„Graph“

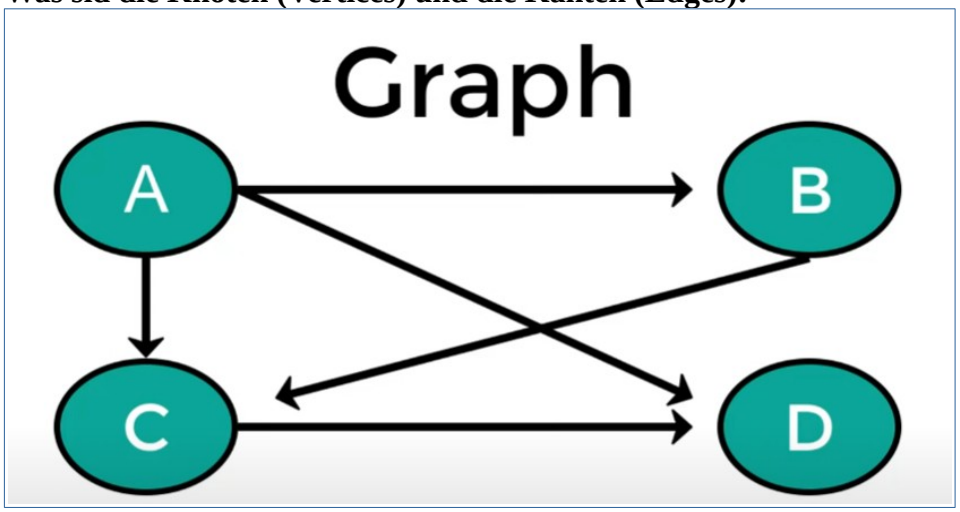
was ist das?

- ist eine abstrakte Datenstruktur, die aus **Knoten** (auch **Vertices** genannt) und **Kanten (Edges)** besteht.

Verwendung:

- Ein Graph wird verwendet, um Beziehungen zwischen Objekten darzustellen, und ist in vielen Bereichen wie Netzwerken, sozialen Medien, Routenplanung usw. relevant

Was sid die Knoten (Vertices) und die Kanten (Edges):



V (Vertices) = {A,B,C,D}
E (Edges) = {(A,B),(A,D),(A,C),(B,C),(C,D)}

1. Arten von Graphen (Types of Graphs):

- **Gerichteter Graph (Directed Graph)** : Ein Graph, in dem die Kanten eine Richtung haben. Eine Kante von einem Knoten zu einem anderen kann nur in eine Richtung durchlaufen werden.

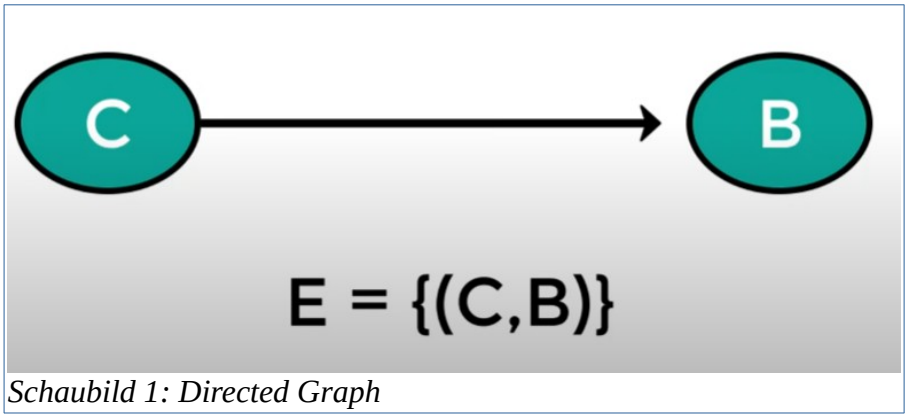


Schaubild 1: Directed Graph

- **Ungerichteter Graph (Undirected Graph)**: Ein Graph, in dem die Kanten keine Richtung haben. Eine Kante zwischen zwei Knoten (Vertices) kann in beide Richtungen durchlaufen werden.

