CENG 201 Veri Yapıları 9: Çizge(Graph)

Öğr.Gör. Şevket Umut ÇAKIR

Pamukkale Üniversitesi

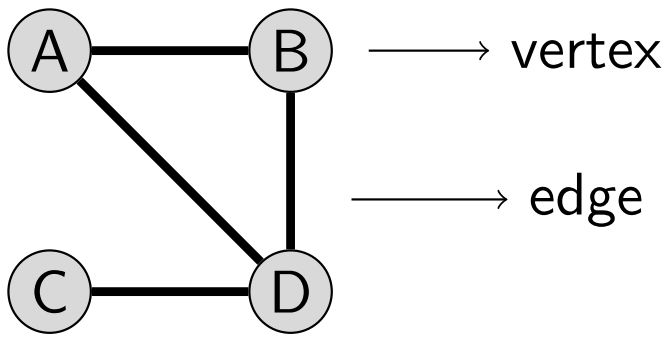
Hafta 9

Anahat

Çizge Tanımı

Tanım

Çizge(Graph) veri yapısı düğüm(vertex) ve kenarlardan(edge) oluşan bir yapıdır. $G = (V, E)$ Kenarlar iki düğüm arasındaki(v, w) yollardır.
$v, w \in V$



Graph Türleri

Tanım

Çizgenin kenarları sıralı ise ya da bir düğümden başka bir düğüme doğru ise buna **yönlü çizge(directed graph)** aksi takdirde **yönsüz çizge(undirected graph)** adı verilir.

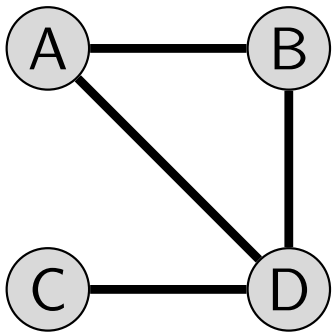


Figure: Yönsüz Çizge

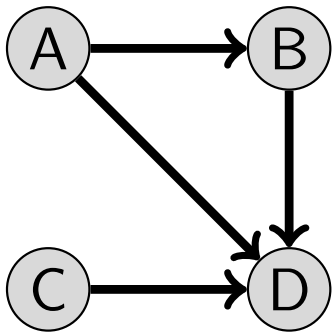


Figure: Yönlü Çizge

En çok kenar sayısı

- Yönsüz çizge için:

Graph Türleri

Tanım

Çizgenin kenarları sıralı ise ya da bir düğümden başka bir düğüme doğru ise buna **yönlü çizge(directed graph)** aksi takdirde **yönsüz çizge(undirected graph)** adı verilir.

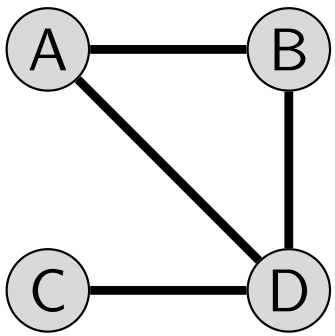


Figure: Yönsüz Çizge

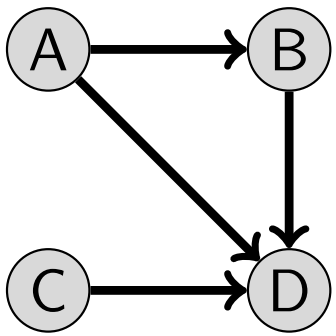


Figure: Yönlü Çizge

En çok kenar sayısı

- Yönsüz çizge için: $\frac{n(n-1)}{2}$
- Yönlü çizge için:

Graph Türleri

Tanım

Çizgenin kenarları sıralı ise ya da bir düğümden başka bir düğüme doğru ise buna **yönlü çizge(directed graph)** aksi takdirde **yönsüz çizge(undirected graph)** adı verilir.

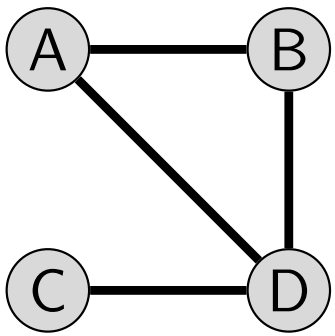


Figure: Yönsüz Çizge

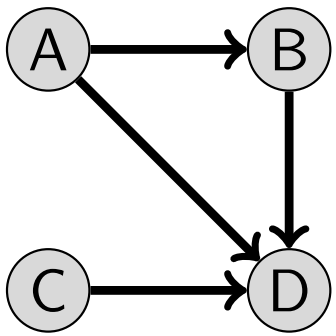


Figure: Yönlü Çizge

En çok kenar sayısı

- Yönsüz çizge için: $\frac{n(n-1)}{2}$
- Yönlü çizge için: $n(n-1)$

Düğüm Derecesi/Vertex Degree

Yönsüz çizgeler için her bir düğümün kenar sayısıdır.

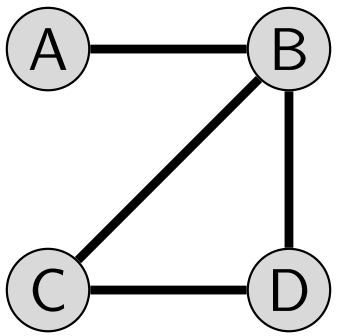


Table: Yönsüz Çizge için Düğüm Dereceleri

	A	B	C	D
Düğüm Derecesi	1	3	2	2

Düğüm Derecesi/Vertex Degree

Yönlü çizgeler için her bir düğüme gelen(Indegree) ve düğümden çıkan(Outdegree) kenar sayısıdır.

