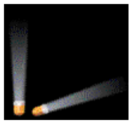
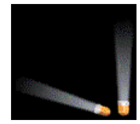
Balanced Binary Search Trees





- height is O(log n), where n is the number of elements in the tree
- AVL (Adelson-Velsky and Landis) trees
- red-black trees
- get, put, and remove take O(log n) time

Balanced Binary Search Trees

- Indexed AVL trees
- Indexed red-black trees
- Indexed operations also take O(log n) time

Balanced Search Trees

- weight balanced binary search trees
- 2-3 & 2-3-4 trees
- AA trees
- B-trees
- BBST
- etc.

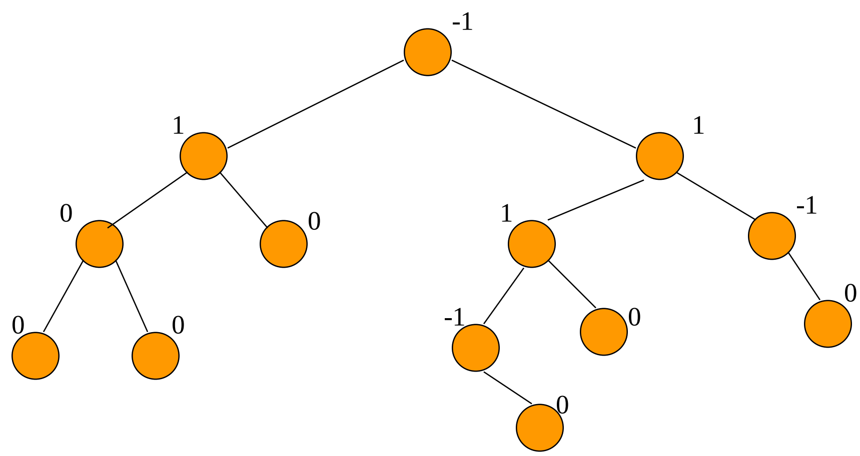
AVL Tree

- binary tree
- for every node x, define its balance factor

  balance factor of x = height of left subtree of x

  - height of right subtree of x
- balance factor of every node x is -1, 0, or 1

Balance Factors



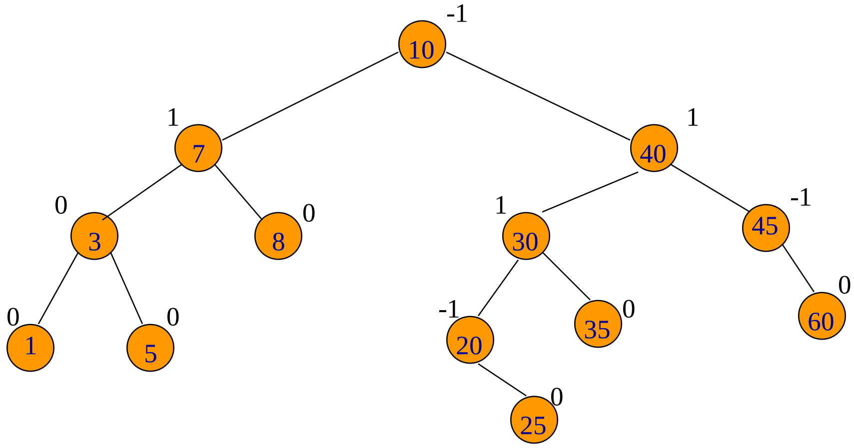
This is an AVL tree.

Height

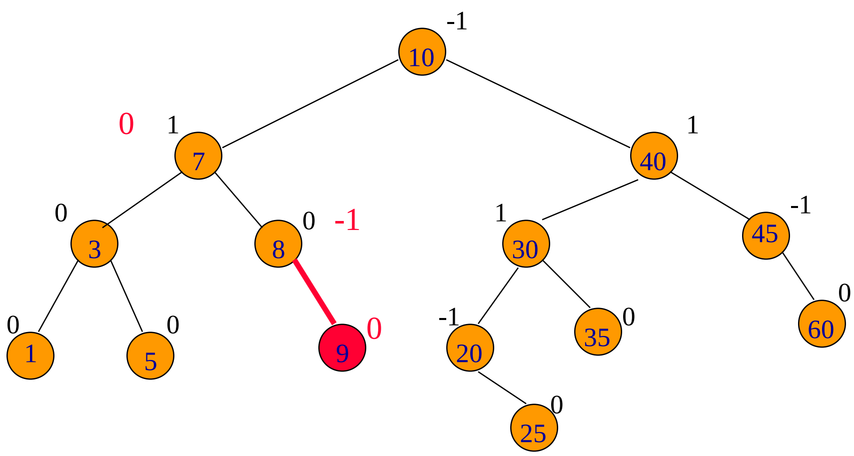
The height of an AVL tree that has n nodes is at most $1.44 \log_2 (n+2)$.

