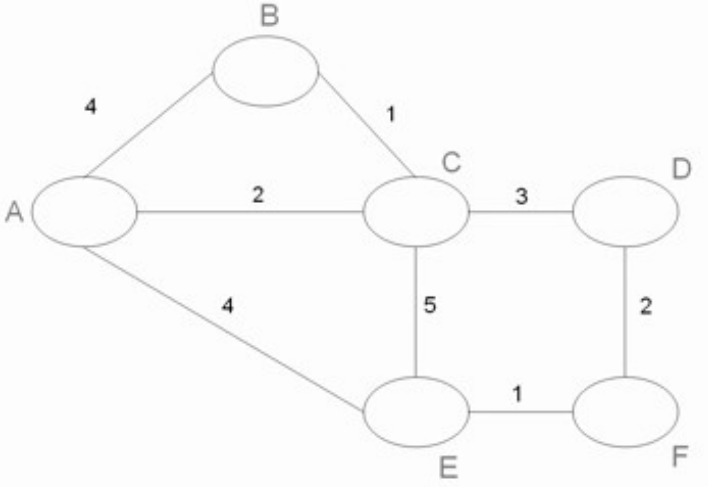
Dijkstra
Single Source Shortest Path

Der Dijkstra-Algorithmus

Von einem gegebenen Startknoten aus:
die kürzesten Wege zu allen anderen Knoten

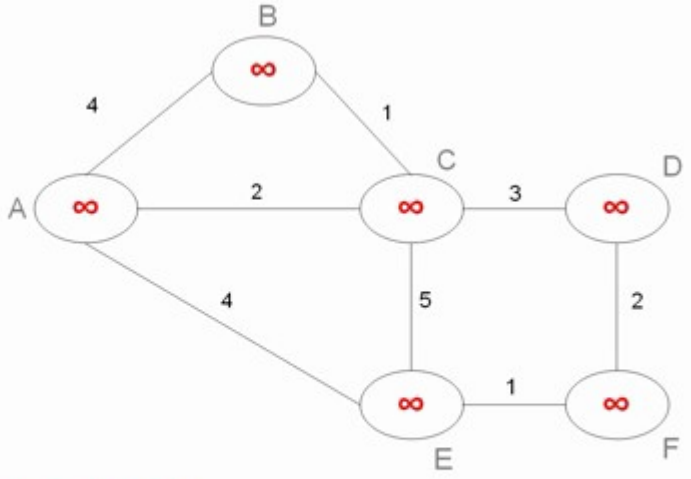
A		
B		
C		
D		
E		
F		



https://www.youtube.com/watch?v=S8y-Sk7u1So&feature=youtu.be

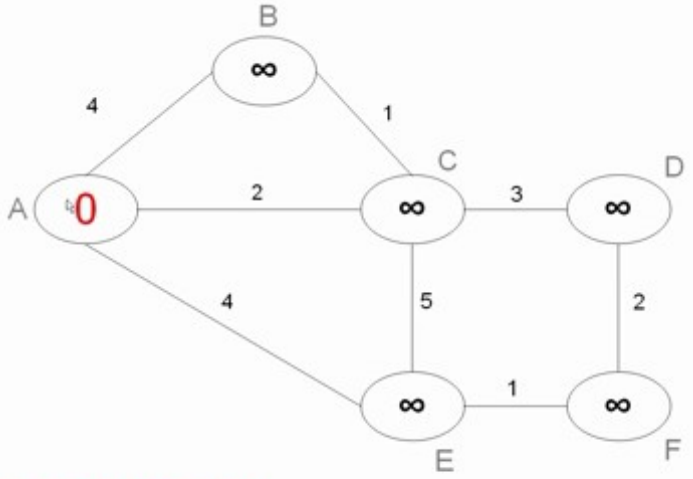
Vorarbeiten

A	∞	
B	∞	
C	∞	
D	∞	
E	∞	
F	∞	



Vorarbeit 1:
- setze jeden Knoten als unbesucht
- setze jede Distanz auf unendlich
- setze jeden Vorgänger auf *null*

