

# CENG 201 Veri Yapıları 4: Ağaçlar

Öğr.Gör. Şevket Umut ÇAKIR

Pamukkale Üniversitesi

Hafta 4

# Anahat

## ① Ağaçlar

## ② İkili Ağaçlar

Dolaşma/Traversal

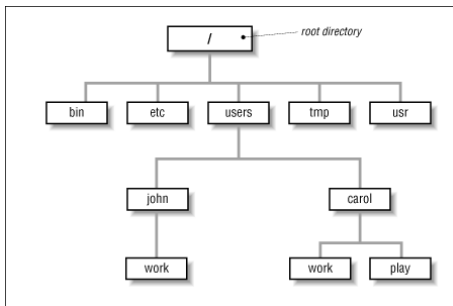
## ③ İkili Arama Ağaçları

Arama, Ekleme ve Silme

# Ağaçlar

## Tanım

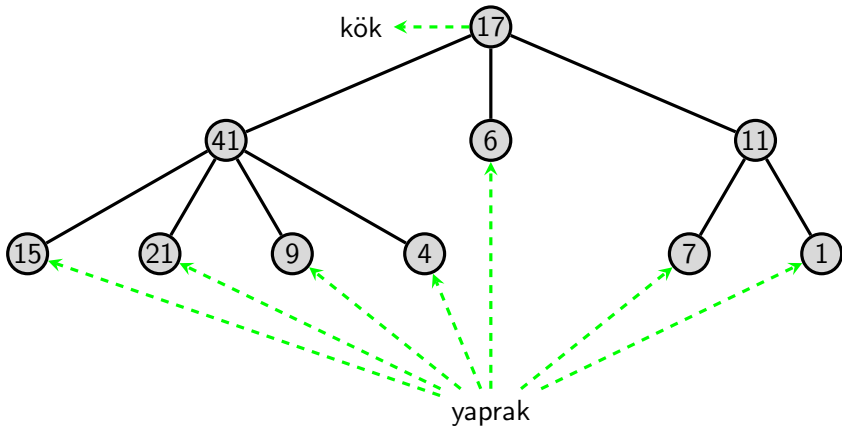
Ağaçlar, düğümlerden ve düğümleri birbirine bağlayan kenarlardan oluşan; herhangi iki düğümü birbirine bağlayan sadece bir yolun bulunduğu veri yapılarıdır.



# Terimler

- **Düğüm(node)**: Ağacın her bir elemanına düğüm adı verilir.
- **Kök(root)**: En tepede bulunan düğümdür.
- **Ebeveyn(parent)**: Altında başka düğümler bulunan, seviye olarak çocuk düğümlerden üstte olan düğümdür.
- **Çocuk(child)**: Bir ebeveyn düğüme bağlı olan düğümlerdir.
- **Derinlik(depth)**: Bir düğümün kök düğüme olan uzaklığıdır.
- **Ağacın derinliği(depth of the tree)**: Derinliği en büyük olan yaprağın derinliğine eşittir.
- **Yaprak(leaf)**: Çocuğu bulunmayan düğümlerdir.
- **Kardeş(sibling)**: Aynı ebeveyne sahip düğümlerdir.
- **Ata(ancestor)**: Bir düğümden köke kadar olan düğümlerin hepsi
- **Soy(descendant)**: Bir düğümün tüm alt dallarındaki düğümler

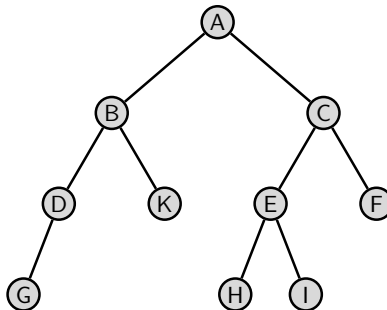
# Örnek ağaç



# İkili Ağaçlar

## Tanım

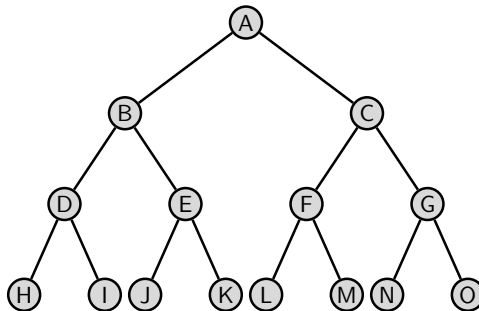
Her bir düğümünde en fazla iki çocuğu olan ağaçlara ikili ağaçlar denir. Bir düğümün sol ve sağ çocuğu olabilir.



# Dolu İkili Ağaçlar/Full Binary Tree

## Tanım

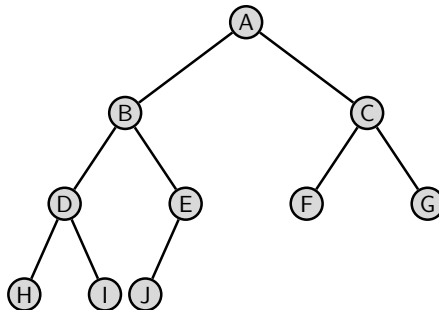
Tüm yaprakları aynı seviyede olan ikili ağaçlardır.



# Bütün İkili Ağaçlar/Complete Binary Tree

## Tanım

Dolu ikili ağaca yaprak değerleri soldan itibaren eklendiğinde oluşan ikili ağaç.



