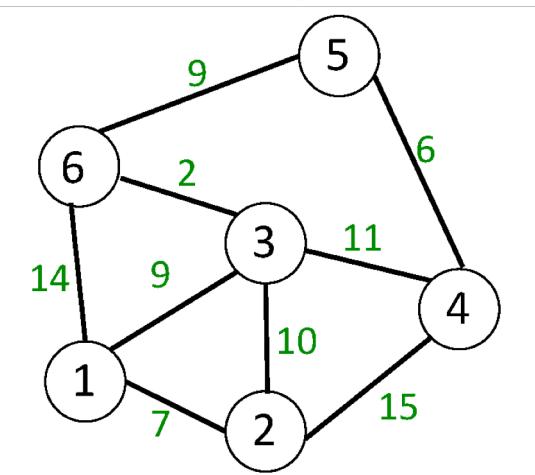


Veri Yapıları ve Algoritmalar

Graflar

Graf

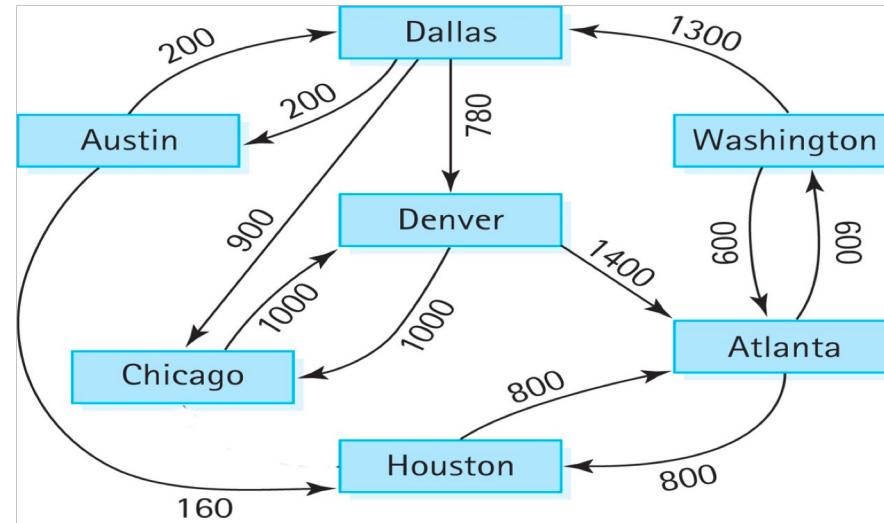
- Graf düğümler (node/vertex) ve düğümler arasındaki ilişkileri gösteren kenarlardan (edge) oluşan veri yapısıdır.
 - G: Graf
 - V: Vertex
 - E: Edge



https://cdn-images-1.medium.com/max/1600/1*3tLY8VADFc5Cr71aEnZ5mg.png

Graf

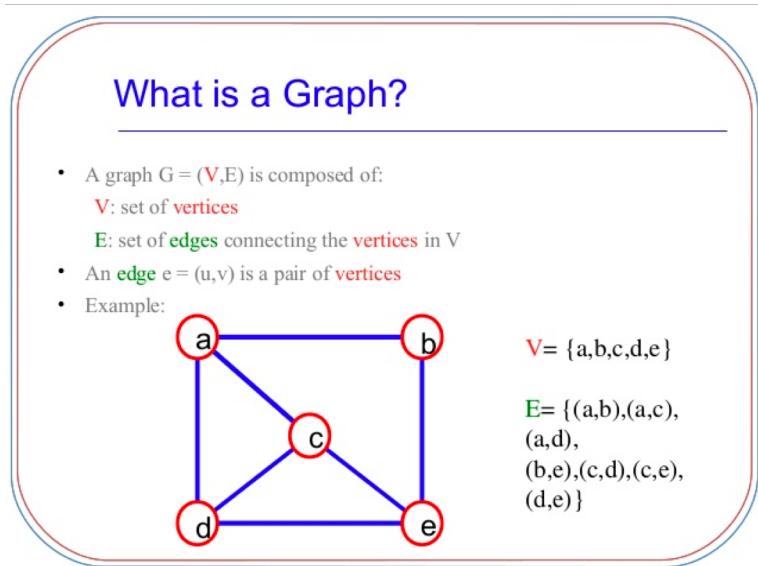
- Nerelerde kullanılır.
 - Elektronik devreler.
 - Baskı devreler
- Bilgisayar ağları.
 - Local/Wide, internet
- Navigasyon sistemleri.