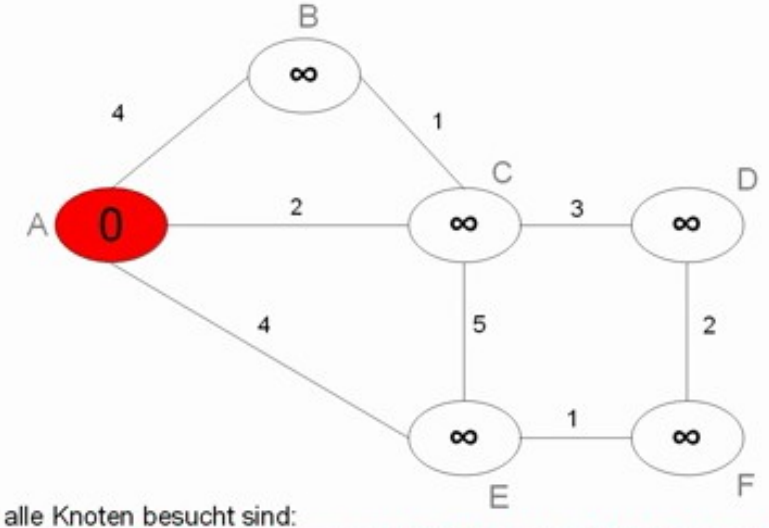
A	0	A
B	∞	
C	∞	
D	∞	
E	∞	
F	∞	



Vorarbeit 2:
- setze die Distanz des Startknotens auf 0
- setze seinen Vorgänger auf sich selber (hilfreich, aber nicht nötig)

Algorithmus: start

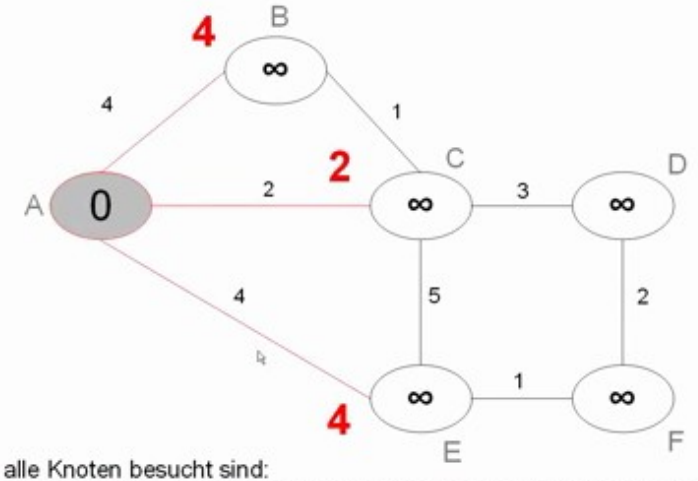
A	0	A
B	∞	
C	∞	
D	∞	
E	∞	
F	∞	



Wiederhole, bis alle Knoten besucht sind:
- setze den unbesuchten **Knoten** mit der geringsten Distanz als aktuell und besucht
- für alle unbesuchten **Nachbarn**: addiere eigene Distanz und das Kantengewicht
  - wenn Summe geringer ist als deren aktuelle Distanz,
    - dann setze sie
    - und setze dich als seinen Vorgänger