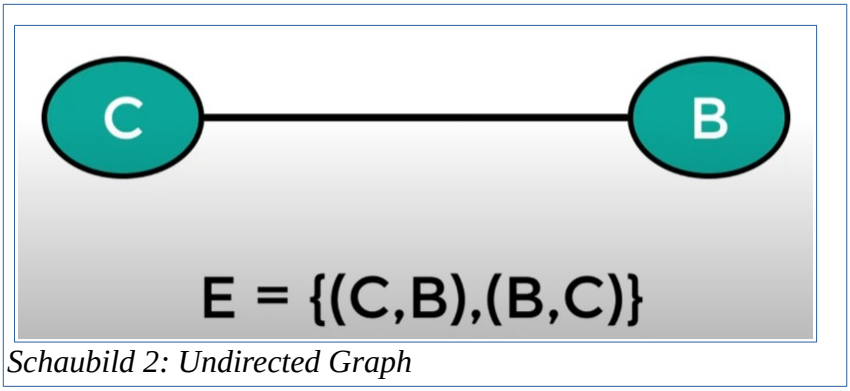
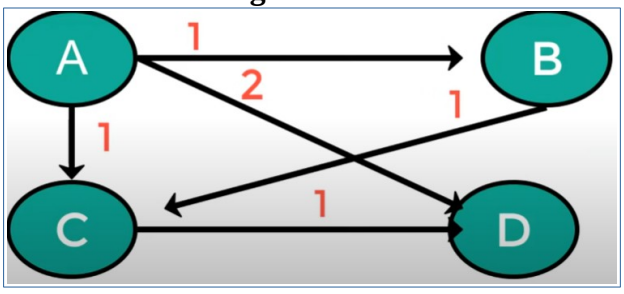
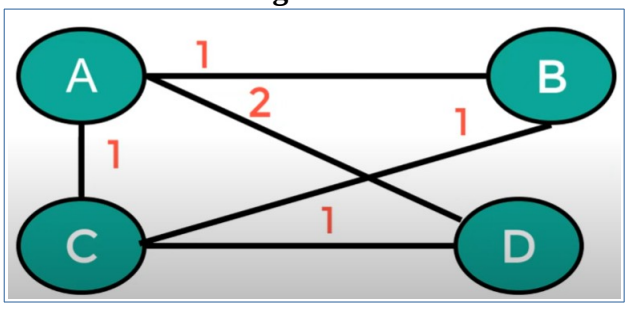


Schaubild 2: Undirected Graph

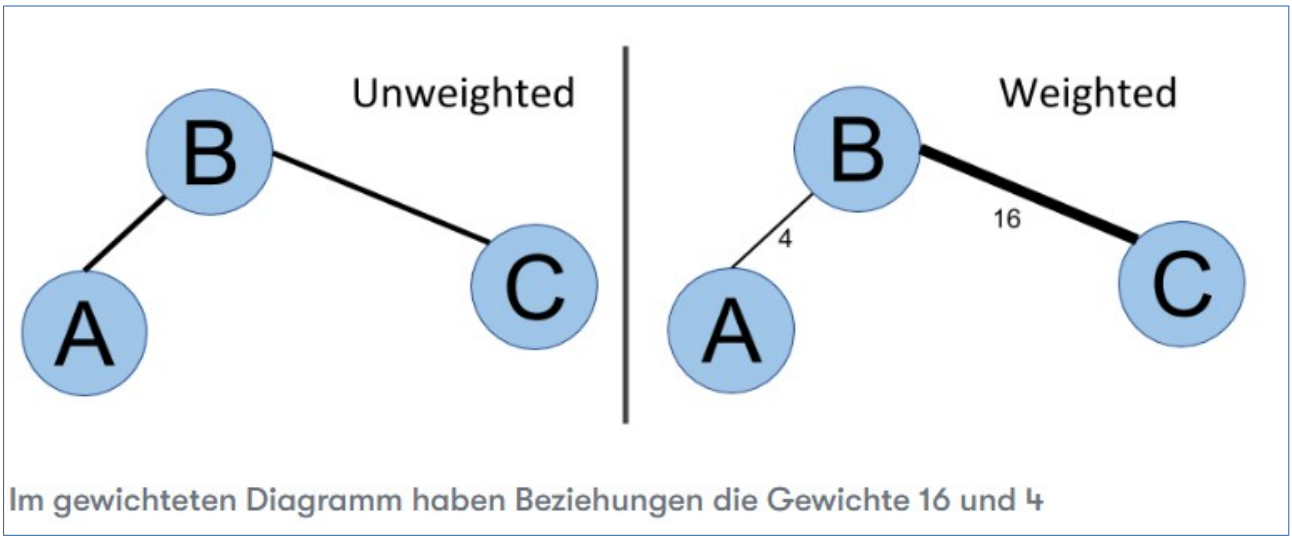
- **Gewichteter Graph (Weighted Graph)**: Ein Graph, in dem den Kanten Gewichte zugeordnet sind. Diese Gewichte können Entfernungen, Kosten, Zeiten usw. darstellen.
  - **Directed Weighted:**