<http://cs.middlesexcc.edu/~schatz/csc236/handouts/graph.weighted.directed.jpg>

Graf

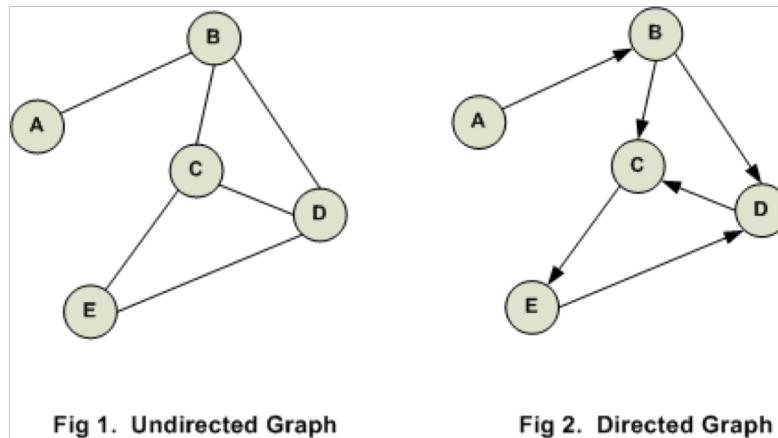
- İki düğüm ortak bir kenarla ilişkiliyse komşudurlar (**adjacent, neighbor**)



<https://image.slidesharecdn.com/graphs1-130126021226-phpapp01/95/graphs-in-data-structure-2-638.jpg?cb=1359166826>

Graf

- Yönlendirilmiş Graf (Directed Graphs)
 - İki düğüm arasında yön bilgisi vardır.
- Yönlendirilmemiş Graf (Undirected Graphs)
 - Hiçbir yönlendirme bilgisi içermez.

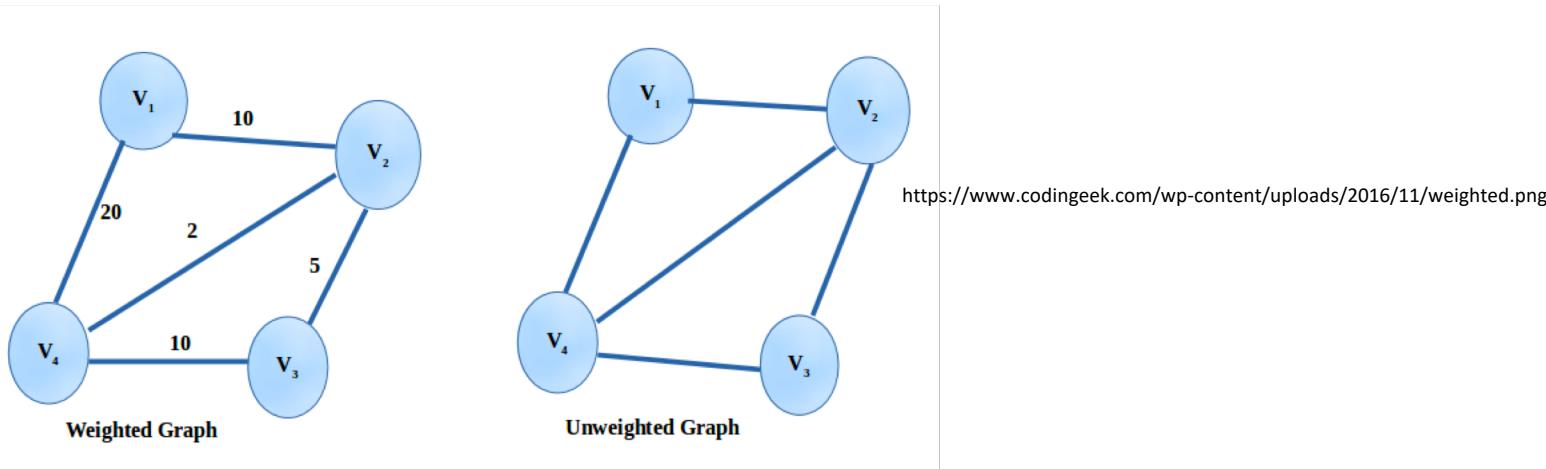


Graf

- Yönlendirilmiş kenar (Directed Edge)
 - $(a,b) \Rightarrow a$ orjin, b hedeftir.
 - (a,b) ile (b,a) aynı değildir.
- Yönlendirilmemiş kenar (Undirected Edge)
 - $(a,b) \Leftrightarrow (b,a)$
 - Yön bilgisi olmadığı için