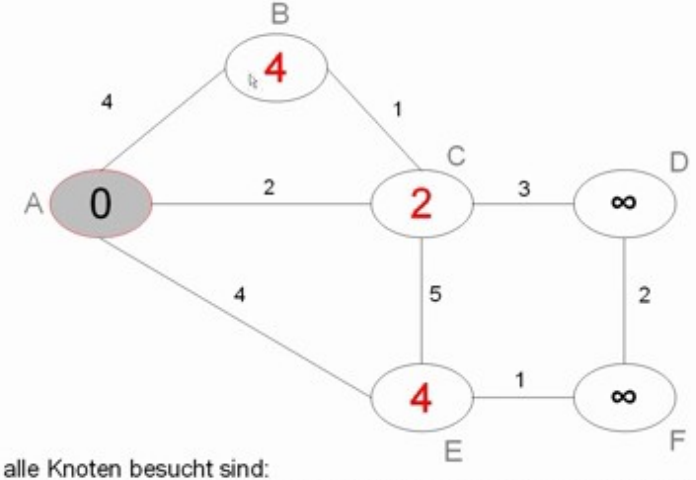
Nachbarn Distanzen berechnen

A	0	A
B	∞	
C	∞	
D	∞	
E	∞	
F	∞	



Wiederhole, bis alle Knoten besucht sind:
- setze den unbesuchten **Knoten** mit der geringsten Distanz als aktuell und besucht
- für alle unbesuchten **Nachbarn**: addiere eigene Distanz und das Kantengewicht
  - wenn Summe geringer ist als deren aktuelle Distanz,
    - dann setze sie
    - und setze dich als seinen Vorgänger

A	0	A
B	4	A
C	2	A
D	∞	
E	4	A
F	∞	



Wiederhole, bis alle Knoten besucht sind:
- setze den unbesuchten **Knoten** mit der geringsten Distanz als aktuell und besucht
- für alle unbesuchten **Nachbarn**: addiere eigene Distanz und das Kantengewicht
  - wenn Summe geringer ist als deren aktuelle Distanz,
    - dann setze sie
    - und setze dich als seinen Vorgänger

Nachbar wählen

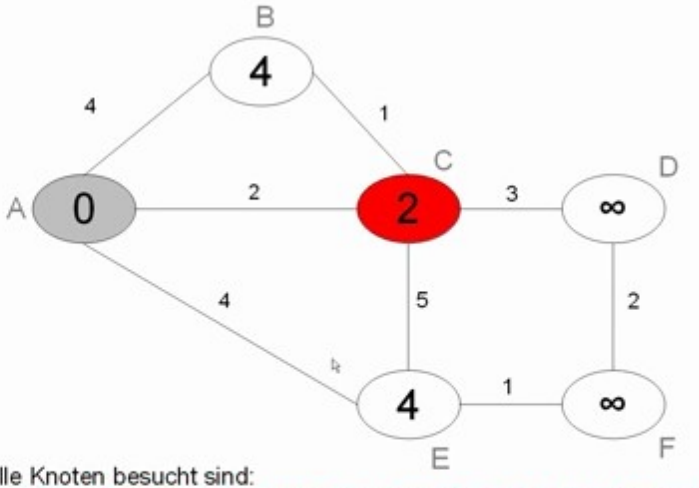
Nun beginnen wir wieder von vorne.

Wir haben 5 unbesuchte Knoten (B,C,D,E,F)

Setze den unbesuchten Knoten mit der geringsten Distanz als aktuell und besucht.

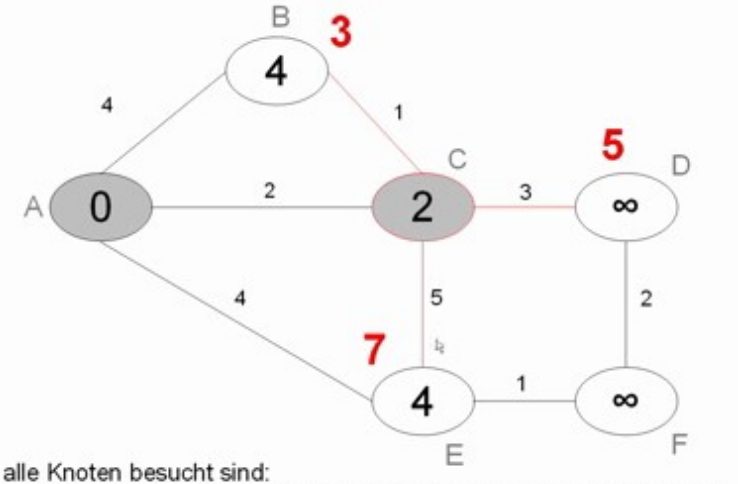
Das ist C mit der Distanz 2

A	0	A
B	4	A
C	2	A
D	∞	
E	4	A
F	∞	



Nachbar: neue Distanzen kürzer?