  - **Undirected Weighted:**

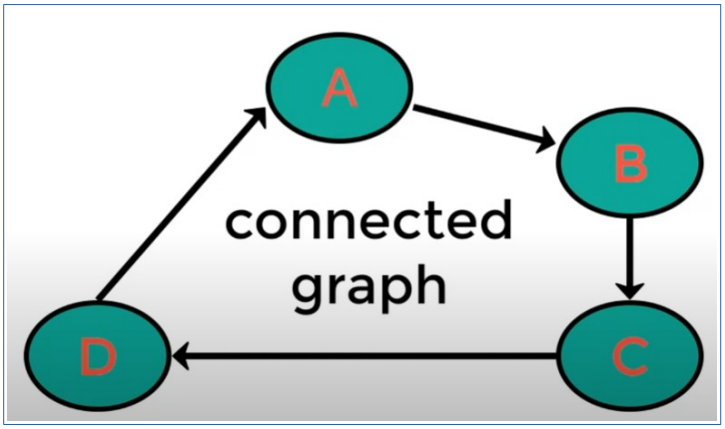


- **Gewichtet versus ungewichtet**

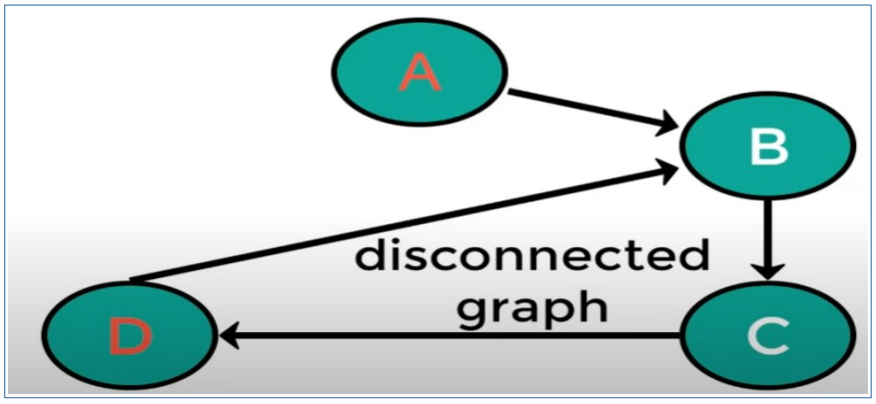


Gewichtet = Linien mit Zahlen
ungewichtet = Linien ohne Zahlen

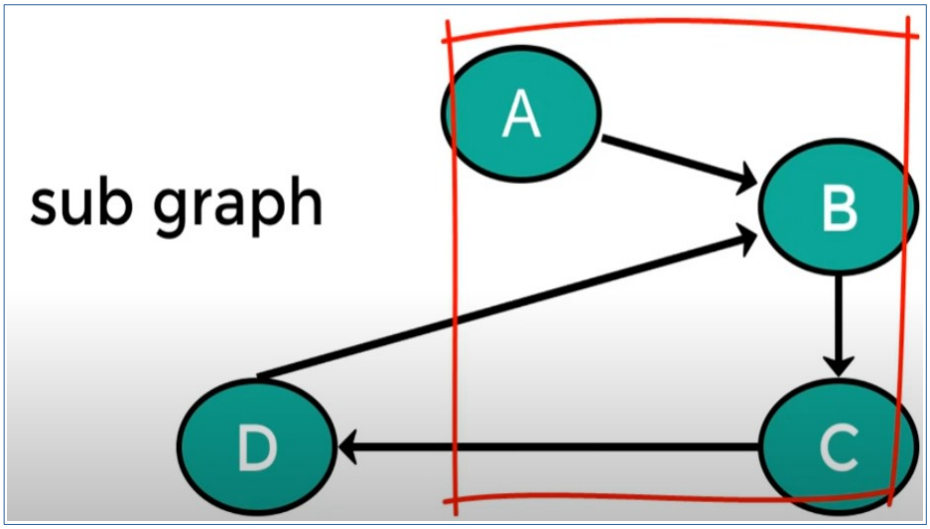
- **Zusammenhängender Graph (Connected Graph):** Ein ungerichteter Graph, in dem es für jedes Paar von Knoten einen Pfad gibt. Das bedeutet, dass jeder Knoten durch Kanten erreichbar ist.



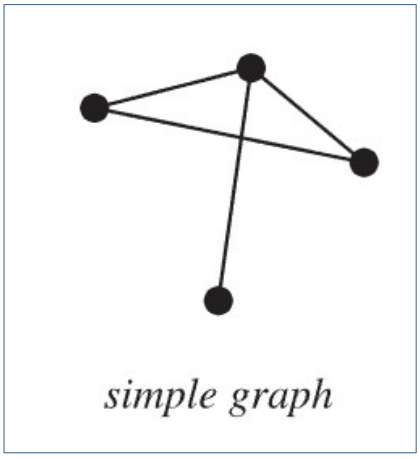
- **Unzusammenhängender Graph (Disconnected Graph)**: Ein ungerichteter Graph, in dem nicht alle Knoten verbunden sind. Es gibt also mindestens zwei Knotenpaare, die keinen Pfad zwischen sich haben.



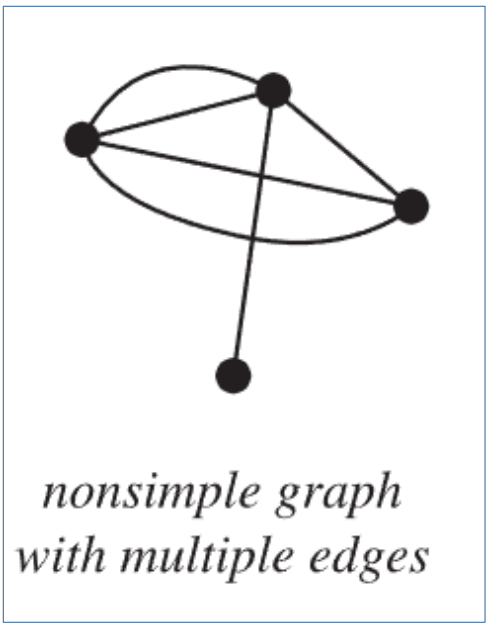
- **Teilgraph (Subgraph):** ist ein Graph, der aus einer Teilmenge der Knoten (Vertices) und Kanten (Edges) eines größeren Graphen gebildet wird.