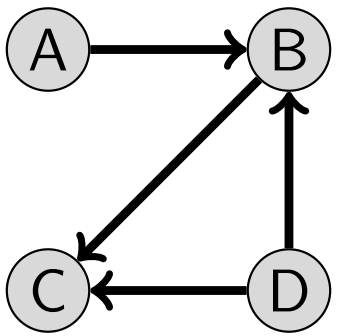


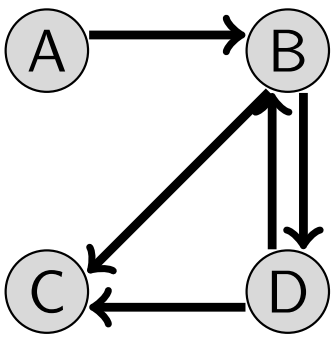
Table: Yönlü Çizge için Düğüm Dereceleri

	A	B	C	D
Indegree	0	2	2	0
Outdegree	1	1	0	2

Yol/Path

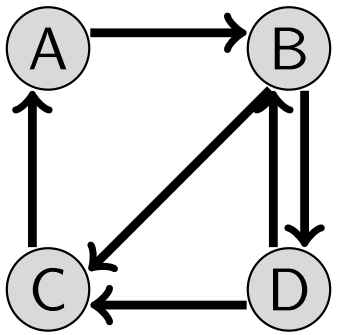
Tanım

Bir düğümden başka bir düğüme ulaşılabiliyorsa, bu iki düğüm arasında **yol(path)** vardır. $A, X_1, X_2, \cdots, X_n, C$ kenarları mevcutsa A'dan C'ye yol vardır.



- A, B, C
- A, B, D, C
- A, B, D, B, C

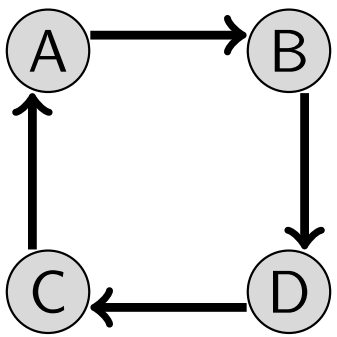
Yol/Path



- **Basit Yol(Simple Path):** Aradaki düğümlere sadece 1 defa uğranıyorsa basit yol vardır
  - A, B, C
  - A, B, D, C
  - A, B, C, A
  - A, B, D, C, A
- Aksi takdirde yol basit değildir
  - A, B, D, B, C

Döngü, Çevrim/Cycle

Bir yol başladığı yerde bitiyorsa çizgede döngü(cycle) vardır.



Alt Çizge/Subgraph

Tanım

Bir çizgenin hernagi bir parçasına **alt çizge(subgraph)** denir.

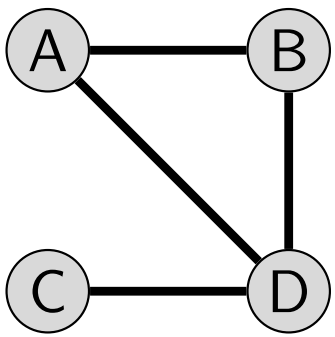


Figure: Örnek Çizge

Figure: Alt Çizge 1

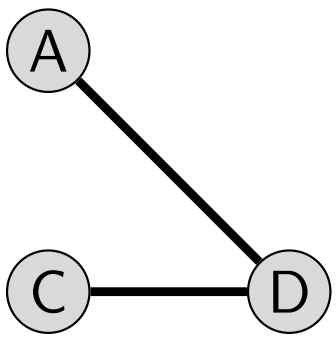


Figure: Alt Çizge 2

Bağlı Çizgeler/Connected Graphs

Tanım

V_i ile V_j düğümleri arasında yol varsa bu iki düğüm **bağlıdır(connected)**. Herhangi iki düğüm arasında bir yol varsa bu çizgeye **bağlı çizge (connected graph)** adı verilir.

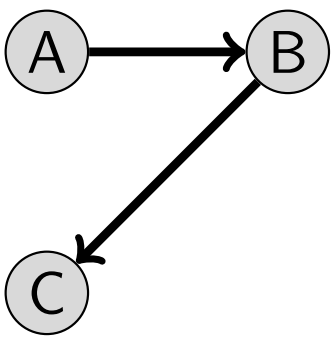


Figure: Bağlı Olmayan Çizge

- A, B
- B, C
- ~~B, A~~
- ~~C, A~~
- ~~C, B~~

Bağlı Çizgeler/Connected Graphs

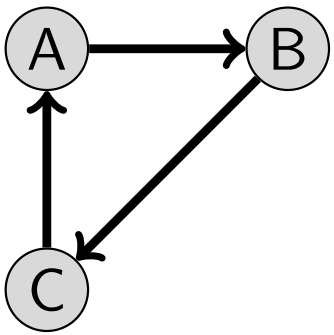


Figure: Bağlı Çizge(Connected Graph)

- A, B
- B, C
- B, A
- C, A
- C, B

Tam Bağlı Çizge/Completely Connected Graphs

Tanım

Herhangi iki düğüm arasında bir kenar varsa buna **tam bağlı çizge(completely connected graph) adı verilir**.

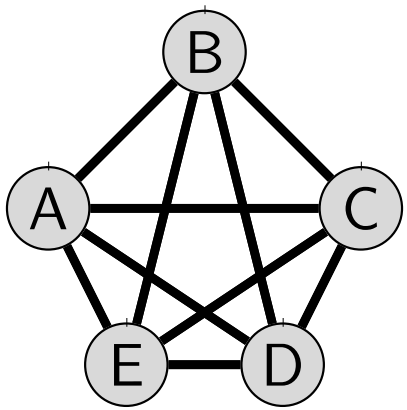


Figure: Tam Bağlı Çizge

Derinine Arama/Depth First Search

Tanım

Dolaşma işlemi bir düğümden başlar. Gidilebilecek mümkün komşulardan bir tanesini seçer ve ilerler. Gidilebilecek diğer düğümlere bakmadan önce yeni gidilen düğümün komşuları incelenir ve bir tanesine gidilir. Gidilebilecek komşu kalmayınca geri sarılarak mümkün komşular aranır. Tek biçimlikik sağlamak için mümkün komşulardan alfabetik olarak küçük olan önce seçilir.