A	0	A
B	4	A
C	2	A
D	∞	
E	4	A
F	∞	



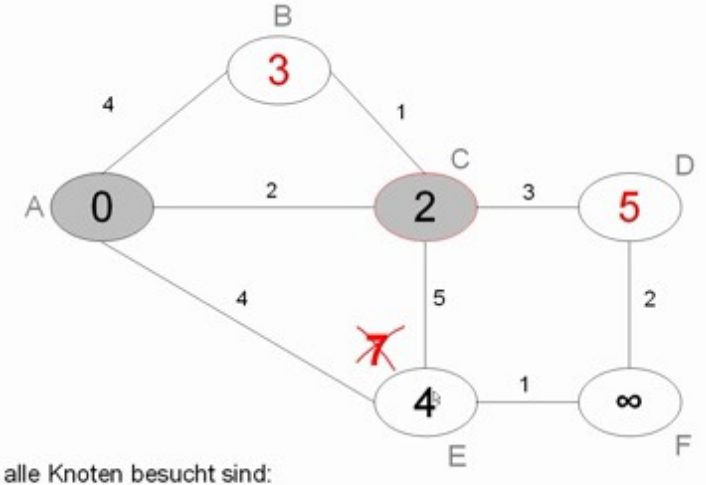
Wiederhole, bis alle Knoten besucht sind:
- setze den unbesuchten **Knoten** mit der geringsten Distanz als aktuell und besucht
- für alle unbesuchten **Nachbarn**: addiere eigene Distanz und das Kantengewicht
  - wenn Summe geringer ist als deren aktuelle Distanz,
    - dann setze sie
    - und setze dich als seinen Vorgänger

Es gibt also einen Weg:
- von A nach B (Distanz 3), über C
- von A nach D (Distanz 5), über C
- von A nach E (Distanz 7), über C

Vergleiche bisherige u. neue Distanzen:
- in E ist bereits eine kleinere Distanz (4) gespeichert (von A nach E)
- ~~Wir streichen also den Weg von A nach E (Distanz 7), der über C läuft~~

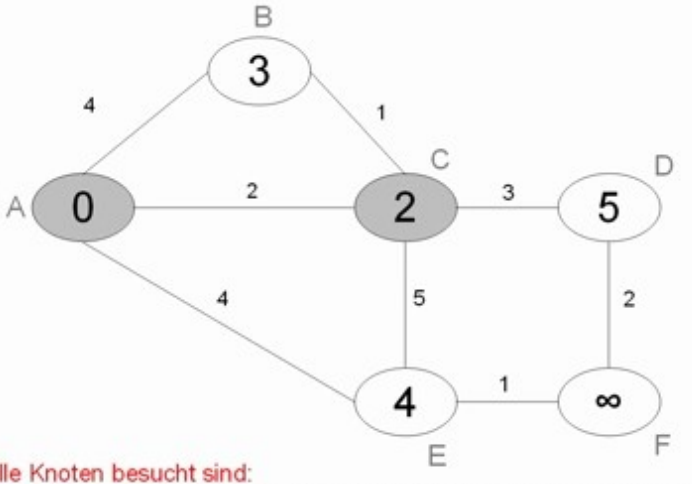
Cont.