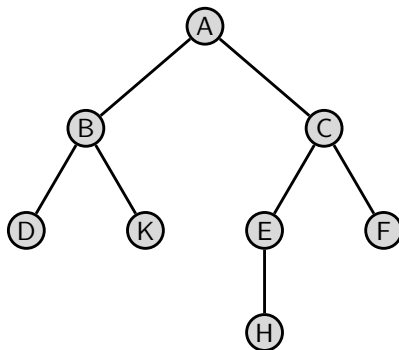


# Ağaçlarda dolaşma/Traversal

İkili ağaçlarda üç farklı dolaşma şekli vardır:

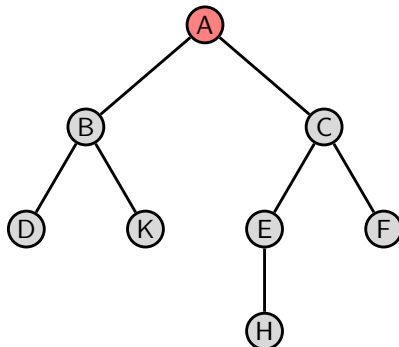
- **Önce değer(Preorder):** Önce değer elde edilir, sonra sırasıyla sol ve sağ alt ağaç önce değer olarak dolaşılır.
- **Ortada değer(Inorder):** Önce sol alt ağaç ortada değer olarak dolaşılır, sonra değer elde edilir, son olarak sağ alt ağaç ortada değer olarak dolaşılır.
- **Sonra değer(Postorder):** Önce sırasıyla sol ve sağ alt ağaçlar sonra değer şeklinde gezilir, sonra değer elde edilir.