Graf

- Ağırılıklı Graf (Weighted Graphs)
 - Kenarlar üzerinde ağırlık bilgisi vardır.
 - *Navigasyon uygulamasında iki şehir arasındaki mesafe bilgisi*
- Ağırılıksız Graf (Unweighted Graphs)
 - Kenarlar üzerinde ağırlık bilgisi yoktur.



Graf

- Yol (path): Düğümlerin sırasıdır.
- Döngü (cycle): Kapalı yoldur.

Graph Theory

Path and Cycle

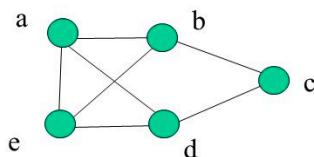
□ **Path:** a sequence of **distinct** vertices such that two consecutive vertices are adjacent.

- Example: (a, d, c, b, e) is a path
- (a, b, e, d, c, b, e, d) is not a path; it is a walk.

https://images.slideplayer.com/29/9489856/slides/slide_17.jpg

□ **Cycle:** a closed Path

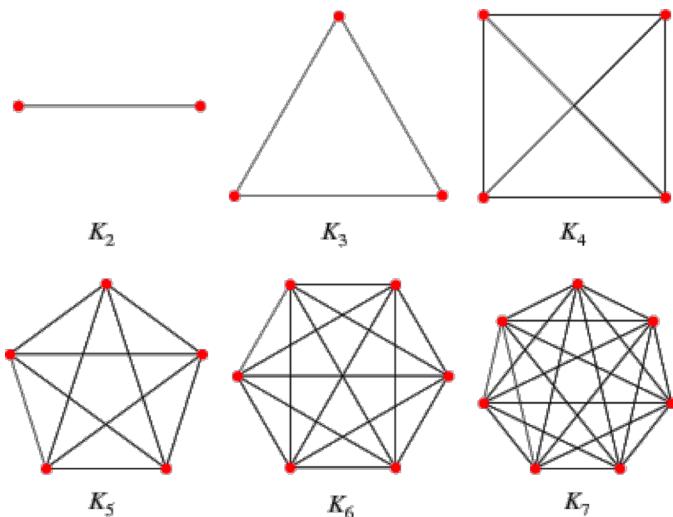
- Example: (a, d, c, b, e, a) is a cycle



17

Graf

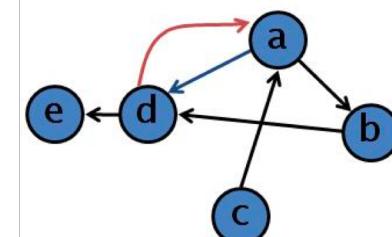
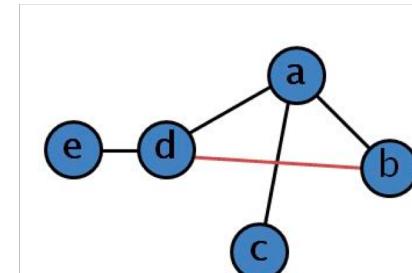
- Tam Graf (Complete Graph): Eğer her düğüm, diğer tüm düğümlerle kenar içeriyorsa tam grafdır.
- Yönlendirilmiş tam grafta n düğüm var ise $n*(n-1)$ kenar vardır.
- Yönlendirilmemiş tam grafta n düğüm var ise $n*(n-1)/2$ kenar vardır.



http://mathworld.wolfram.com/images/eps-gif/CompleteGraphs_801.gif

Graf

- Komşuluk matrisi (Adjacency Matrice): Düğümler arasındaki bağlantıyı gösteren bir kare matristir.
- Yönlendirilmiş bir graf ise
 - satırlar orjin, sütunlar hedeftir.
 - satırdan-> sütuna 1 olmalıdır.
- Yönlendirilmemiş bir graf ise
 - Satır ve sütunlar komşuluğu gösterir.
 - Komşuluk değerleri 1 olmalıdır.