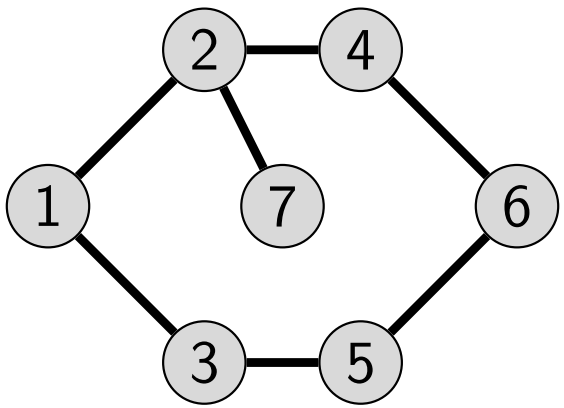


Figure: DFS Örneği(1 düğümünden başlansın)

Derinine Arama/Depth First Search

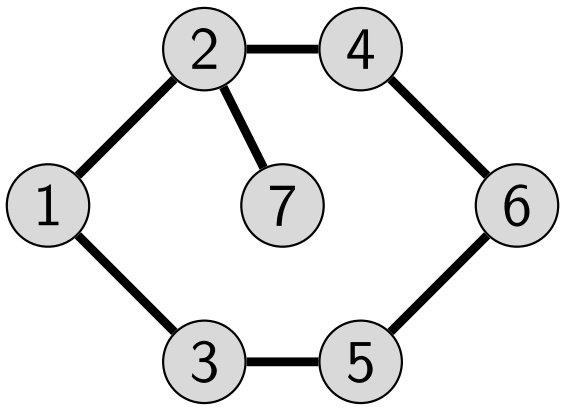


Figure: DFS Örneği(1 düğümünden başlansın)

1,

Derinine Arama/Depth First Search

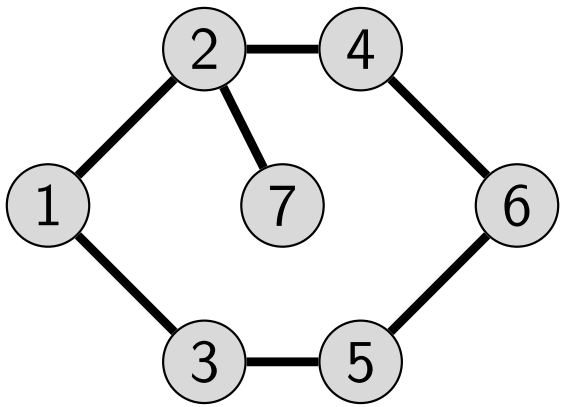


Figure: DFS Örneği(1 düğümünden başlansın)

1, 2,

Derinine Arama/Depth First Search

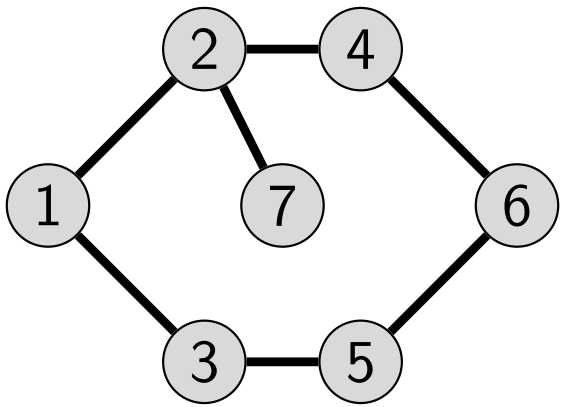


Figure: DFS Örneği(1 düğümünden başlansın)

1, 2, 4,

Derinine Arama/Depth First Search

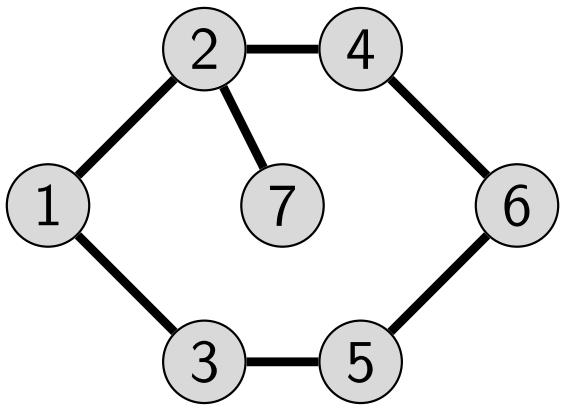


Figure: DFS Örneği(1 düğümünden başlansın)

1, 2, 4, 6,

Derinine Arama/Depth First Search

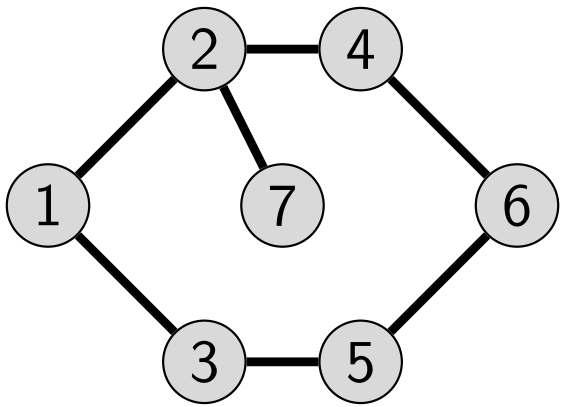


Figure: DFS Örneği(1 düğümünden başlansın)