## Preorder dolaşma



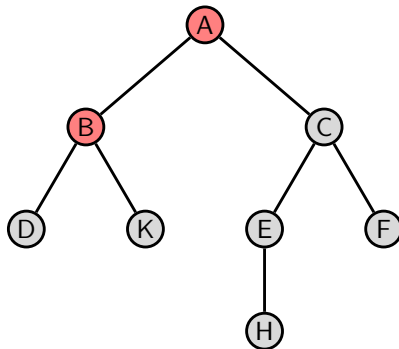
## Preorder dolaşma 1

**Dolaşma: A**



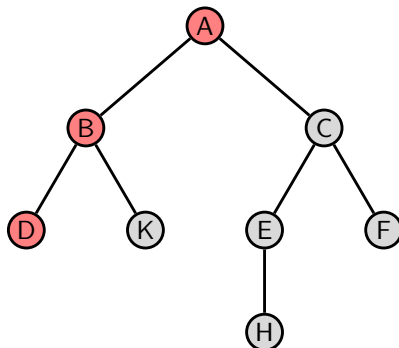
## Preorder dolaşma 2

**Dolaşma:** A, B



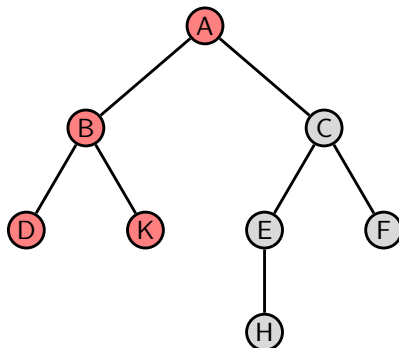
## Preorder dolaşma 3

**Dolaşma:** A, B, D



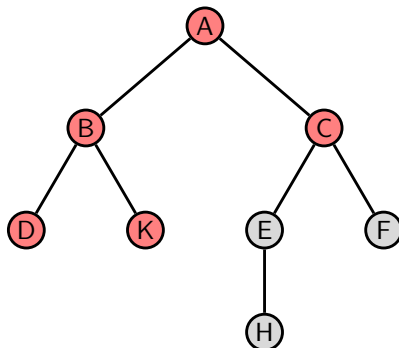
# Preorder dolaşma 4

**Dolaşma:** A, B, D, K



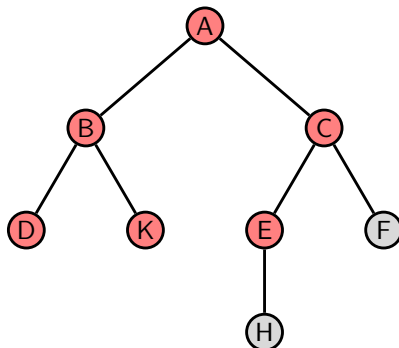
# Preorder dolaşma 5

**Dolaşma:** A, B, D, K, C



# Preorder dolaşma 6

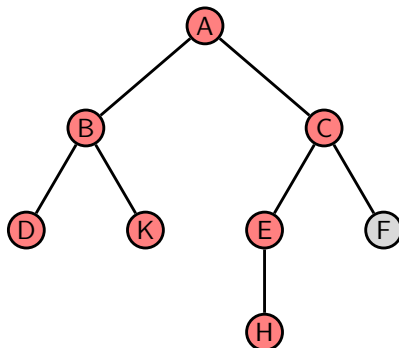
**Dolaşma:** A, B, D, K, C, E





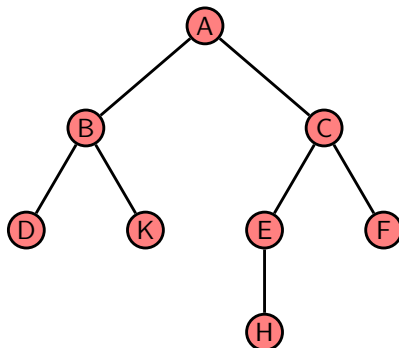
# Preorder dolaşma 7

**Dolaşma:** A, B, D, K, C, E, H

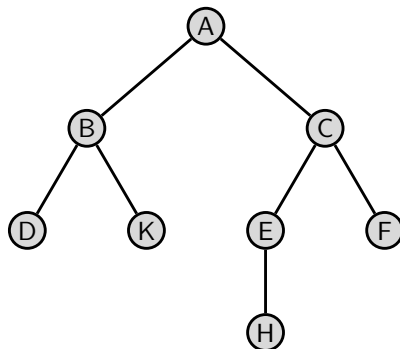


# Preorder dolaşma 8

**Dolaşma:** A, B, D, K, C, E, H, F

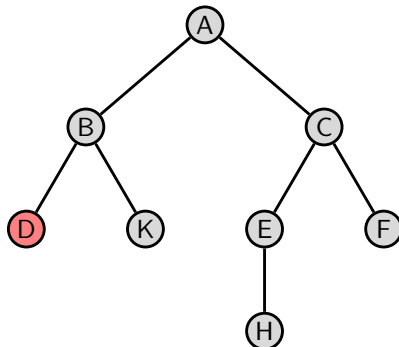


# Inorder dolaşma



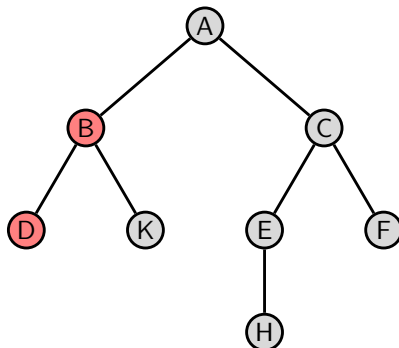
# Inorder dolaşma 1

**Dolaşma:** D



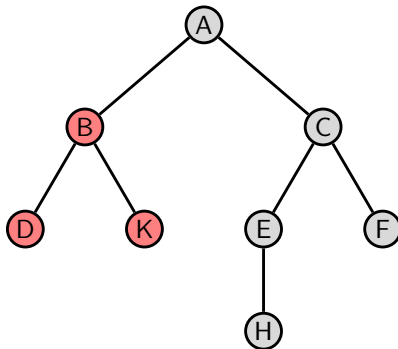
# Inorder dolaşma 2

**Dolaşma:** D, B



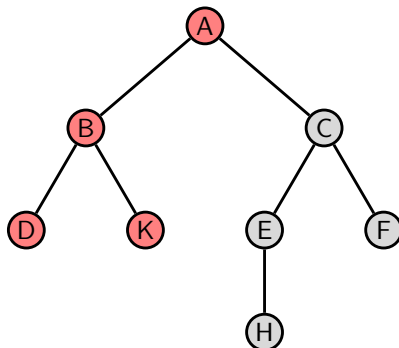
# Inorder dolaşma 3

**Dolaşma:** D, B, K



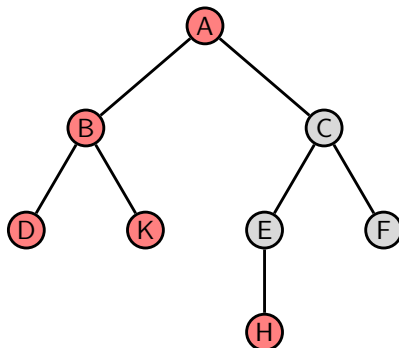
# Inorder dolaşma 4

**Dolaşma:** D, B, K, A



# Inorder dolaşma 5

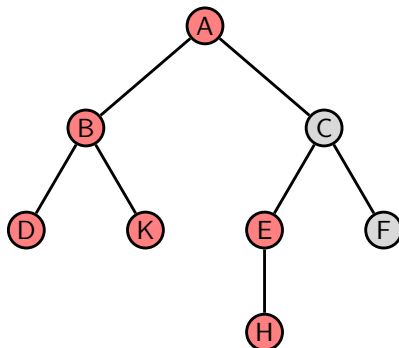
**Dolaşma:** D, B, K, A, H





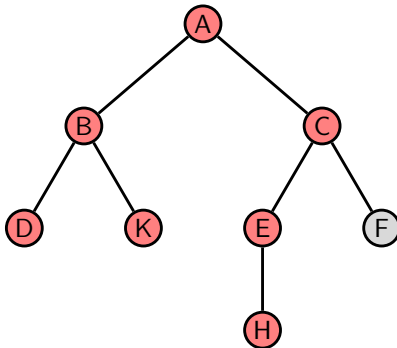