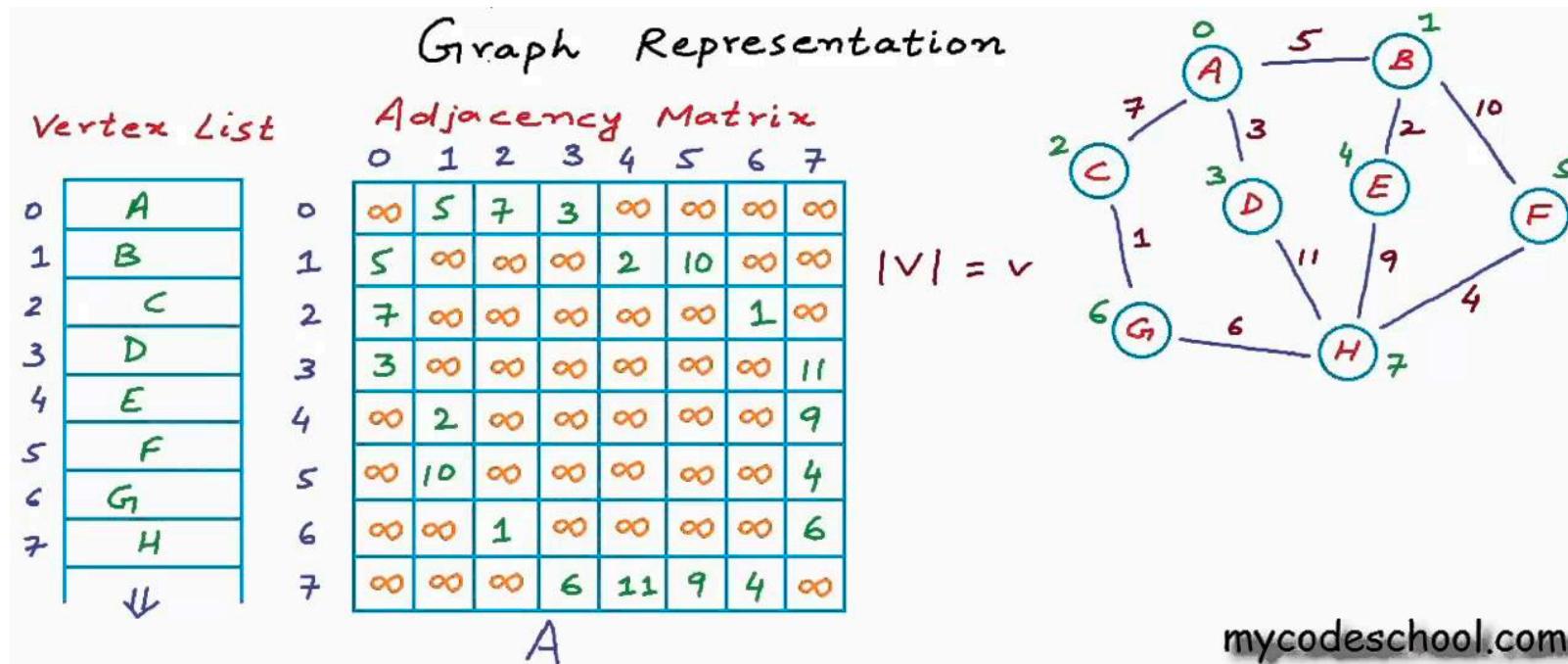
	a	b	c	d	e
a	0	1	1	1	0
b	1	0	0	1	0
c	1	0	0	0	0
d	1	1	0	0	1
e	0	0	0	1	0

	a	b	c	d	e
a	0	1	0	1	0
b	0	0	0	1	0
c	1	0	0	0	0
d	1	0	0	0	1
e	0	0	0	0	0

https://bournetocode.com/projects/AQA_A_Theory/pages/img/adjacency-matrix-of-graph.jpg

Graf

- Ağırlıklı graflarda komşuluk matrisi



<https://i.ytimg.com/vi/9C2cpQZVRBA/maxresdefault.jpg>

Graf

- Graf Üzerinde Dolaşma
- DFS (**Depth First Search**)
- BFS (**Breadth First Search**)