A	0	A
B	3	C
C	2	A
D	5	C
E	4	A
F	∞	



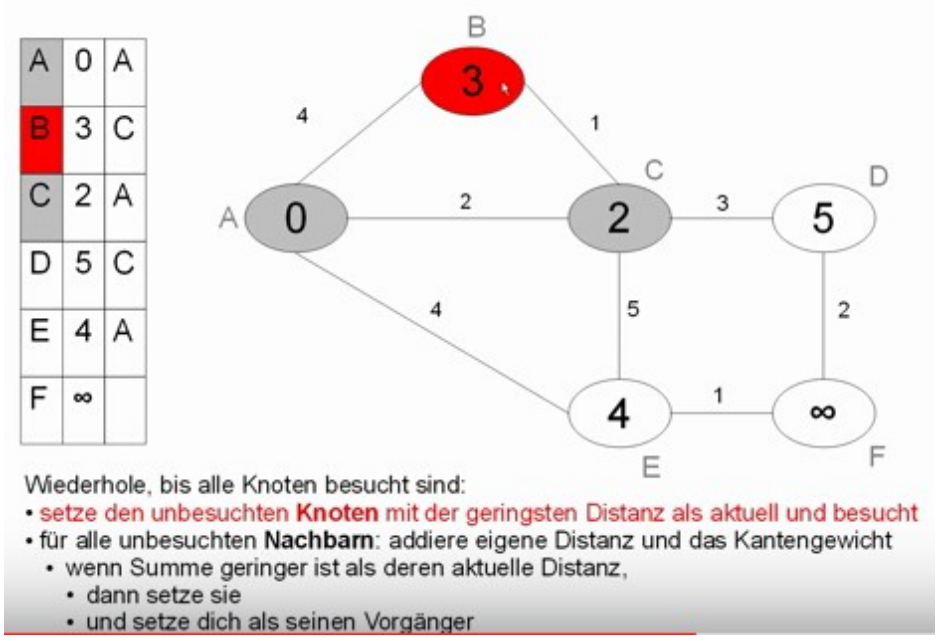
Wiederhole, bis alle Knoten besucht sind:
- setze den unbesuchten **Knoten** mit der geringsten Distanz als aktuell und besucht
- für alle unbesuchten **Nachbarn**: addiere eigene Distanz und das Kantengewicht
  - wenn Summe geringer ist als deren aktuelle Distanz,
    - dann setze sie
    - und setze dich als seinen Vorgänger

A	0	A
B	3	C
C	2	A
D	5	C
E	4	A
F	∞	



Wiederhole, bis alle Knoten besucht sind:
- setze den unbesuchten **Knoten** mit der geringsten Distanz als aktuell und besucht
- für alle unbesuchten **Nachbarn**: addiere eigene Distanz und das Kantengewicht
  - wenn Summe geringer ist als deren aktuelle Distanz,
    - dann setze sie
    - und setze dich als seinen Vorgänger

Alle Nachbarn sind besucht



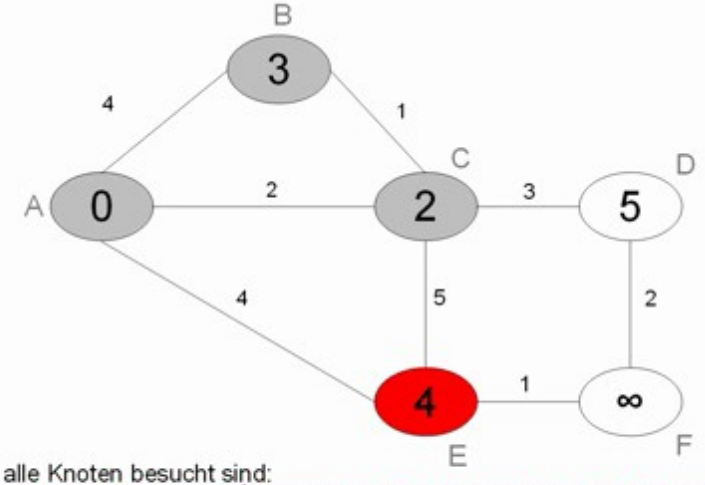
Für alle unbesuchten Nachbarn von B ….

ABER: Es gibt keine unbesuchten Nachbarn von B

→ wir sind fertig.