# Inorder dolaşma 6

**Dolaşma:** D, B, K, A, H, E



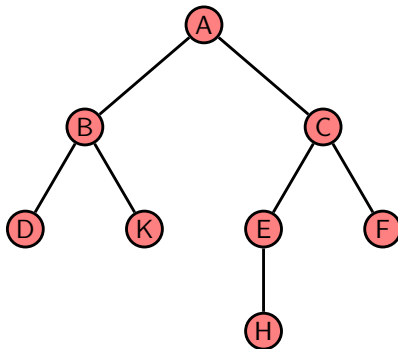
# Inorder dolaşma 7

**Dolaşma:** D, B, K, A, H, E, C

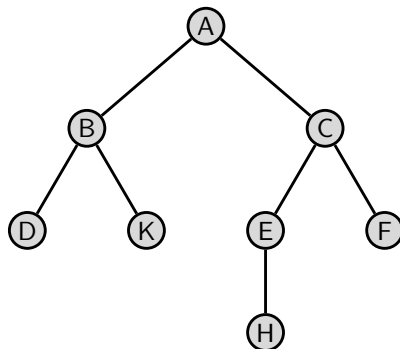


# Inorder dolaşma 8

**Dolaşma:** D, B, K, A, H, E, C, F

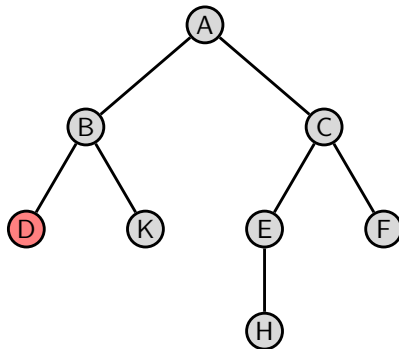


# Postorder dolaşma



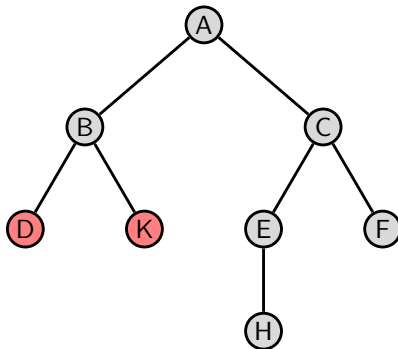
# Postorder dolaşma 1

**Dolaşma:** D



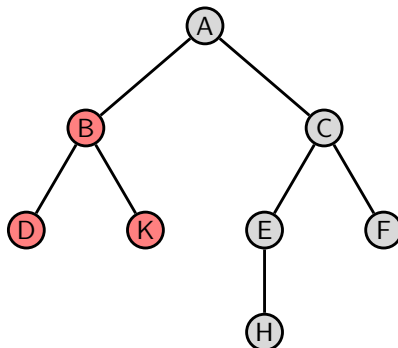
# Postorder dolaşma 2

**Dolaşma:** D, K



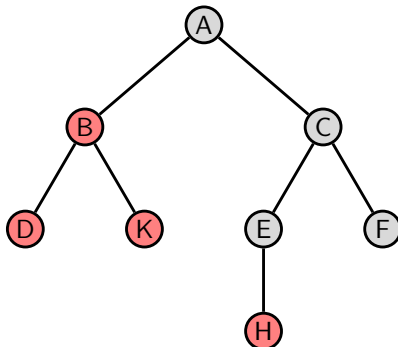
# Postorder dolaşma 3

**Dolaşma:** D, K, B



# Postorder dolaşma 4

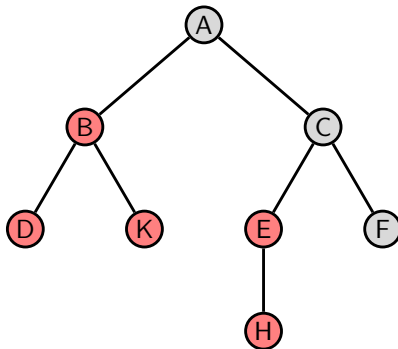
**Dolaşma:** D, K, B, H





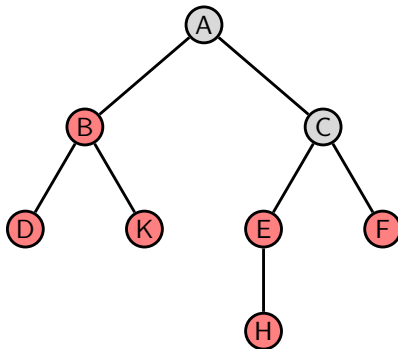
# Postorder dolaşma 5

**Dolaşma:** D, K, B, H, E



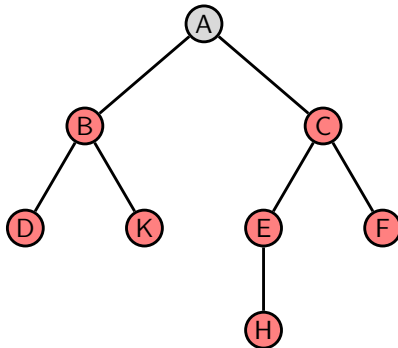
# Postorder dolaşma 6

**Dolaşma:** D, K, B, H, E, F



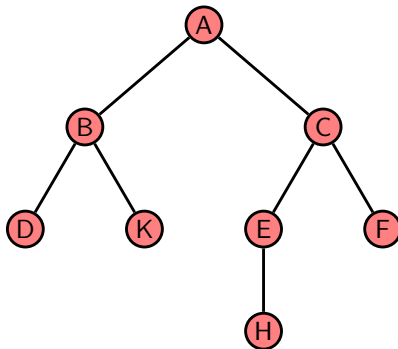
# Postorder dolaşma 7

**Dolaşma:** D, K, B, H, E, F, C



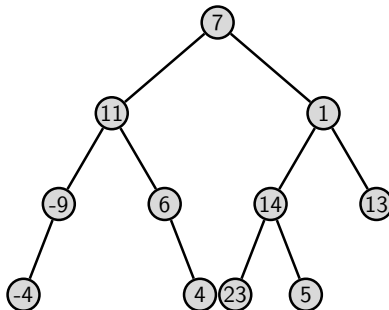
# Postorder dolaşma 8

**Dolaşma:** D, K, B, H, E, F, C, A



# Soru

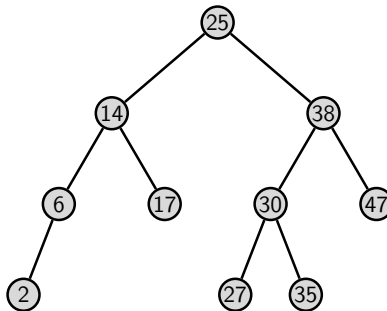
Aşağıdaki ikili ağacın preorder, inorder ve postorder dolaşımları nasıl olur?



