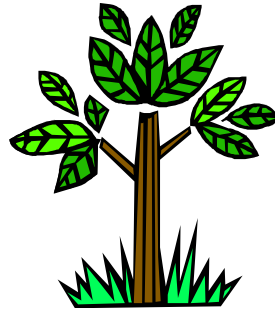
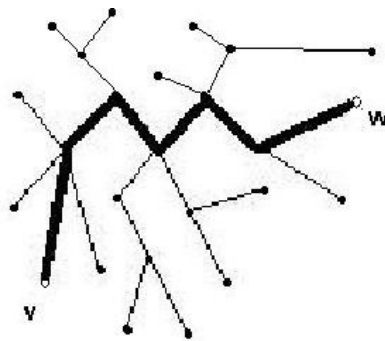


BÖLÜM 4 TREES (AĞAÇLAR)



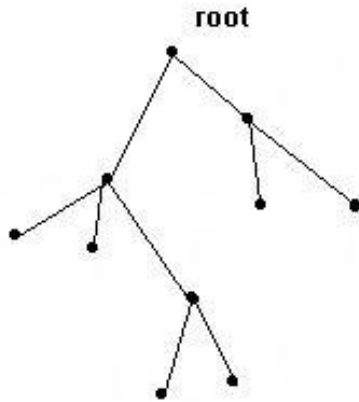
GİRİŞ



Bir ağaç

- v ve w düğüm çiftinden oluşan basit bir graftır
- v 'den w 'ya tek bir yol vardır
- dairesel bir yapı içermez

Köklü Ağaç (Rooted tree)



Düğümlerinden biri kök olarak (root) belirlenmiş bir ağaçtır

Düğüm seviyesi ve ağacın yüksekliği

T köklü bir ağaç olsun:

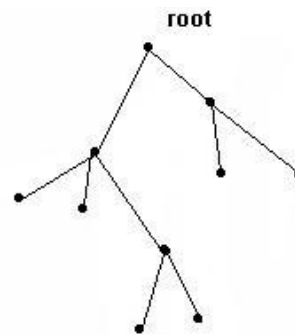
- Bir düğümün seviyesi, o düğümün $l(v)$ bulunduğu yerden ağacın köküne olan yolun uzunluğudur
- Bir ağacın yüksekliği, o ağacın maksimum seviyeye sahip olan düğümünün değerine eşittir

$$h = \max \{ l(v) \}$$

$$v \in V(T)$$

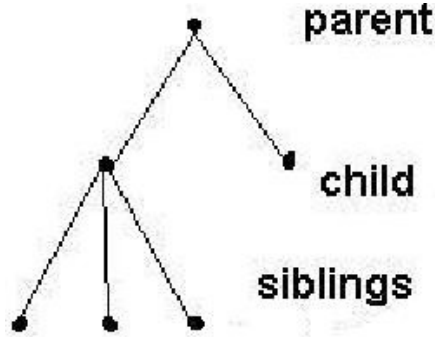
□ Örnek:

- sağdaki ağacın yüksekliği 3

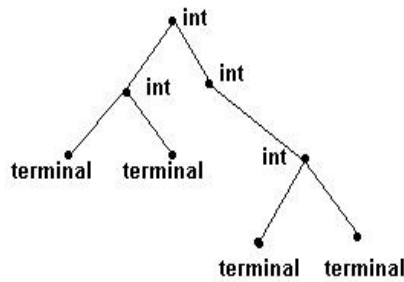


Terminoloji

- Parent (ana/baba)
- Ancestor (ata)
- Child (çocuk)
- Descendant (torun)
- Siblings(kardeş düğüm)
- Terminal vertices
(uç düğüm)
- Internal vertices
(ara düğüm)
- Subtrees (alt ağaçlar)