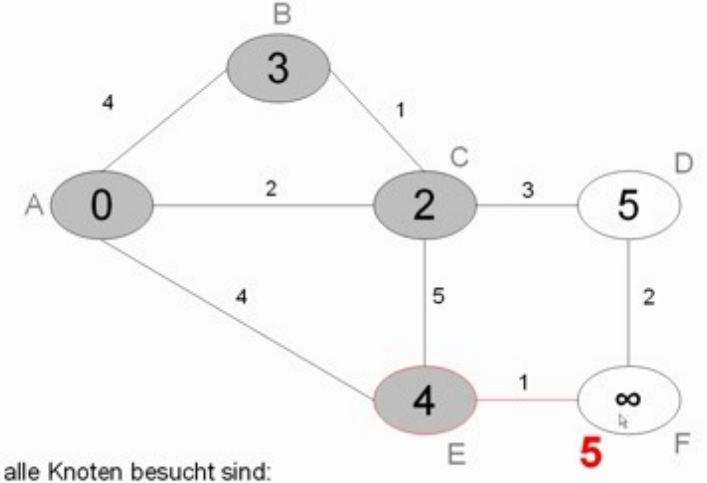
beginne wieder von vorn: bei E

A	0	A
B	3	C
C	2	A
D	5	C
E	4	A
F	∞	



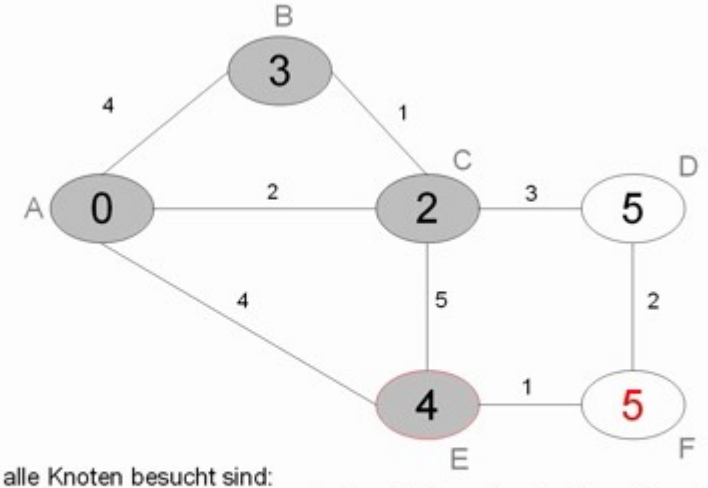
Wiederhole, bis alle Knoten besucht sind:
- setze den unbesuchten **Knoten** mit der geringsten Distanz als aktuell und besucht
- für alle unbesuchten **Nachbarn**: addiere eigene Distanz und das Kantengewicht
  - wenn Summe geringer ist als deren aktuelle Distanz,
    - dann setze sie
    - und setze dich als seinen Vorgänger

A	0	A
B	3	C
C	2	A
D	5	C
E	4	A
F	∞	



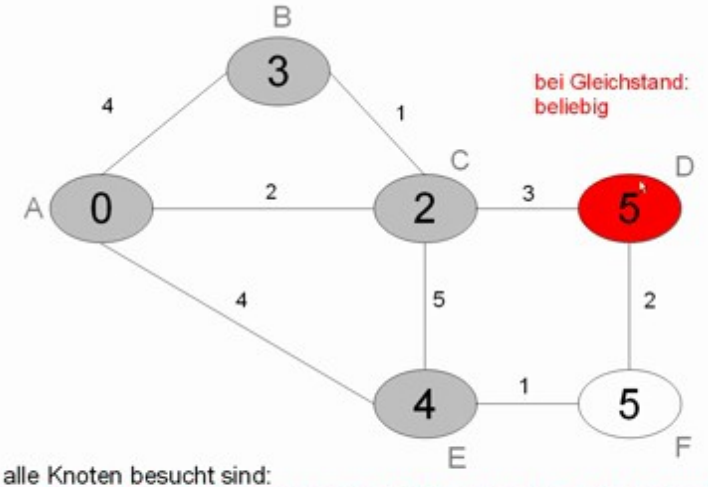
Wiederhole, bis alle Knoten besucht sind:
- setze den unbesuchten **Knoten** mit der geringsten Distanz als aktuell und besucht
- für alle unbesuchten **Nachbarn**: addiere eigene Distanz und das Kantengewicht
  - wenn Summe geringer ist als deren aktuelle Distanz,
    - dann setze sie
    - und setze dich als seinen Vorgänger