# İkili Arama Ağaçları

## Tanım

Kökün solundaki tüm anahtar değerleri kökten küçük, kökün sağındaki her bir anahtar değeri kökten büyük ve bu iki özelliği tüm düğümlerde geçerli olan ikili ağaç yapısıdır.

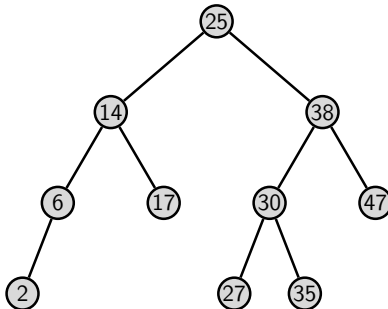


# Arama İşlemi

```
Düğüm Ara(Düğüm, Aranacak)
    Eğer Düğüm boş ise (null)
        return null
    Aksi halde
        Eğer Düğüm.değer=Aranacak
            return Düğüm
        Eğer Düğüm.değer<Aranacak
            return Ara(Düğüm.sağ, Aranacak)
        Aksi halde
            return Ara(Düğüm.sol, Aranacak)
```

# Arama işlemi

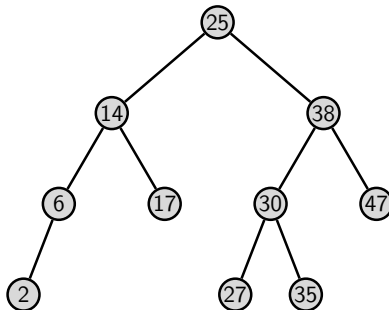
35 ve 15 değerlerini aşağıdaki ağaçta arayalım:





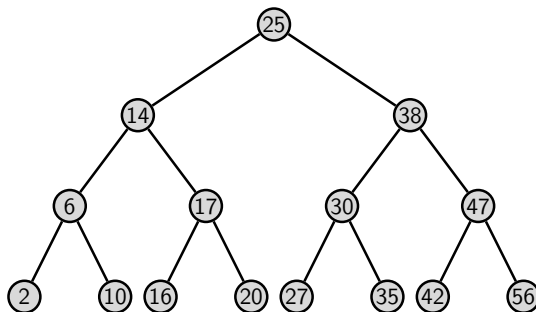
# Arama işlemi

35 ve 15 değerlerini aşağıdaki ağaçta arayalım:

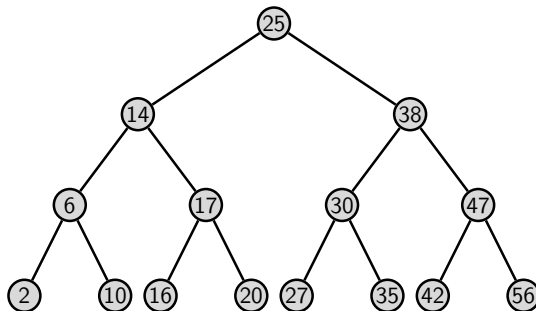


- Arama işlemi bu ağaçta en iyi ve en kötü ihtimalle kaç adımda biter?
- Karşılaşılabilecek en iyi ve en kötü ağaçlar nasıldır?

# Arama en iyi durum

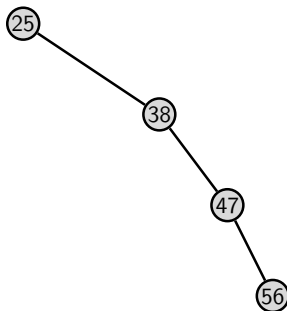


# Arama en iyi durum

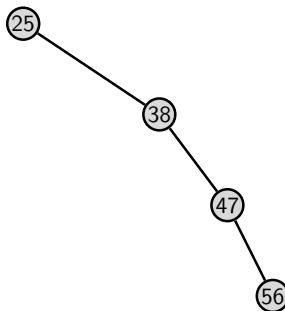


N elemanlı bir dengeli ağaçta en fazla kaç adımda aranılan değer bulunabilir?

# Arama en kötü durum



# Arama en kötü durum



N elemanlı bir ikili ağaçta en fazla kaç adımda aranılan değer bulunabilir?