Graf

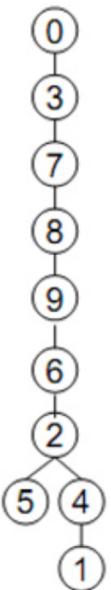
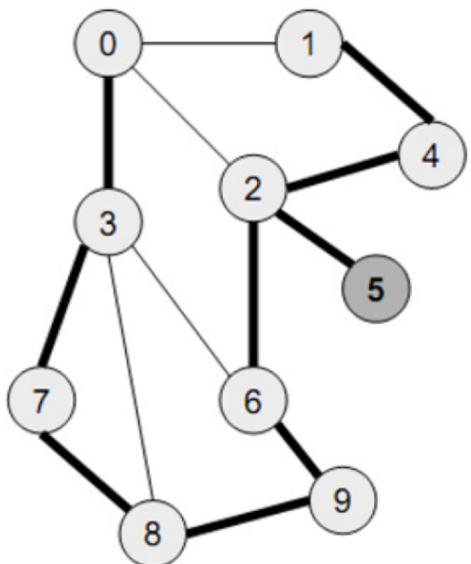
- **Depth First Search**

1. Başlangıç düğümünü seç ve ziyaret et.
2. Seçilen düğümün komşusunu ziyaret et.
3. Ziyaret edilecek komşu kalmayana kadar 2.adımı tekrar et.
4. Komşu kalmadığında geri dön ve 2-3.adımları tekrar et.

Graf

- DFS

- Depth first arama



<http://www.barkankisa.com/algoritmalar/dsf-derin-oncelikli-arama-algoritmasi-dept-first-search-algorithm/>

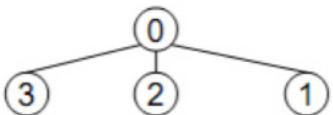
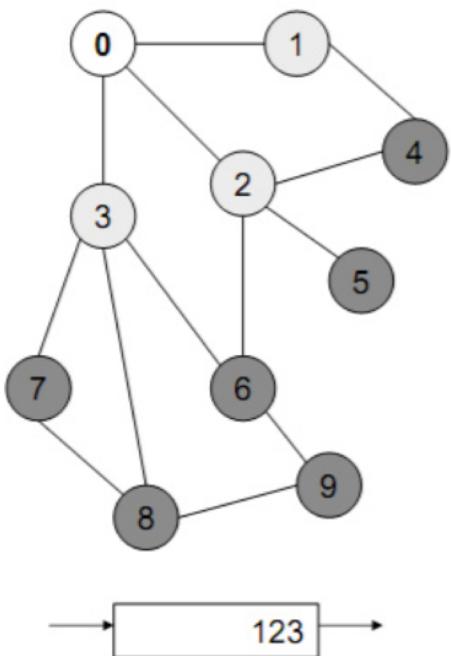
Graf

- Breadth First Search
 - 1. Seçilen nodun tüm komşuları sırayla ziyaret edilir.
 - 2. Her komşu kuyruk içine atılır.
 - 3. Komşu kalmazsa kuyruktaki ilk düğüm alınır ve 2.adım tekrarlanır.

Graf

- BFS

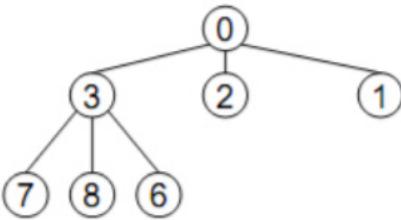
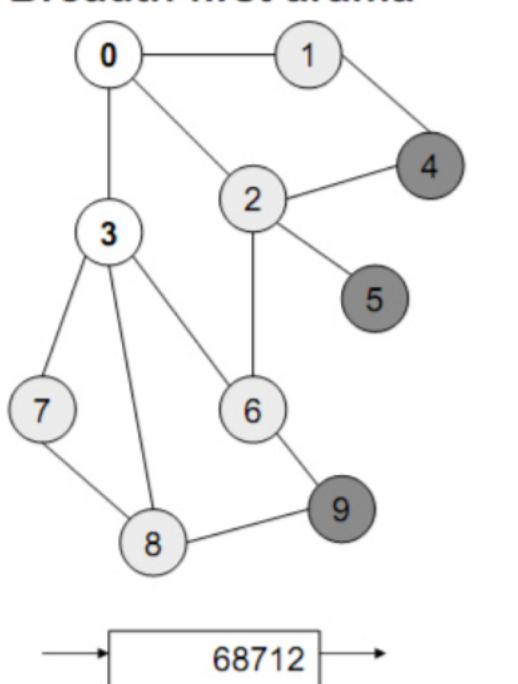
- Breadth first arama