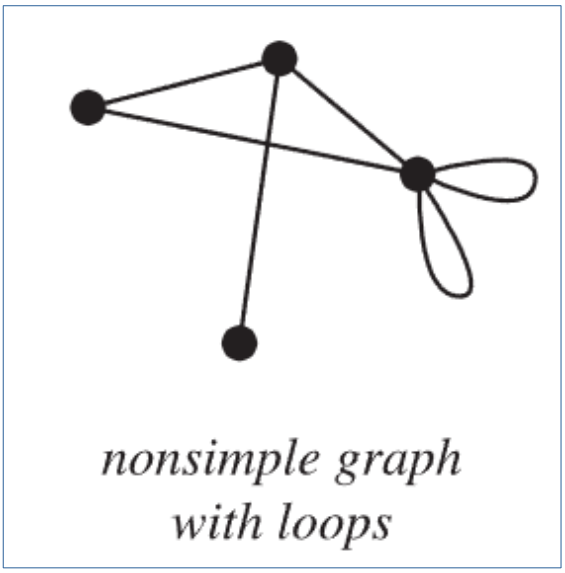


- **Simple Graph (Einfacher Graph):** Ein einfacher Graph (Simple Graph) ist ein Graph, der die folgenden Eigenschaften hat:
  - **Keine Schleifen (Loops)**: Es gibt keine Kanten, die von einem Knoten zu sich selbst führen.
  - **Keine mehrfachen Kanten (Multiple Edges)**: Es gibt zwischen jedem Paar von Knoten höchstens eine Kante.

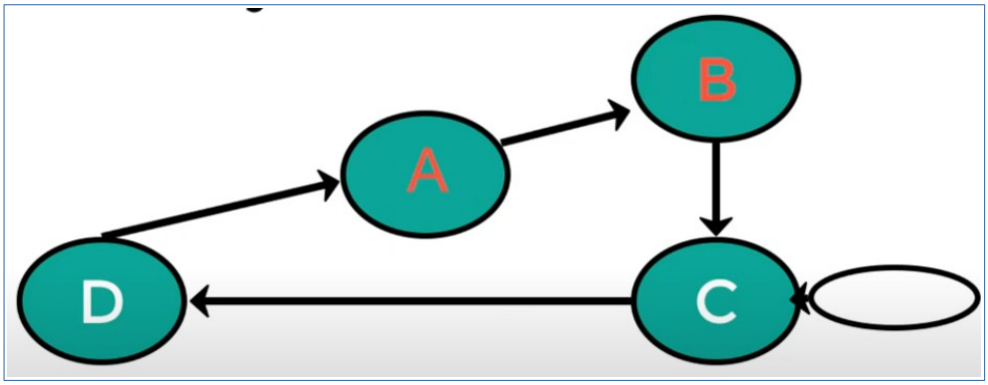


- **Not Simple Graph (Nicht einfacher Graph):** Ein nicht einfacher Graph (Not Simple Graph) ist ein Graph, der mindestens eine der folgenden Eigenschaften aufweist:
  - **Schleifen (Loops)**: Es gibt Kanten, die von einem Knoten zu sich selbst führen.
    - **Mehrfache Kanten (Multiple Edges)**: Es gibt zwischen einem Paar von Knoten mehr als eine Kante.





- **Adjazente Knoten (Adjacent Vertices):** gibt an, diese Knote mit welchem weitere Knoten verbunden ist!



Beispiele:

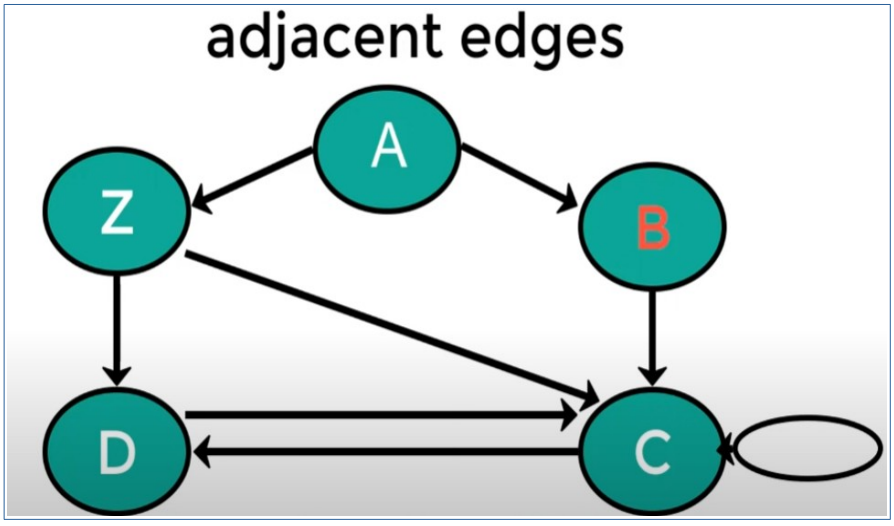
Adjazente Knote (A): {B},Warum?