# Ekleme İşlemi

Ekle(Düğüm, Eklenecek)

Eğer Düğüm.değer=Eklenecek

HATA!!! veya ekleme yapma

Eğer Düğüm.değer<Eklenecek

Eğer Düğüm.sağ boş ise

Düğüm.sağ=yeni düğüm

Aksi Halde

Ekle(Düğüm.sağ, Eklenecek)

Eğer Düğüm.değer>Eklenecek

Eğer Düğüm.sol boş ise

Düğüm.sol=yeni düğüm

Aksi Halde

Ekle(Düğüm.sol, Eklenecek)

# Ekleme işlemi

20, 14, 25, 29, 36, 12, 5, 16, 21, 27, 35, 56, 8 değerlerini sırasıyla ikili arama ağacına ekleyin.

Ekleme işlemi: 20

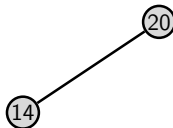
20, 14, 25, 29, 36, 12, 5, 16, 21, 27, 35, 56, 8

20



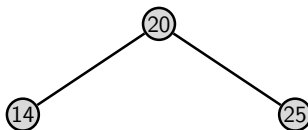
# Ekleme işlemi: 14

20, 14, 25, 29, 36, 12, 5, 16, 21, 27, 35, 56, 8



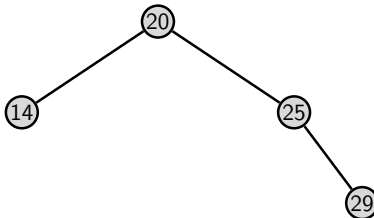
# Ekleme işlemi: 25

20, 14, 25, 29, 36, 12, 5, 16, 21, 27, 35, 56, 8



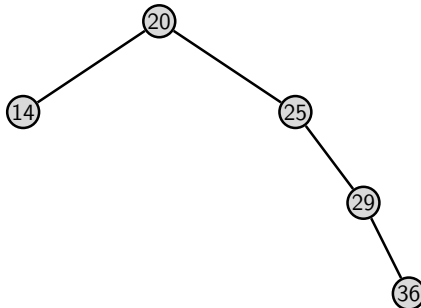
# Ekleme işlemi: 29

20, 14, 25, 29, 36, 12, 5, 16, 21, 27, 35, 56, 8



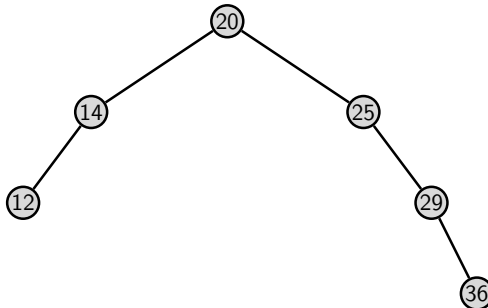
# Ekleme işlemi: 36

20, 14, 25, 29, 36, 12, 5, 16, 21, 27, 35, 56, 8



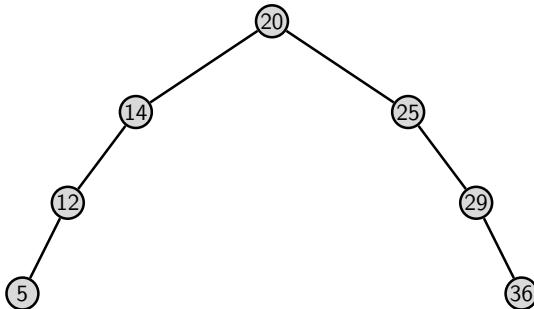
# Ekleme işlemi: 12

20, 14, 25, 29, 36, 12, 5, 16, 21, 27, 35, 56, 8



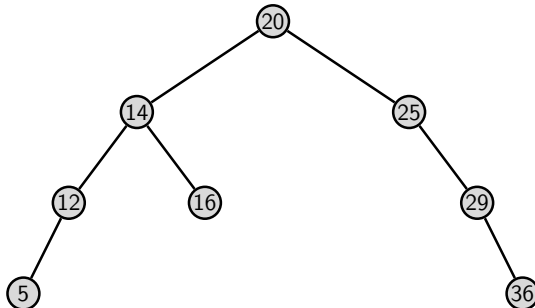
# Ekleme işlemi: 5

20, 14, 25, 29, 36, 12, 5, 16, 21, 27, 35, 56, 8



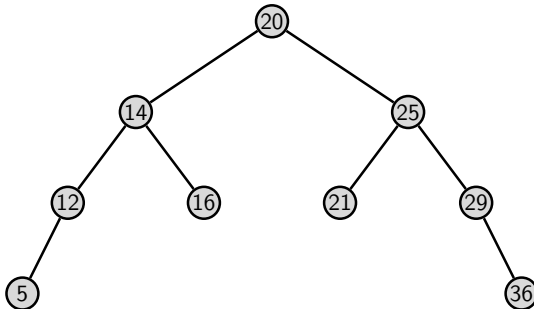
# Ekleme işlemi: 16

20, 14, 25, 29, 36, 12, 5, 16, 21, 27, 35, 56, 8



# Ekleme işlemi: 21

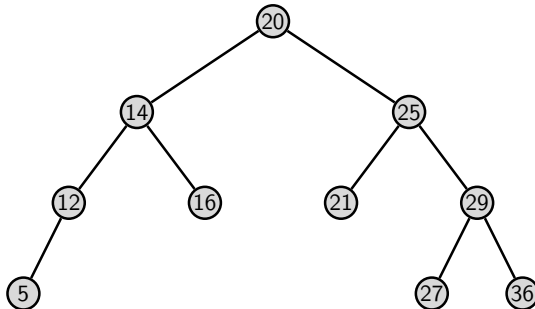
20, 14, 25, 29, 36, 12, 5, 16, 21, 27, 35, 56, 8