The height of every n node binary tree is at least $\log_2 (n+1)$.

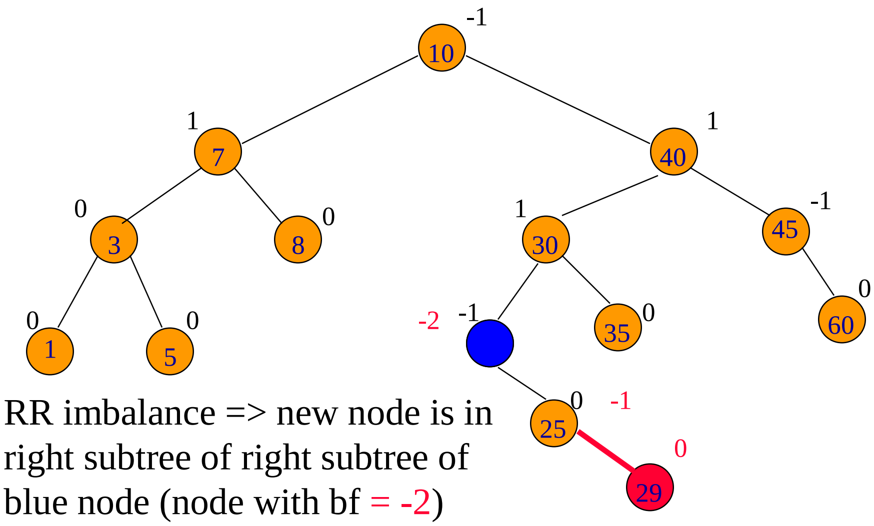
AVL Search Tree



put(9)

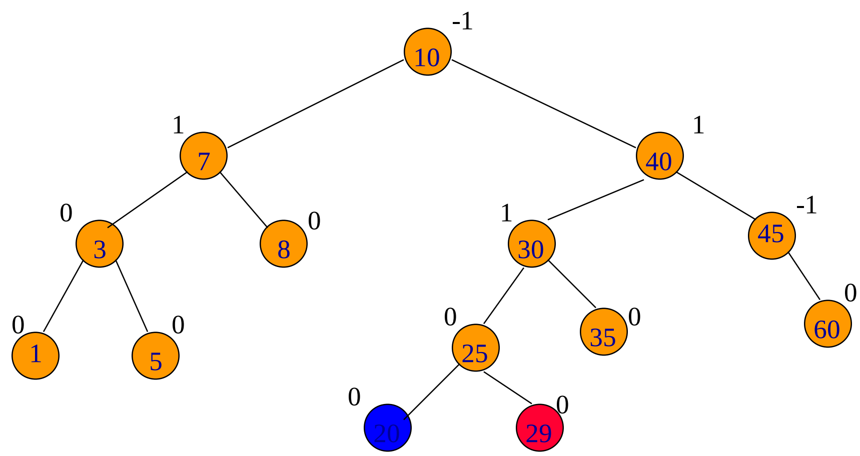


put(29)



RR imbalance => new node is in right subtree of right subtree of blue node (node with bf = -2)

put(29)



RR rotation.