- Weil (A) zeigt nur auf (B)

Adjazente Knote (C): {D},{C}Warum?

- Weil (C) zeigt auf (B) und auf sich selbst(Loop)

- **Adjazente Kanten (Adjacent Edges):** zwei Knoten und dann eingeben, wo ich weiter laufen kann (Siehe Beispiel)



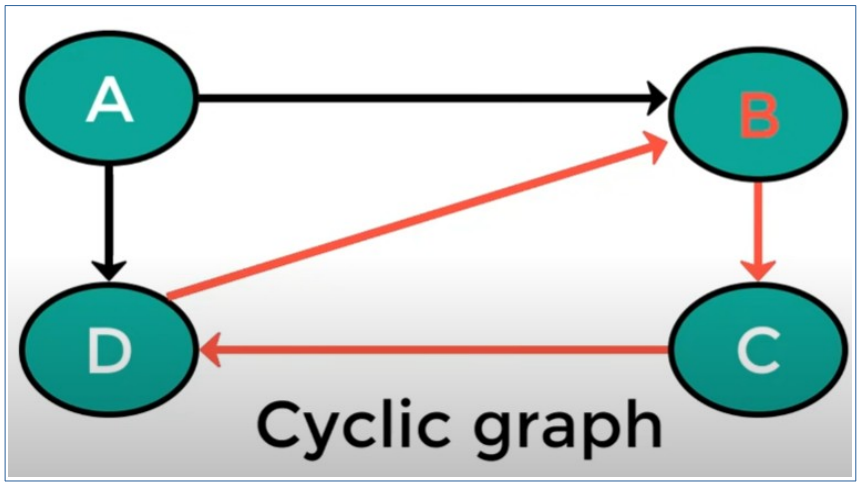
Beispiel:

Adjazente Kanten(A,B): {(A,Z),(B,C)}

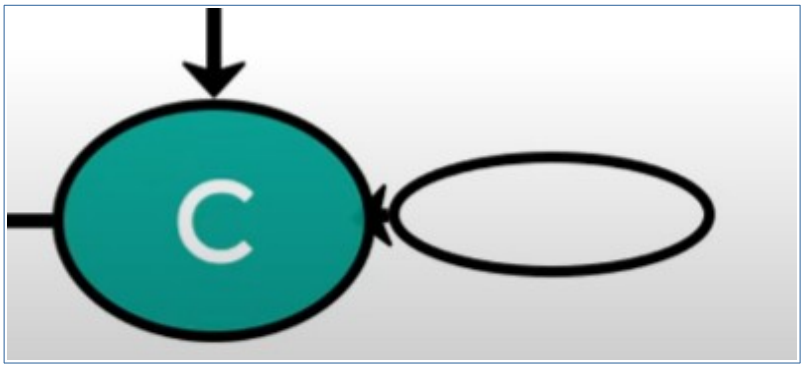
Adjazente Kanten(B,C): {(C,D),(C,C)}

- **Zyklischer Graph (Cyclic Graph)**: Ein Zyklus ist eine Schleife. Ein Graph ist zyklisch, wenn Sie mindestens einen Pfad finden können, der von einem Knoten ausgeht und zu genau diesem Knoten zurückführt.

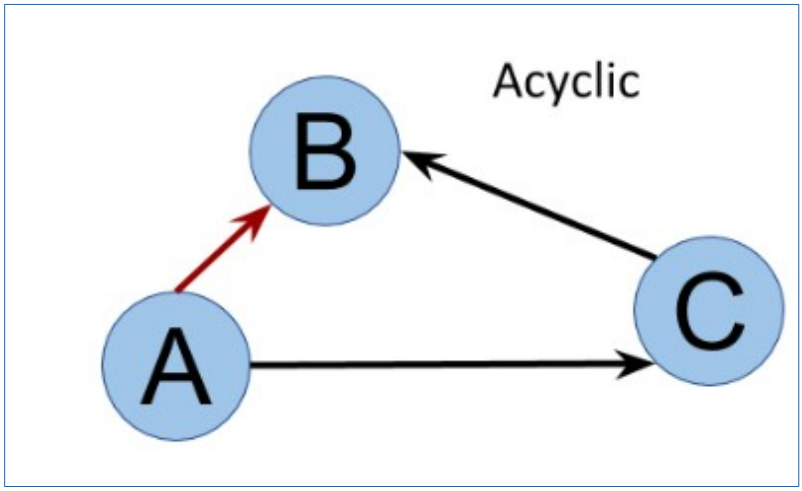
Beispiel: obwohl ich aus B raus gegangen bin, konnte ich trotzdem zurück kommen (C → D → B)