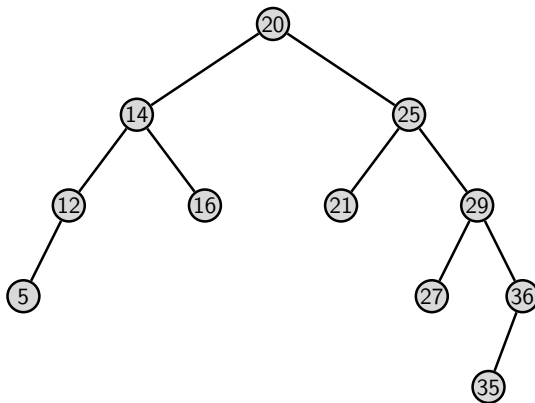
# Ekleme işlemi: 27

20, 14, 25, 29, 36, 12, 5, 16, 21, 27, 35, 56, 8



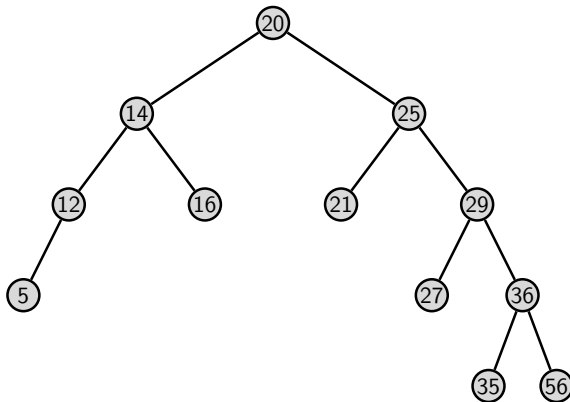
# Ekleme işlemi: 35

20, 14, 25, 29, 36, 12, 5, 16, 21, 27, 35, 56, 8



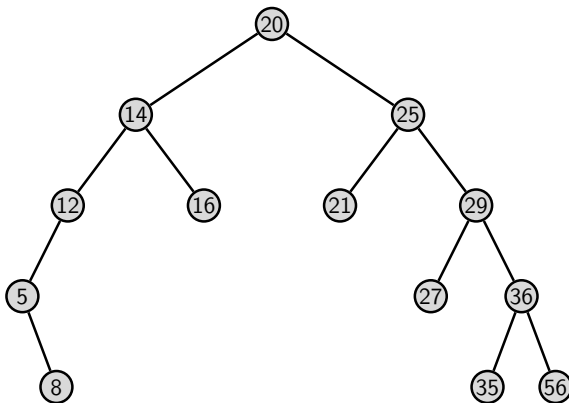
# Ekleme işlemi: 56

20, 14, 25, 29, 36, 12, 5, 16, 21, 27, 35, 56, 8



# Ekleme işlemi: 8

20, 14, 25, 29, 36, 12, 5, 16, 21, 27, 35, 56, 8

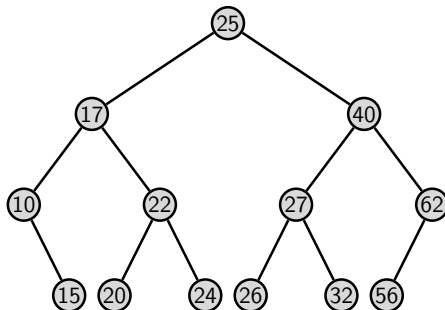


# Ekleme işlemi

25, 17, 40, 22, 27, 62, 26, 32, 20, 56, 10, 24, 15 değerlerini sırasıyla ikili arama ağacına ekleyin.

# Ekleme işlemi

25, 17, 40, 22, 27, 62, 26, 32, 20, 56, 10, 24, 15 değerlerini sırasıyla ikili arama ağacına ekleyin.



# Silme işlemi

Silme için 3 farklı durum söz konusudur:

- Silinecek eleman yaprak ise
- Silinecek elemanın sadece bir çocuğu varsa
- Silinecek elemanın iki çocuğu da varsa

# Silme işlemi

Silme için 3 farklı durum söz konusudur:

- Silinecek eleman yaprak ise
- Silinecek elemanın sadece bir çocuğu varsa
- Silinecek elemanın iki çocuğu da varsa

## Ardıl/Successor

Ağaçta verilen değerden büyük değerlerin en küçüğüdür.

## Öncel/Predecessor

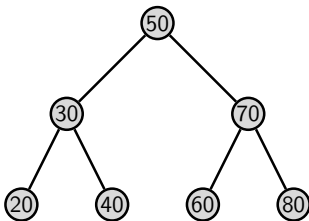
Ağaçta verilen değerden küçük değerlerin en büyüğüdür.