1, 2, 4, 6, 5,

Derinine Arama/Depth First Search

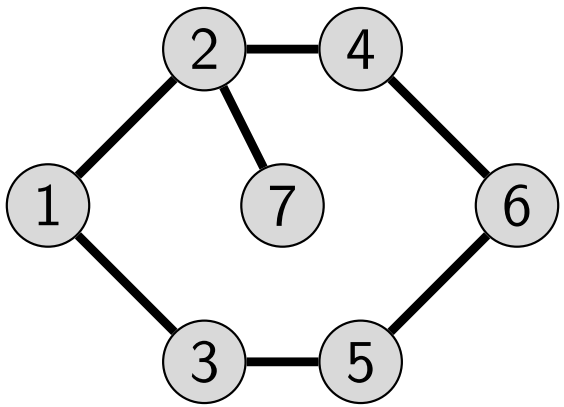


Figure: DFS Örneği(1 düğümünden başlansın)

1, 2, 4, 6, 5, 3,

Derinine Arama/Depth First Search

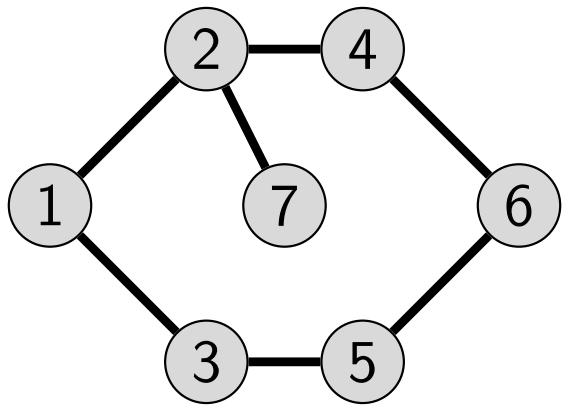


Figure: DFS Örneği(1 düğümünden başlansın)

1, 2, 4, 6, 5, 3, 7

DFS Yalancı Kodu

```
DFS(kaynak):
  s <- yeni yığıt(new stack)
  visited <- {} // boş küme
  s.push(kaynak)
  while (s boş olmadığı sürece):
    mevcut <- s.pop()
    if (mevcut, visited kümesi içindeyse):
        continue
    visited.add(mevcut)
    // do something with current
    (mevcut,v) kenarına sahip bütün v düğümleri için:
        s.push(v)
```

Enine Arama(Breadth First Search)

Tanım

Dolaşma işlemi bir düğümden başlar. Gidilebilecek mümkün komşularhepsini sırayla seçer ve ilerler. Gidilebilecek bütün komşular gezildikten sonra ilk komşunun komşuları seçilir. Tek biçimlikik sağlamak için mümkün komşulardan alfabetik olarak küçük olan önce seçilir.

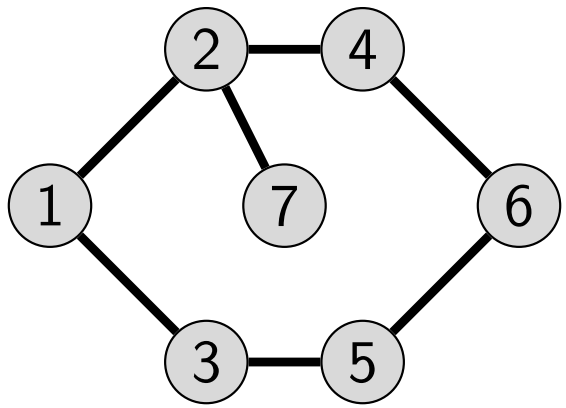


Figure: BFS Örneği(1 düğümünden başlansın)

Enine Arama(Breadth First Search)

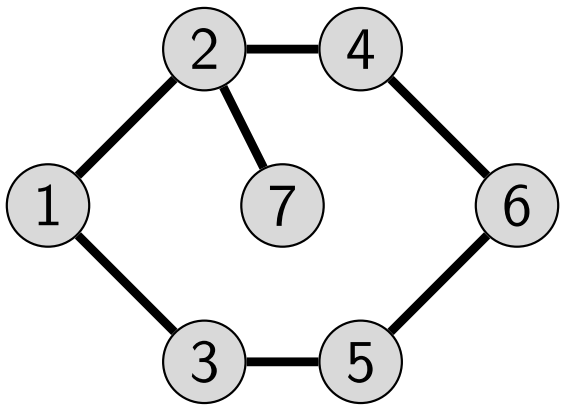


Figure: BFS Örneği(1 düğümünden başlansın)

1,

Enine Arama(Breadth First Search)

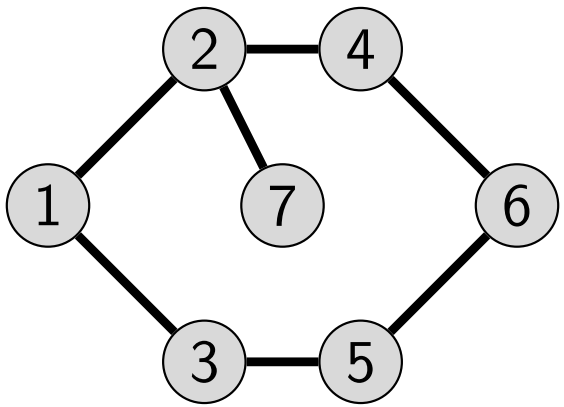


Figure: BFS Örneği(1 düğümünden başlansın)

1, 2,

Enine Arama(Breadth First Search)