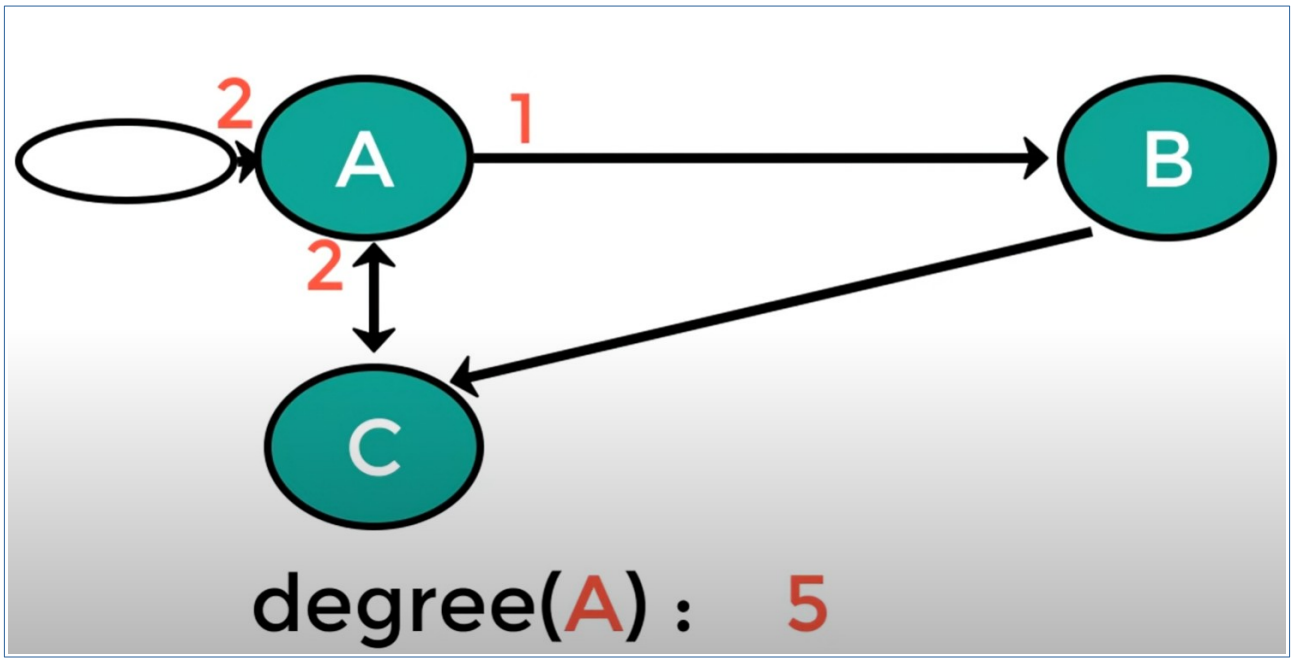
Loop ist präsentiert kein Zyklischer Graph (Cyclic Graph)!



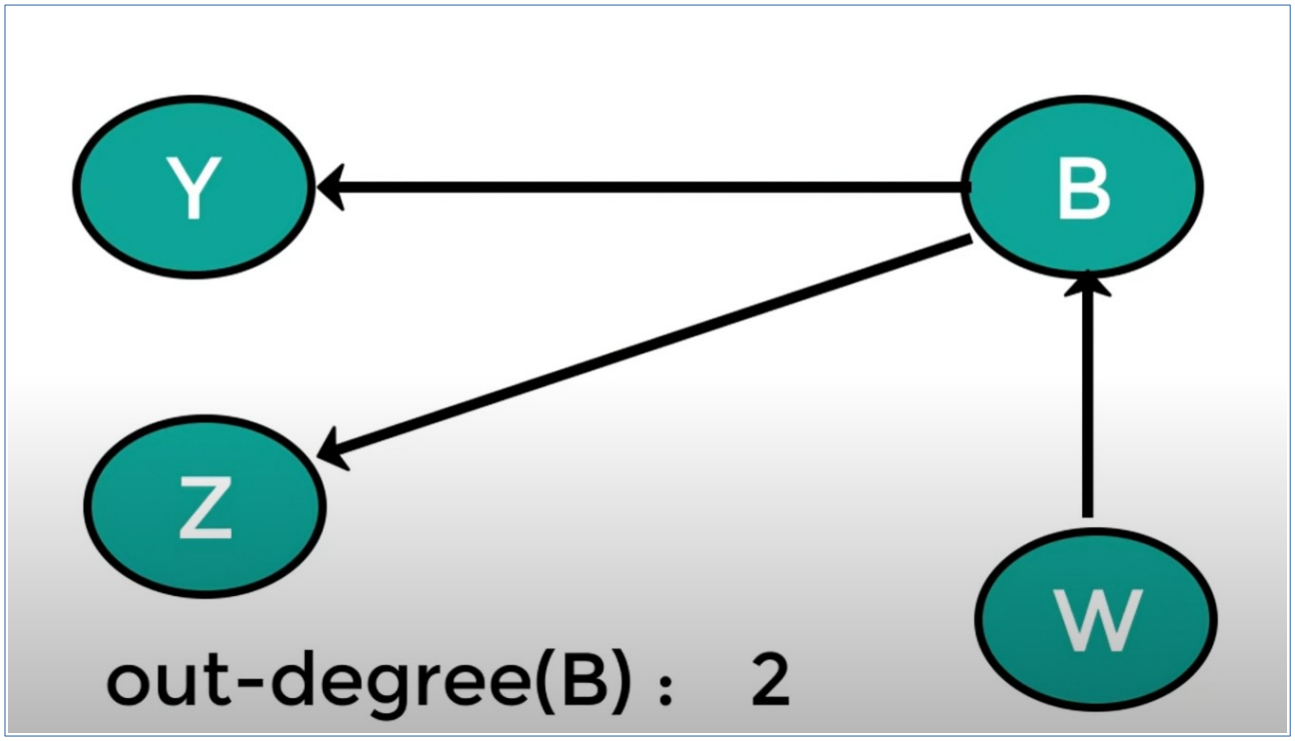
- **Azyklischer Graph (Acyclic Graph)**: wenn ich von (A) raus bin, kann ich nicht zu (A) zurückführt.