# Silinecek eleman yaprak ise

Silinecek eleman yaprak ise ebeveyni ile bağlantısını kopartmak yeterlidir.

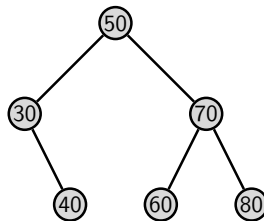
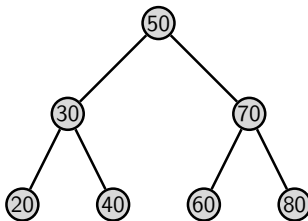
Delete 20



# Silinecek eleman yaprak ise

Silinecek eleman yaprak ise ebeveyni ile bağlantısını kopartmak yeterlidir.

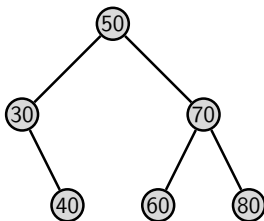
Delete 20



# Silinecek elemanın sadece bir çocuğu varsa

Silinecek elemanın sadece bir çocuğu varsa çocuk düğümü ilgili düğümün yerine kopyala ve düğümü sil.

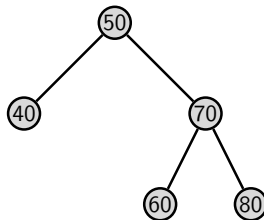
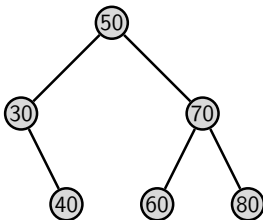
Delete 30



## Silinecek elemanın sadece bir çocuğu varsa

Silinecek elemanın sadece bir çocuğu varsa çocuk düğümü ilgili düğümün yerine kopyala ve düğümü sil.

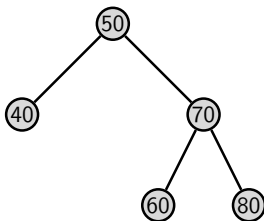
Delete 30



## Silinecek elemanın iki çocuğu da varsa

Silinecek elemanın iki çocuğu da varsa düğümün ardılı(successor) bulunur ve silinir, silinen düğüm ilgili düğümün yerine yerleştirilir.

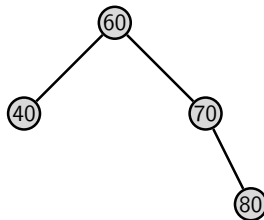
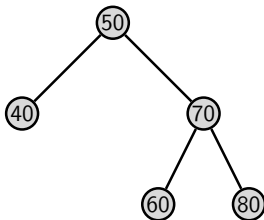
Delete 50



## Silinecek elemanın iki çocuğu da varsa

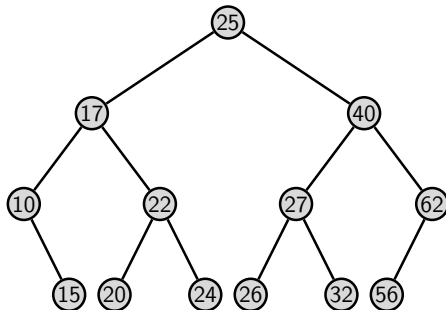
Silinecek elemanın iki çocuğu da varsa düğümün ardılı(successor) bulunur ve silinir, silinen düğüm ilgili düğümün yerine yerleştirilir.

Delete 50



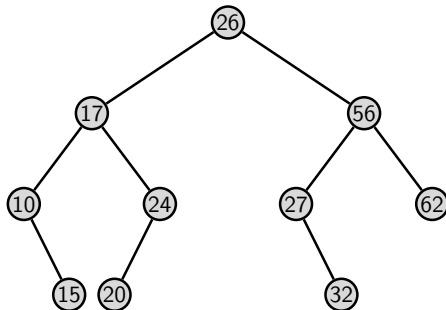
# Silme işlemi

22, 25, 40 değerlerini sırasıyla aşağıdaki ikili arama ağacından silin.

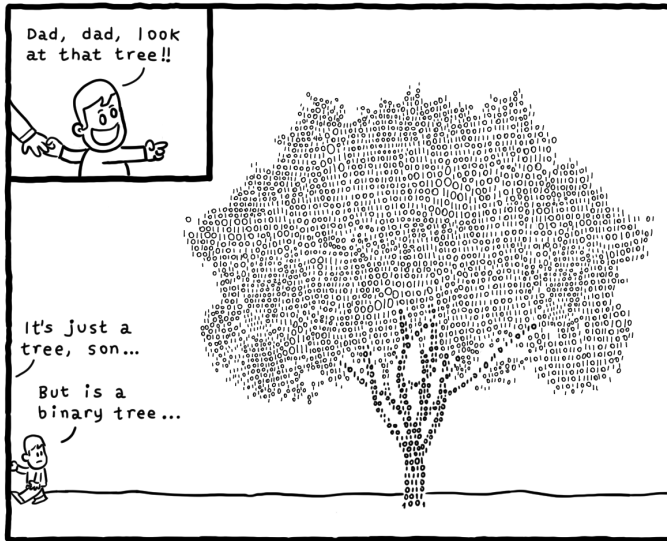


# Silme işlemi

22, 25, 40 değerleri silindikten sonra.







Daniel Stori {turnoff.us}