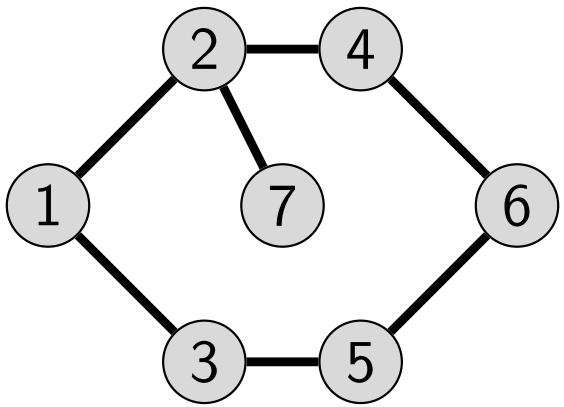


Figure: BFS Örneği(1 düğümünden başlansın)

1, 2, 3,

Enine Arama(Breadth First Search)

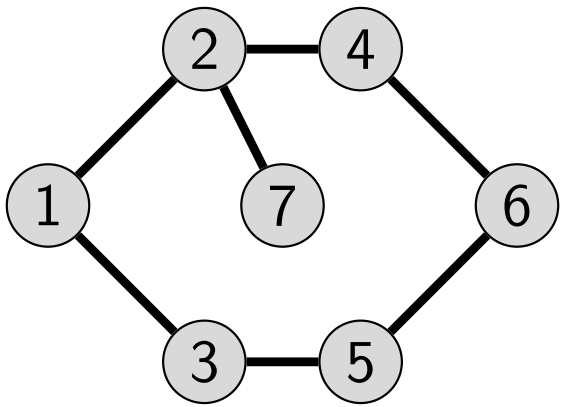


Figure: BFS Örneği(1 düğümünden başlansın)

1, 2, 3, 4,

Enine Arama(Breadth First Search)

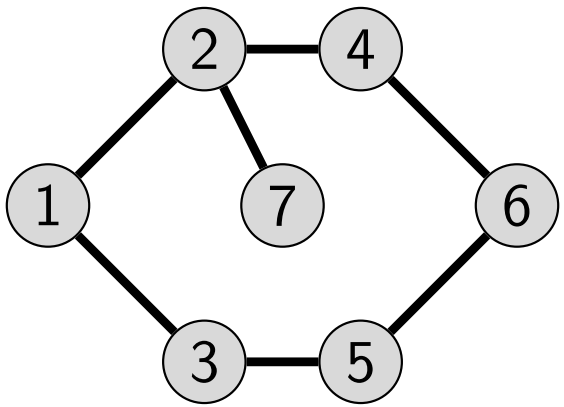


Figure: BFS Örneği(1 düğümünden başlansın)

1, 2, 3, 4, 7,

Enine Arama(Breadth First Search)

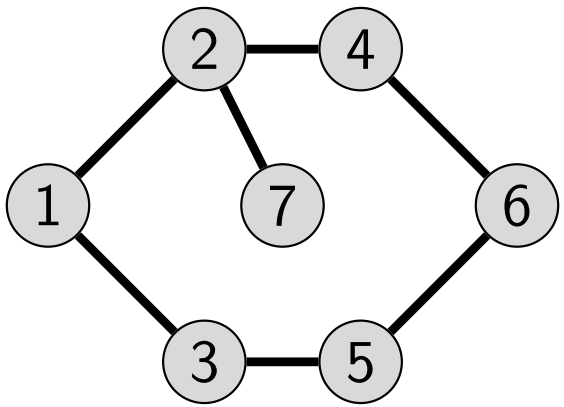


Figure: BFS Örneği(1 düğümünden başlansın)

1, 2, 3, 4, 7, 5,

Enine Arama(Breadth First Search)

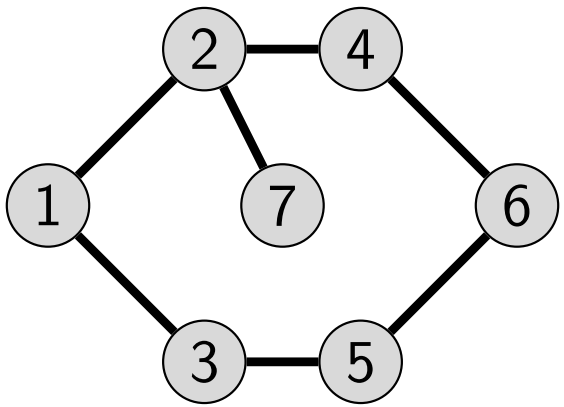


Figure: BFS Örneği(1 düğümünden başlansın)

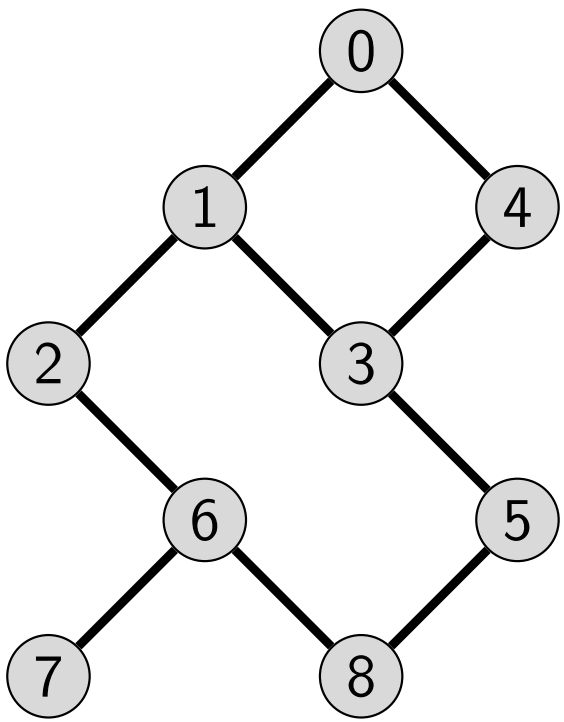
1, 2, 3, 4, 7, 5, 6

BFS Yalancı Kodu

```
BFS(kaynak):
  q <- yeni kuyruk(new queue)
  visited <- {} // boş küme
  q.enqueue(kaynak)
  while (q boş olmadığı sürece):
    mevcut <- q.dequeue()
    if (mevcut, visited kümesi içindeyse):
        continue
    visited.add(mevcut)
    // do something with current
    (mevcut,v) kenarına sahip bütün v düğümleri için:
        q.enqueue(v)
```