AVL Rotations

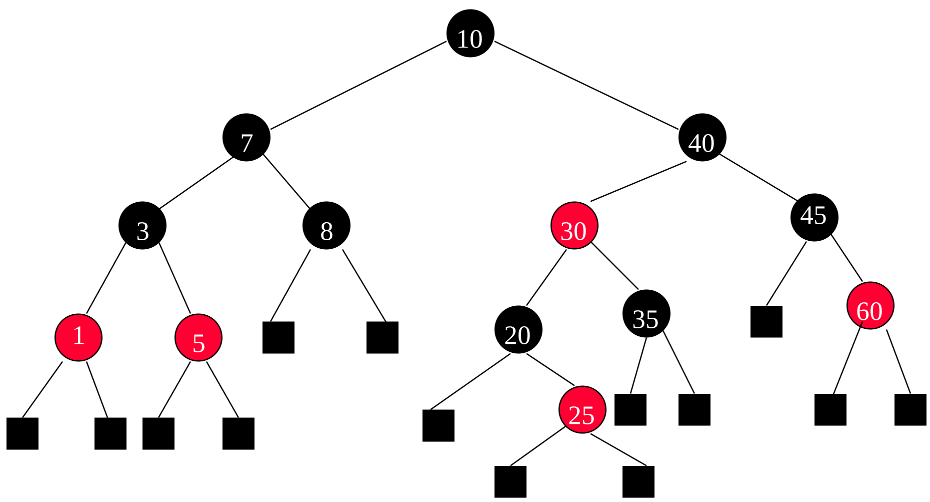
- RR
- LL
- RL
- LR

Red Black Trees

Colored Nodes Definition

- Binary search tree.
- Each node is colored red or black.
- Root and all external nodes are black.
- No root-to-external-node path has two consecutive red nodes.
- All root-to-external-node paths have the same number of black nodes

Example Red Black Tree

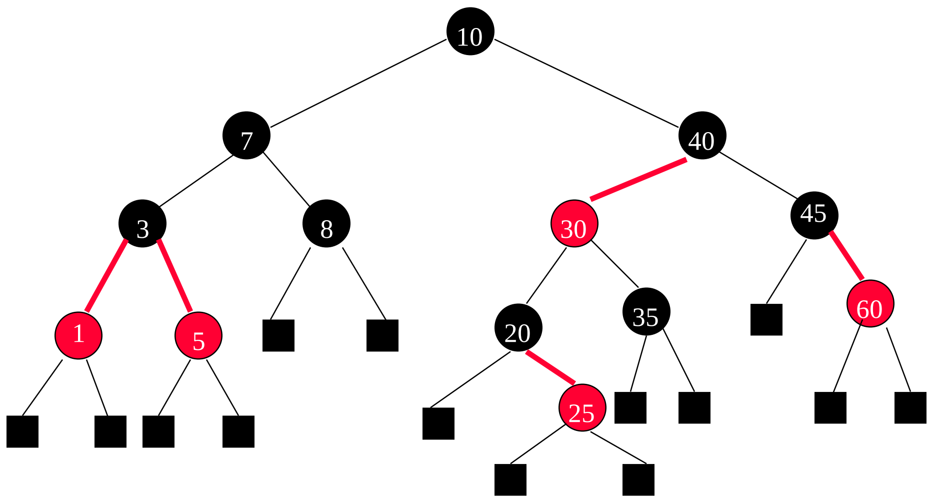


Red Black Trees

Colored Edges Definition

- Binary search tree.
- Child pointers are colored red or black.
- Pointer to an external node is black.
- No root to external node path has two consecutive red pointers.
- Every root to external node path has the same number of black pointers.

Example Red Black Tree



Red Black Tree

- The height of a red black tree that has n (internal) nodes is between $\log_2(n+1)$ and $2\log_2(n+1)$.
- java.util.TreeMap => red black tree

Graphs

- G = (V,E)
- V is the vertex set.
- Vertices are also called nodes and points.
- E is the edge set.
- Each edge connects two different vertices.
- Edges are also called arcs and lines.
- Directed edge has an orientation (u,v).

  u ⟶ v

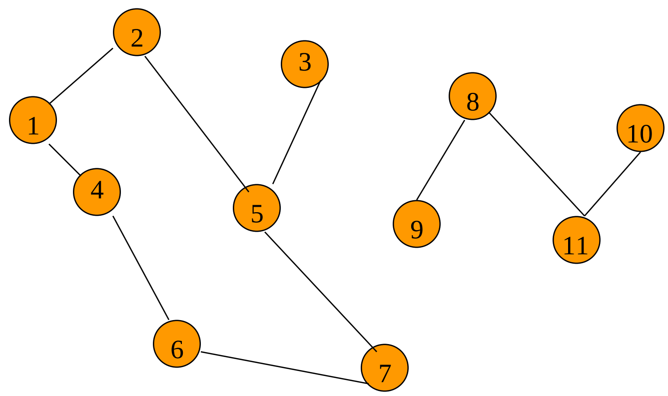
Graphs

- Undirected edge has no orientation (u,v).