Graf

- BFS

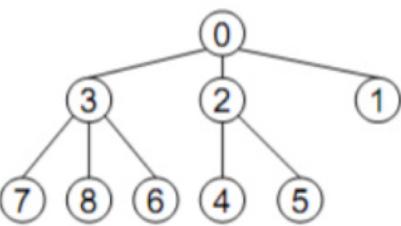
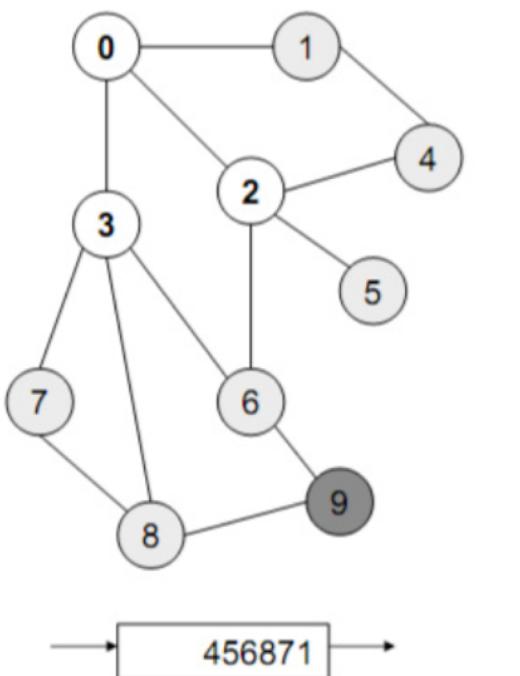
- Breadth first arama



Graf

- BFS

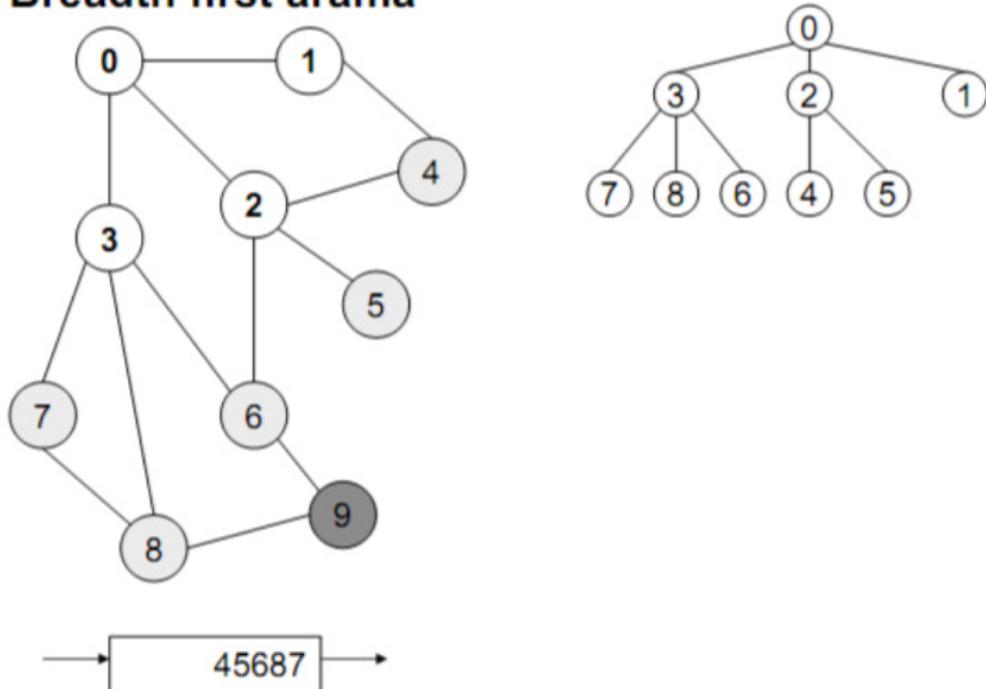
- Breadth first arama



Graf

- BFS

- Breadth first arama



Graf

- BFS

- Breadth first arama