A	0	A
B	3	C
C	2	A
D	5	C
E	4	A
F	5	E



Wiederhole, bis alle Knoten besucht sind:
- setze den unbesuchten **Knoten** mit der geringsten Distanz als aktuell und besucht
- für alle unbesuchten **Nachbarn**: addiere eigene Distanz und das Kantengewicht
  - wenn Summe geringer ist als deren aktuelle Distanz,
    - dann setze sie
    - und setze dich als seinen Vorgänger

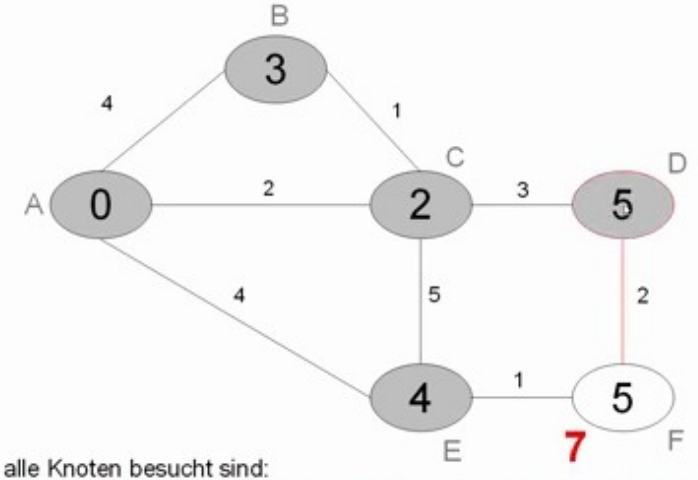
A	0	A
B	3	C
C	2	A
D	5	C
E	4	A
F	5	E



bei Gleichstand:
beliebig

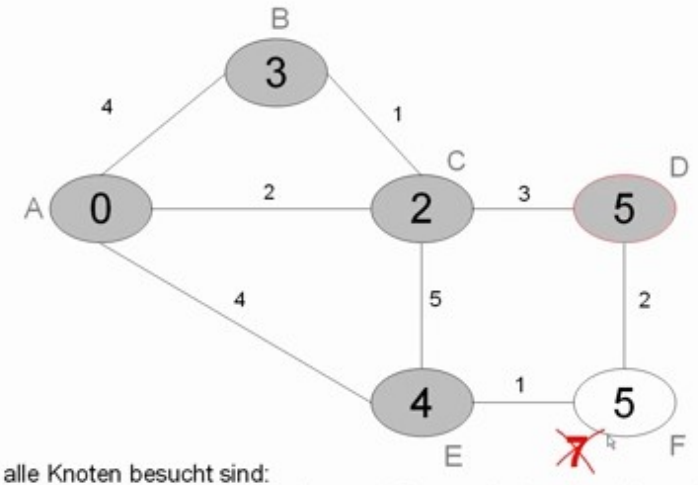
Wiederhole, bis alle Knoten besucht sind:
- setze den unbesuchten **Knoten** mit der geringsten Distanz als aktuell und besucht
- für alle unbesuchten **Nachbarn**: addiere eigene Distanz und das Kantengewicht
  - wenn Summe geringer ist als deren aktuelle Distanz,
    - dann setze sie
    - und setze dich als seinen Vorgänger

A	0	A
B	3	C
C	2	A
D	5	C
E	4	A
F	5	E



Wiederhole, bis alle Knoten besucht sind:
- setze den unbesuchten **Knoten** mit der geringsten Distanz als aktuell und besucht
- für alle unbesuchten **Nachbarn**: addiere eigene Distanz und das Kantengewicht
  - wenn Summe geringer ist als deren aktuelle Distanz,
    - dann setze sie
    - und setze dich als seinen Vorgänger