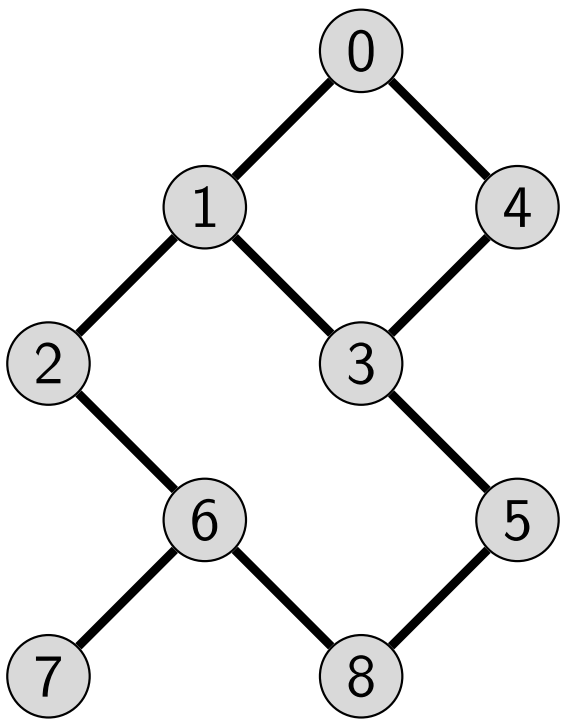
Soru

Aşağıdaki çizgede 0 düğümünden başlayarak derinine(DFS) ve enine(BFS) arama gerçekleştirin.



Soru

Aşağıdaki çizgede 0 düğümünden başlayarak derinine(DFS) ve enine(BFS) arama gerçekleştirin.

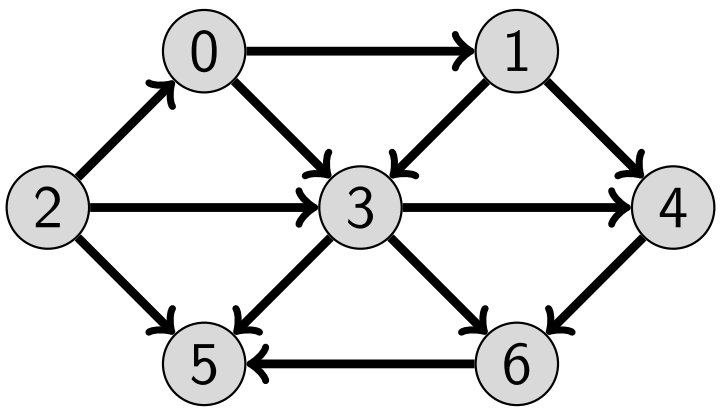


DFS: 0, 1, 2, 6, 7, 8, 5, 3, 4
BFS: 0, 1, 4, 2, 3, 6, 5, 7, 8

Topolojik Sıralama/Topological Sorting

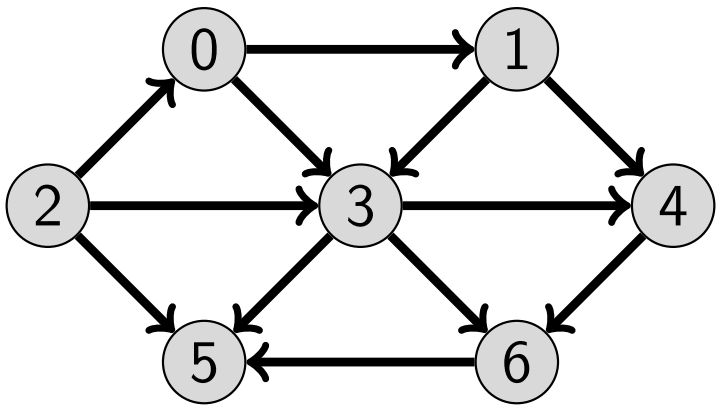
Tanım

Topolojik sıralama yönlü döngü içermeyen çizgelerde(directed acyclic graphs(DAG)) çalışan bir sıralama çeşididir.



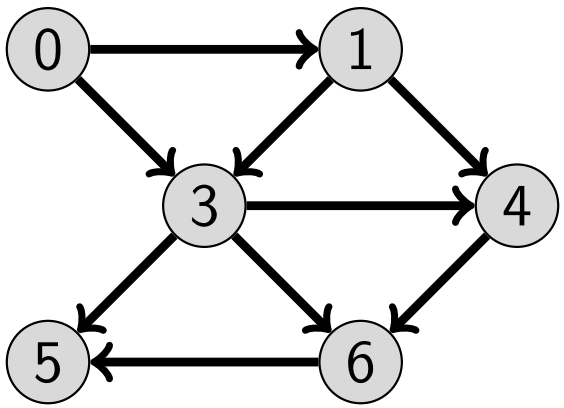
- Çizgede düğüm olduğu sürece giriş derecesi(Indegree) 0 olan bir düğüm seç ve çizgeden çıkar

Topolojik Sıralama Örneği



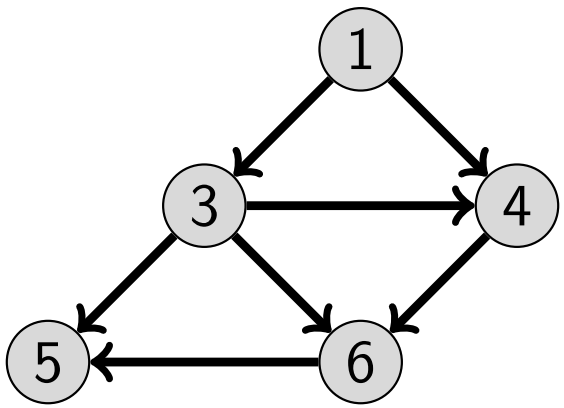
Sıralama:

Topolojik Sıralama Örneği



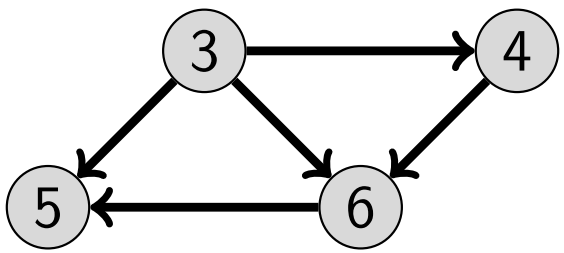
Sıralama: 2

Topolojik Sıralama Örneği



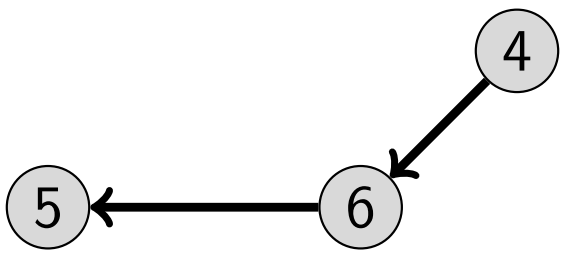
Sıralama: 2, 0

Topolojik Sıralama Örneği



Sıralama: 2, 0, 1

Topolojik Sıralama Örneği



Sıralama: 2, 0, 1, 3

Topolojik Sıralama Örneği

Sıralama: 2, 0, 1, 3, 4

Topolojik Sıralama Örneği

Sıralama: 2, 0, 1, 3, 4, 6,

Topolojik Sıralama Örneği

Sıralama: 2, 0, 1, 3, 4, 6, 5

Komşuluk Matrisi/Adjacency Matrix

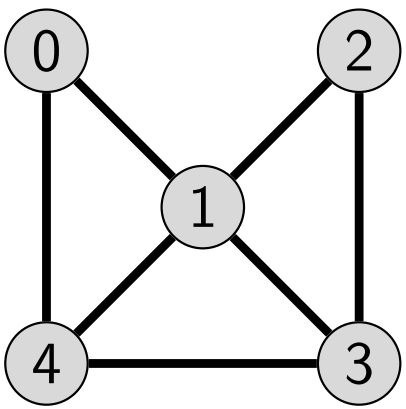
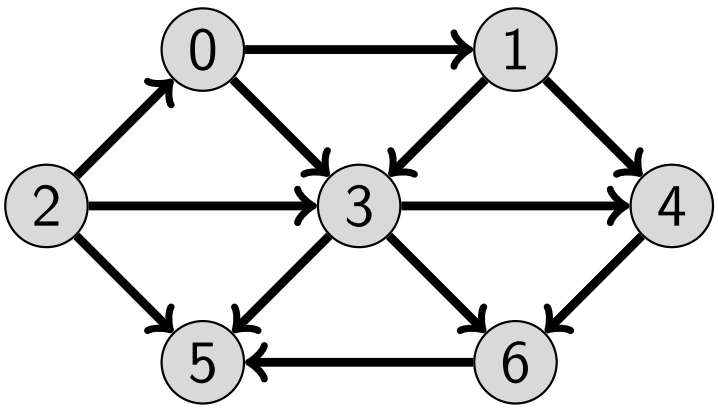


Table: Komşuluk Matrisi

	0	1	2	3	4
0	**0**	**1**	**0**	**0**	**1**
1	**1**	**0**	**1**	**1**	**1**
2	**0**	**1**	**0**	**1**	**0**
3	**0**	**1**	**1**	**0**	**1**
4	**1**	**1**	**0**	**1**	**0**

Soru

Aşağıdaki çizgenin komşuluk matrsini oluşturun.



Soru

Aşağıdaki çizgenin komşuluk matrsini oluşturun.

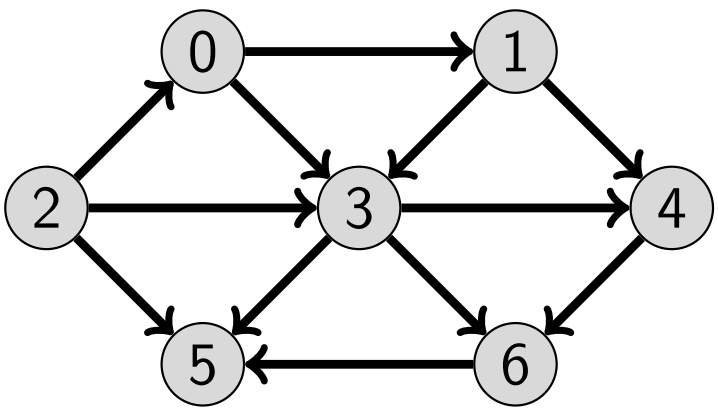
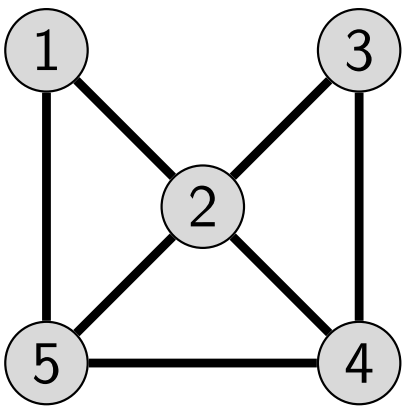
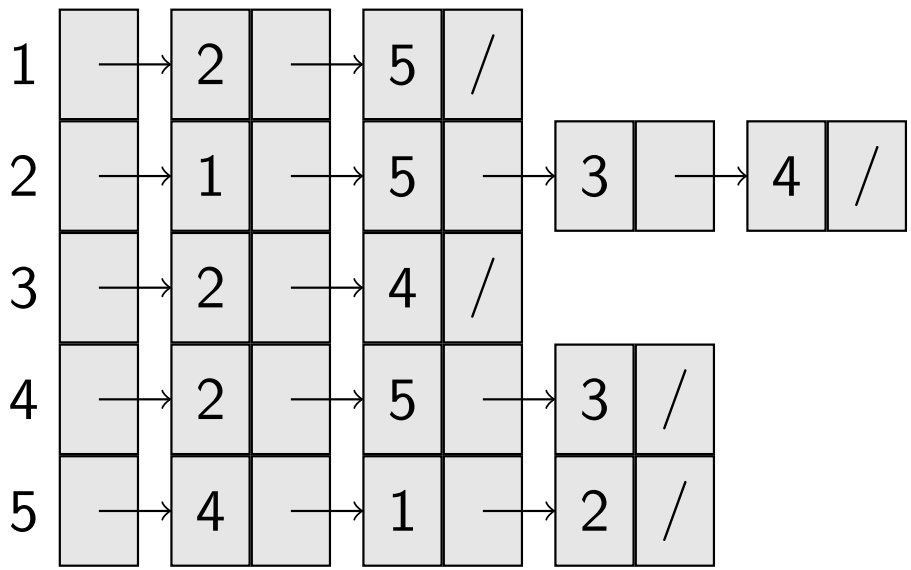