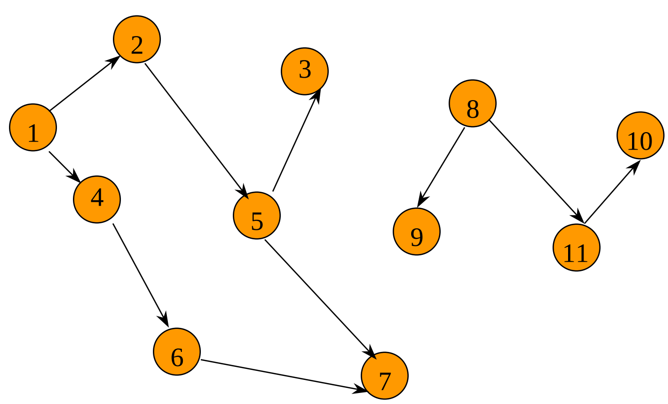
  u —— v

- Undirected graph => no oriented edge.
- Directed graph => every edge has an orientation.

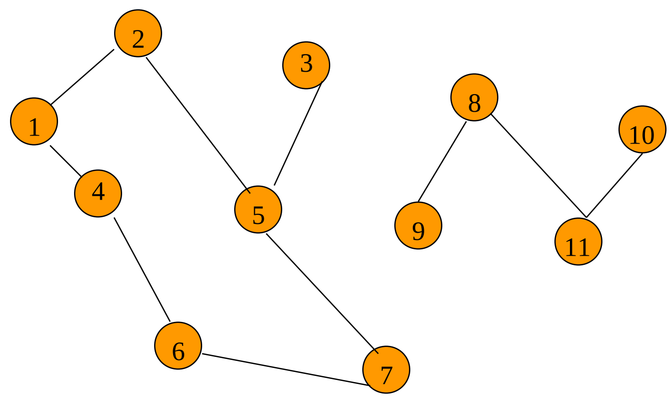
Undirected Graph



Directed Graph (Digraph)

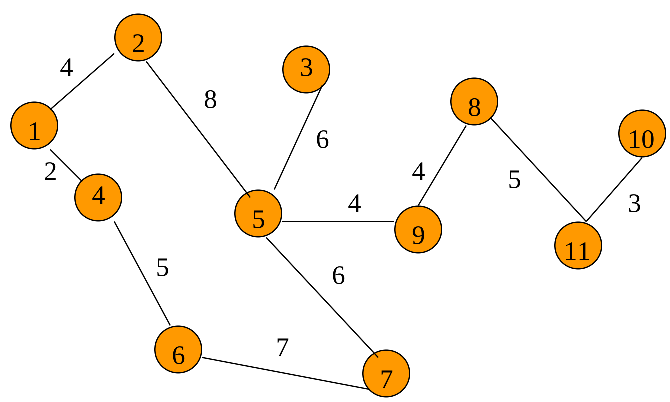


Applications—Communication Network



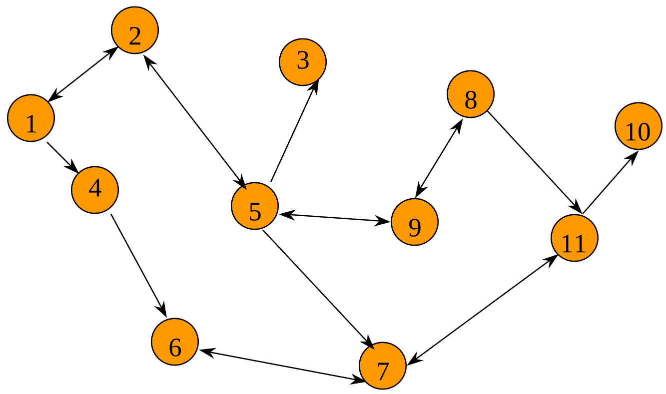
- Vertex = city, edge = communication link.

Driving Distance/Time Map



- Vertex = city, edge weight = driving distance/time.

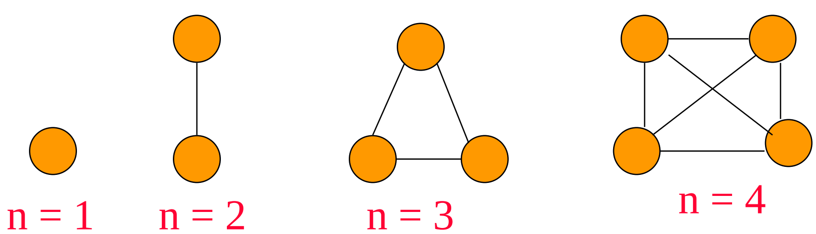
Street Map



- Some streets are one way.