Ara (internal) ve Uç (terminal) Düğümler

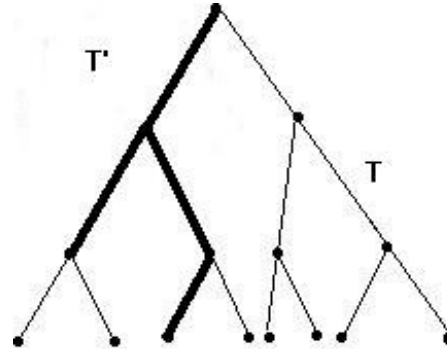


- En az bir çocuğa sahip olan düğümler ara (internal) düğümlerdir
- Hiç çocuğu olmayan düğümler uç (terminal) düğümlerdir
- Örnekteki ağaçta 4 adet internal ve 4 adet terminal düğüm mevcuttur

Alltağaçlar (Subtrees)

Bir T ağacının altağacı (subtree) T' olup

- $V(T') \subseteq V(T)$ and
- $E(T') \subseteq E(T)$



Ağaçların Karakteristiği

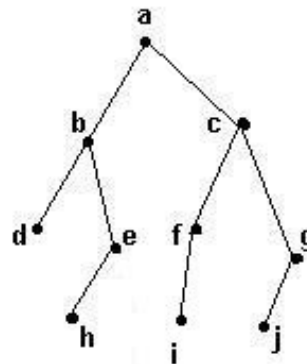
Theorem

Eğer T , n düğümlü bir graf ise aşağıdakiler doğrudur:

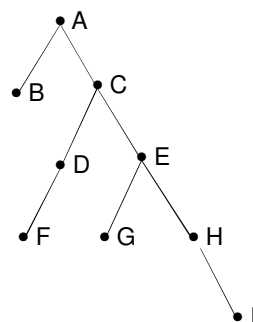
- a) T bir ağaçtır
- b) T bağlantılı (connected) ve acyclic
 - ("acyclic" = döngü mevcut değil)
- c) T bağlantılı ve $n-1$ kenara sahiptir
- d) T acyclic ve $n-1$ kenara sahiptir

İkili Ağaçlar (Binary trees)

Bir ikili ağaçta, her bir düğüm en fazla iki çocuğa sahiptir. Bir düğüm 0, 1 veya 2 çocuğa sahip olabilir

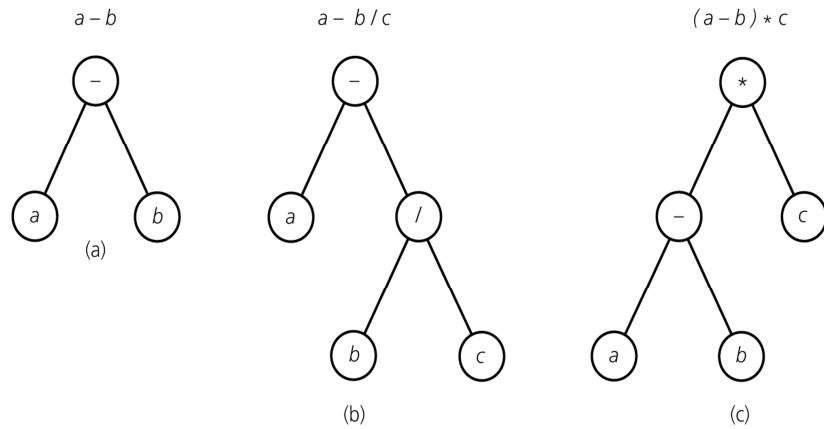


Binary Tree -- Örnek

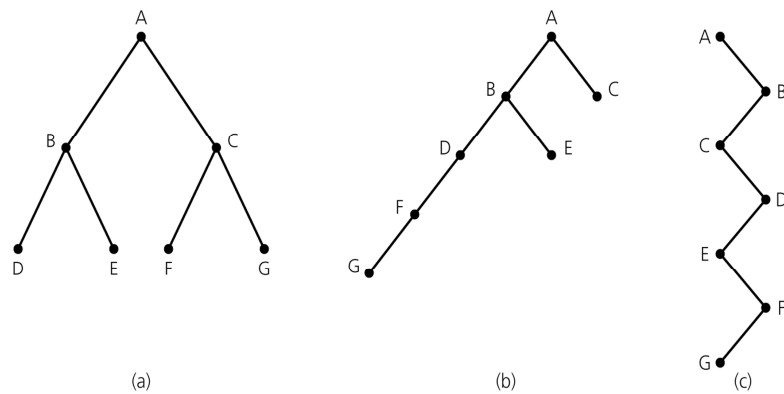


- A is the root.
- B is the left child of A, and C is the right child of A.
- D doesn't have a right child.
- H doesn't have a left child.
- B, F, G and I are leaves.