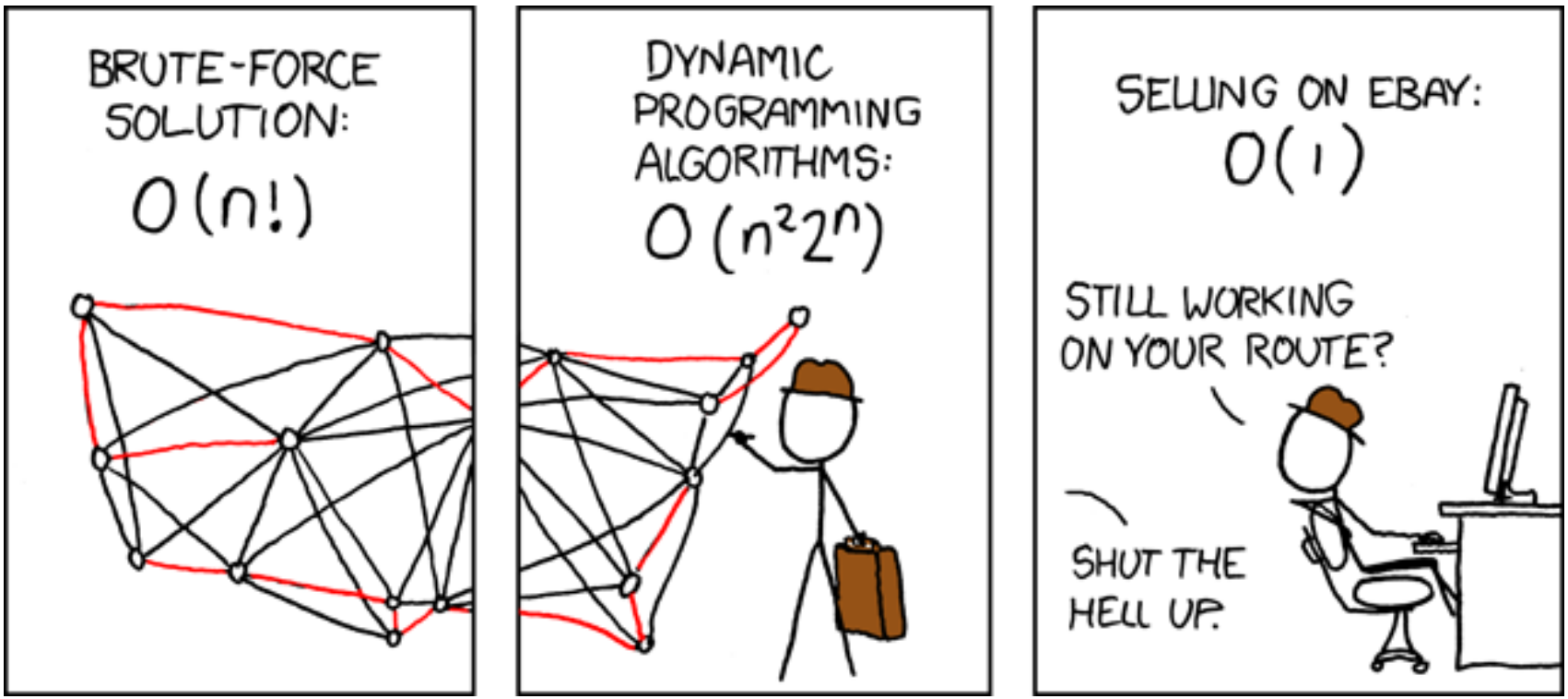
Table: Komşuluk Matrisi

	0	1	2	3	4	5	6
0	0	**1**	0	**1**	0	0	0
1	0	0	0	**1**	**1**	0	0
2	**1**	0	0	**1**	0	**1**	0
3	0	0	0	0	**1**	**1**	**1**
4	0	0	0	0	0	0	**1**
5	0	0	0	0	0	0	0
6	0	0	0	0	0	**1**	0

Komşuluk Listesi/Adjacency List






BRUTE-FORCE SOLUTION:
O(n!)
DYNAMIC PROGRAMMING ALGORITHMS:
O(n²2ⁿ)
SELLING ON EBAY:
O(1)
STILL WORKING ON YOUR ROUTE?
SHUT THE HELL UP.

1

[1] xkcd