Complete Undirected Graph

Has all possible edges.



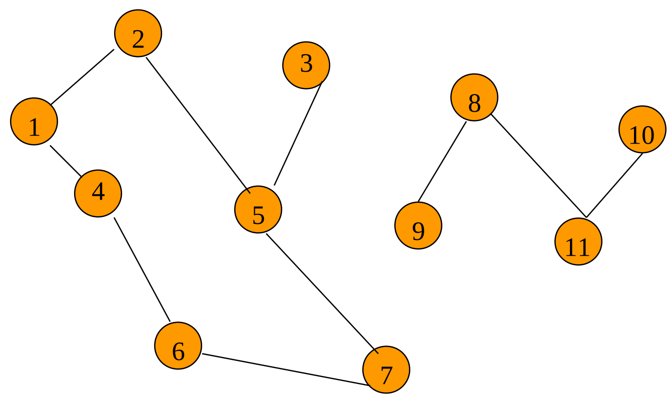
Number Of Edges—Undirected Graph

- Each edge is of the form (u,v), u != v.
- Number of such pairs in an n vertex graph is n(n-1).
- Since edge (u,v) is the same as edge (v,u), the number of edges in a complete undirected graph is n(n-1)/2.
- Number of edges in an undirected graph is <= n(n-1)/2.

Number Of Edges—Directed Graph

- Each edge is of the form (u,v), u != v.
- Number of such pairs in an n vertex graph is n(n-1).
- Since edge (u,v) is not the same as edge (v,u), the number of edges in a complete directed graph is n(n-1).
- Number of edges in a directed graph is <= n(n-1).

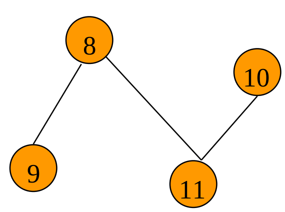
Vertex Degree



Number of edges incident to vertex.

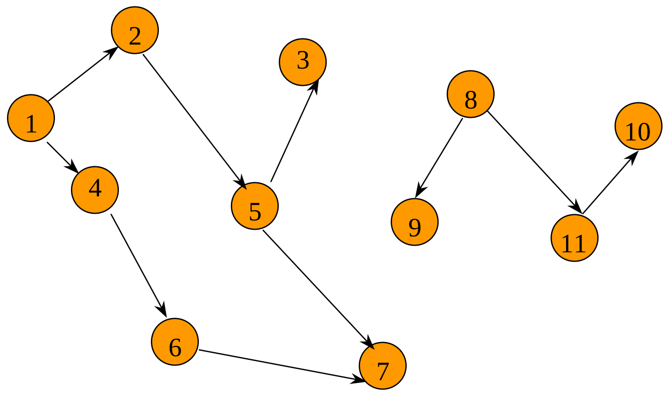
degree(2) = 2, degree(5) = 3, degree(3) = 1

Sum Of Vertex Degrees



Sum of degrees = 2e (e is number of edges)

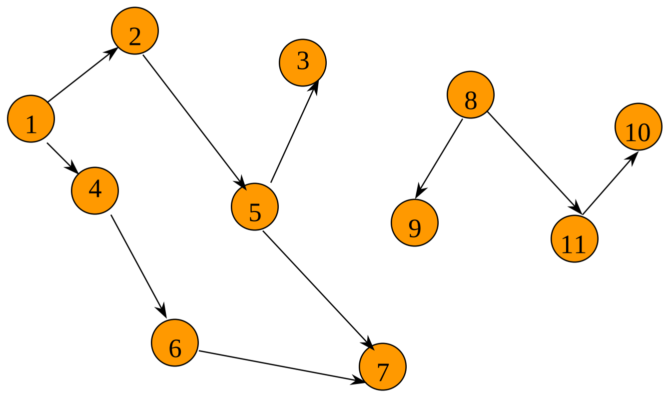
In-Degree Of A Vertex



in-degree is number of incoming edges

indegree(2) = 1, indegree(8) = 0

Out-Degree Of A Vertex



out-degree is number of outbound edges

outdegree(2) = 1, outdegree(8) = 2

Sum Of In- And Out-Degrees

each edge contributes 1 to the in-degree of some vertex and 1 to the out-degree of some other vertex

sum of in-degrees = sum of out-degrees = e,

where e is the number of edges in the digraph