Grad eines Knotens (Degree of a Vertex): Die Anzahl der Kanten, die mit einem Knoten verbunden sind. In gerichteten Graphen unterscheidet man zwischen Eingangsgrad (In-degree) und Ausgangsgrad (Out-degree).



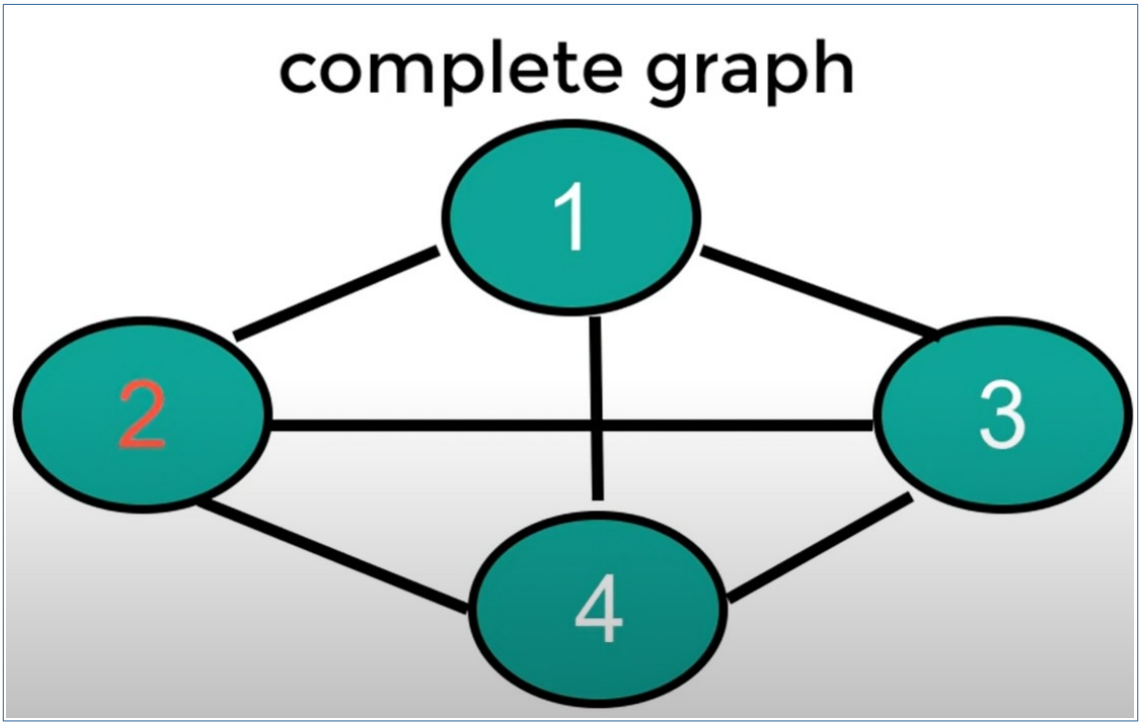
Eingangsgrad (In-degree) und Ausgangsgrad (Out-degree):