Binary Tree – Matematiksel İfadelerin Gösterilimi



11



Aynı düğüm sayısına sahip, farklı yüksekliklerdeki İkili Ağaçlar


12

Aynı düğüm sayısına sahip farklı İkili Ağaçlar

$n=0 \rightarrow$ empty tree

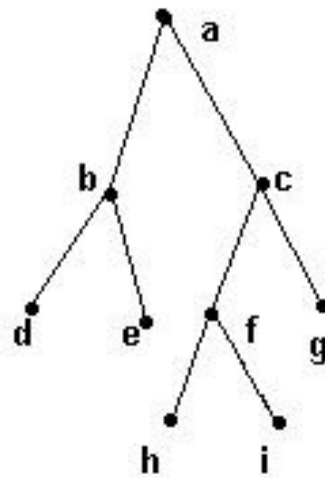
$n=1 \rightarrow$ • (1 tree)

$n=2 \rightarrow$  (2 trees)

$n=3 \rightarrow$  (5 trees)

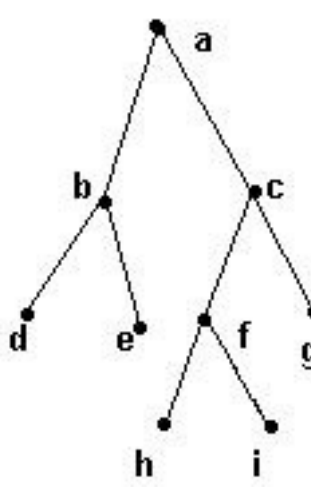
13

Tam İkili Ağaç (Full binary tree)



Bir *full* binary tree 'de uç düğümler hariç her düğüm iki çocuğa sahiptir.

Tam ikili ağaç (full binary tree)'da :



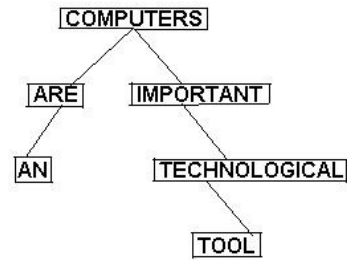
- * t adet uç düğüm varsa ağacın yüksekliği $\lg(t) \leq h$
- * i adet ara düğümü varsa $(i+1)$ adet uç düğümü vardır
- * i adet ara düğümü varsa toplam düğüm sayısı $(2i+1)$

■ Örnek: $k = 4$ internal vertices (a, b, c ve f) ve 5 terminal vertices (d, e, g, h and i) toplamda 9 düğüm

İkili Arama Ağaçları (Binary search trees)

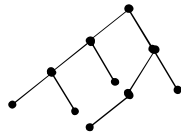
- Veri düğümlere yerleştirilir
- Veri alfabetik sırada ağaca yerleştirilir. Düğümün sol tarafındaki veri, o düğümdeki veriden daha küçüktür
- Ve düğümün sağ tarafındaki veri de o düğümdeki veriden daha büyüktür

Örnek: "Computers are an important technological tool"

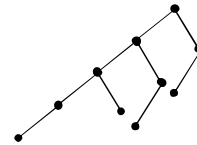


Dengeli Ağaç (Balanced Tree)

Yüksekliği h olan bir ağacın yapraklarının seviyesi h veya $h-1$ ise dengeli ağaç adını alır