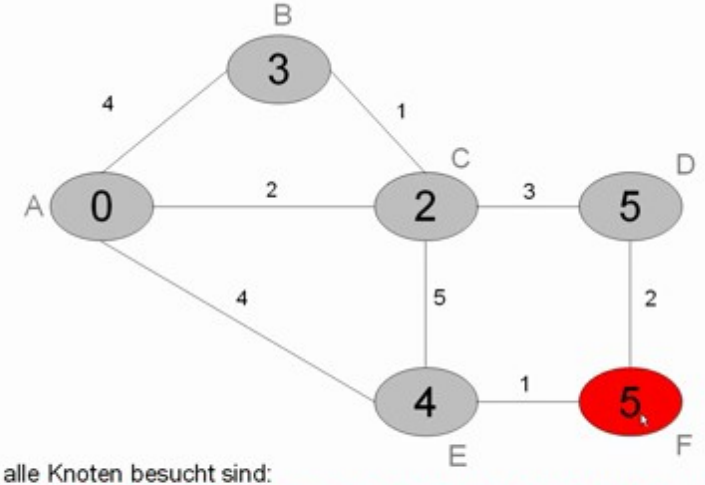
A	0	A
B	3	C
C	2	A
D	5	C
E	4	A
F	5	E



Wiederhole, bis alle Knoten besucht sind:
- setze den unbesuchten **Knoten** mit der geringsten Distanz als aktuell und besucht
- für alle unbesuchten **Nachbarn**: addiere eigene Distanz und das Kantengewicht
  - wenn Summe geringer ist als deren aktuelle Distanz,
    - dann setze sie
    - und setze dich als seinen Vorgänger

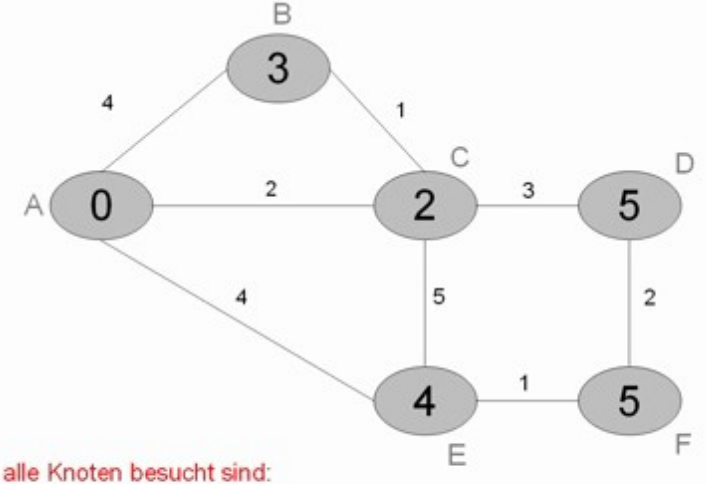
Fertig: Kürzeste Wege von A zu allen anderen Knoten gefunden.

A	0	A
B	3	C
C	2	A
D	5	C
E	4	A
F	5	E



Wiederhole, bis alle Knoten besucht sind:
- setze den unbesuchten **Knoten** mit der geringsten Distanz als aktuell und besucht
- für alle unbesuchten **Nachbarn**: addiere eigene Distanz und das Kantengewicht
  - wenn Summe geringer ist als deren aktuelle Distanz,
    - dann setze sie
    - und setze dich als seinen Vorgänger

A	0	A
B	3	C
C	2	A
D	5	C
E	4	A
F	5	E



Wiederhole, bis alle Knoten besucht sind:
- setze den unbesuchten **Knoten** mit der geringsten Distanz als aktuell und besucht
- für alle unbesuchten **Nachbarn**: addiere eigene Distanz und das Kantengewicht
  - wenn Summe geringer ist als deren aktuelle Distanz,
    - dann setze sie
    - und setze dich als seinen Vorgänger