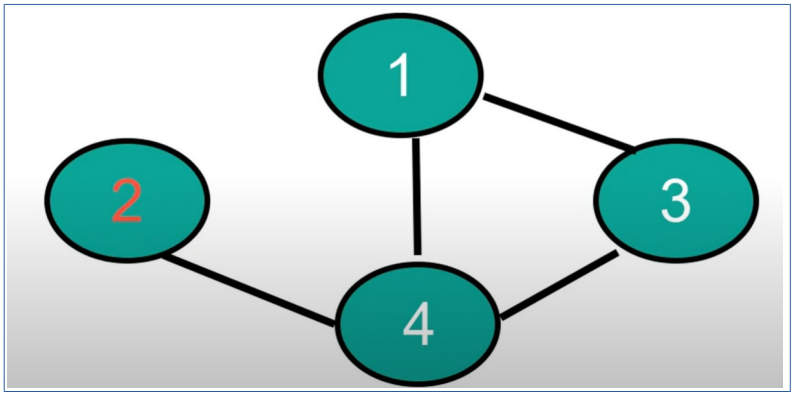
In-degree(B) = 2

Out-degree(B) = 1

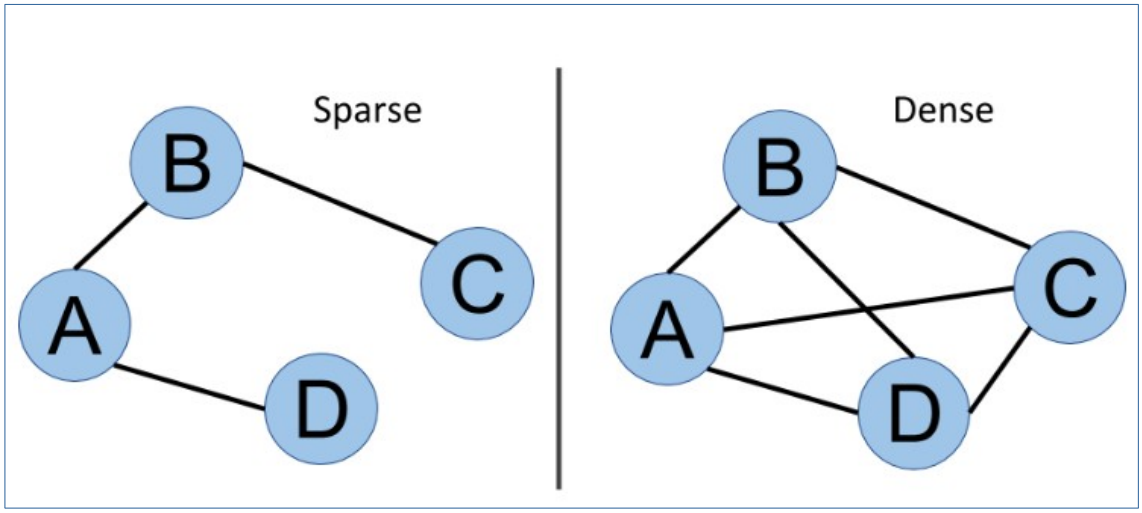
- **vollständiger Graph (Complete Graph):** jeder Knoten (Vertex) mit jedem anderen Knoten direkt durch eine Kante verbunden ist



nicht vollständiger Graph (Complete Graph):



- **Dicht versus spärlich (Sparse and Dense):**



E fast= V²: das Graph ist dense

E fast= V: das Graph is sparse

wie wird Graph repräsentiert? Darstellung von Graphen (Representation of Graphs)

- **Adjazenzliste (Adjacency List)**: Eine Datenstruktur, die für jeden Knoten eine Liste von Nachbarn speichert. Es ist eine effiziente Darstellung für Graphen mit wenigen Kanten.
- **Adjazenzmatrix (Adjacency Matrix)**: Eine 2D-Matrix, bei der der Eintrag in der Zelle (i, j) angibt, ob eine Kante zwischen Knoten i und j existiert und eventuell das Gewicht dieser Kante. Diese Darstellung ist effizient für dichte Graphen.

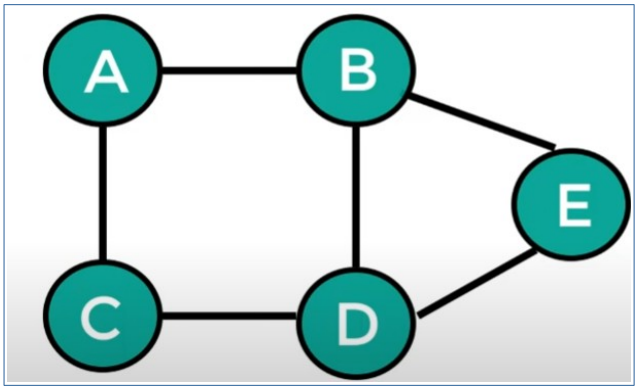
Darstellung 1:

1. Adjazenzmatrix (Adjacency Matrix): Space complexity = **$O(V^2)$**

1.1 Bestimme den Size of matrix:

Anzahl Knoten = 5

matrix = 5 * 5



1 = wenn eine Knote mit einer anderen Knote verbunden ist

0 = Wenn es keine Verbindung gibt

	A	B	C	D	E
A	0	1	1	0	0
B	1	0	0	1	1
C	1	0	0	1	0
D	0	1	1	0	1
E	0	1	0	1	0

Darstellung 2:

2. Adjazenzliste (Adjacency List): Space Complexity = O(V+ E)

2.1 erstelle für jede Knote ein LinkedList,ArrayList etc..