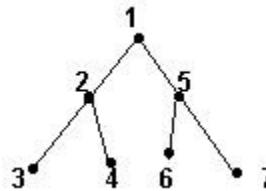
Dengeli ağaç



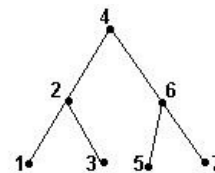
Dengesiz ağaç

Ağacın Düğümlerinin Listesi (Tree Traversals)

- 1: Pre-order traversal (önden sıralı)

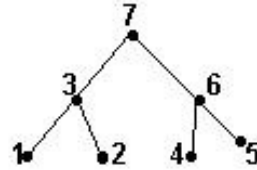


- 2: In-order traversal (içten sıralı)

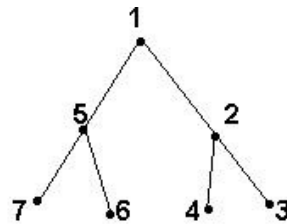


devam....

- 3: Post-order traversal
(sondan sıralı)

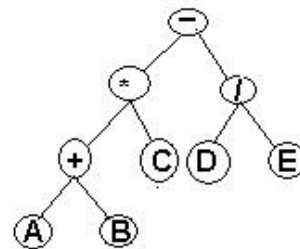


- 4: Reverse post-order traversal (ters önden sıralı)

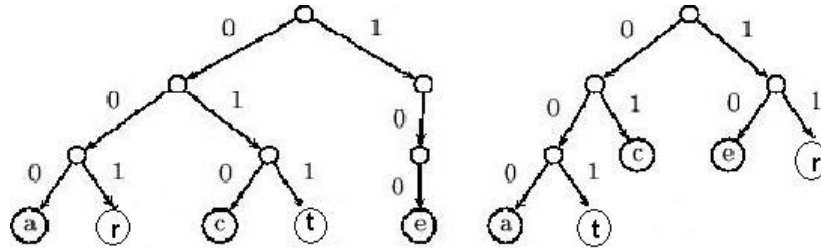


Aritmetik İfadeler

- Standart: *infix* formu
 $(A+B) * C - D / E$
- Tüm parentezli form (in-order & parenthesis):
 $((A + B) * C) - (D / E)$
- Postfix* formu (Polish notasyonunun tersi):
 $A B + C * D E / -$
- Prefix* formu (Polish notasyonu):
 $- * + A B C / D E$



Huffman Kodları (Huffman Codes)



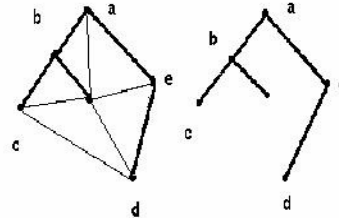
- Sol taraftaki ağacı kullanarak **rate** kelimesini kodlarsak
001 000 011 100
- Sağ taraftaki ağacı kullanarak **rate** kelimesini kodlarsak
11 000 001 10

Spanning Trees

Verilen **G** grafında, **T** bir *spanning tree* ise;

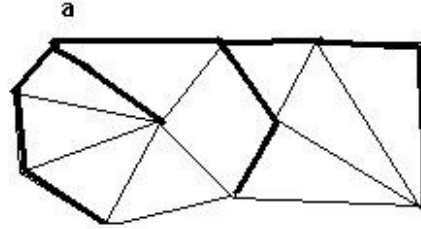
T,

- **G** grafının bir *subtree*'sidir
- ve
- **G** grafının bütün düğümlerini içerir

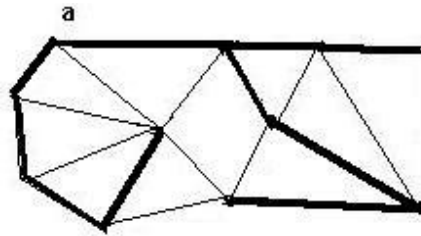


Spanning tree oluřturma yöntemleri

- Breadth-first search method



- Depth-first search method (backtracking)



Minimal spanning trees

Verilen ağırlıklı bir **G** grafının

Minimal spanning tree'si

- **G** grafının spanning tree 'si olup (tüm düğümlerden geçilmiş)
- Ağırlıklar toplamıda minimumdur

