Chapter 12

Abstract Data Type

OBJECTIVES

After reading this chapter, the reader should be able to:

- Understand the concept of an abstract data type (ADT).
- Understand the concept of a linear list as well as its operations and applications.
- Understand the concept of a stack as well as its operations and applications.
- Understand the concept of a queue as well as its operations and applications.
- Understand the concept of a tree as well as its operations and applications.
- Understand the concept of a graph as well as its operations and applications.

12.1

BACKGROUND

Note:

The concept of abstraction means:
1. You know what a data type can do.
2. How it is done is hidden.

Brooks/Cole
Thomson Learning

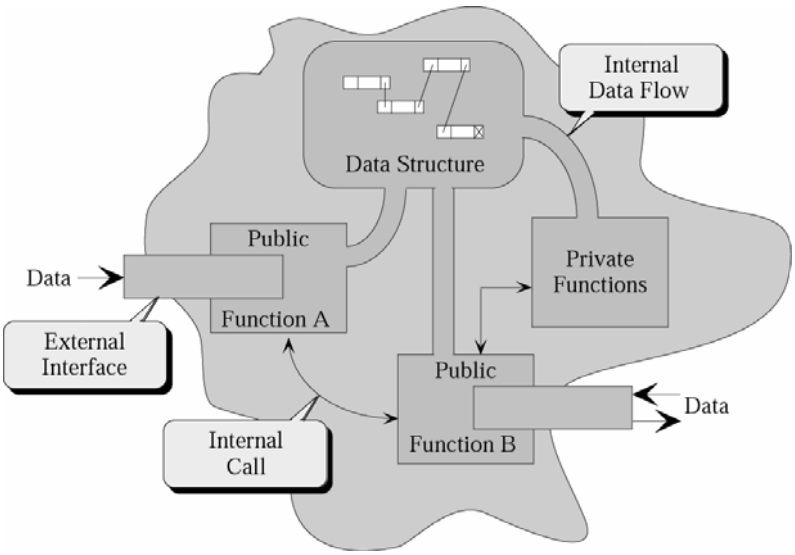
©Brooks/Cole, 2003

Abstract Data Type

1. ***Declaration of data***
2. ***Declaration of operations***
3. ***Encapsulation of data and operations***

Figure 12-1

Model for ADT



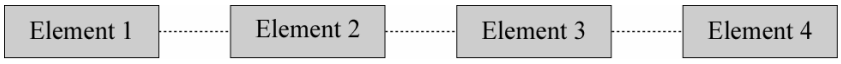
12.2

LINEAR LISTS

Brooks/Cole Thomson Learning

Figure 12-2

Linear list

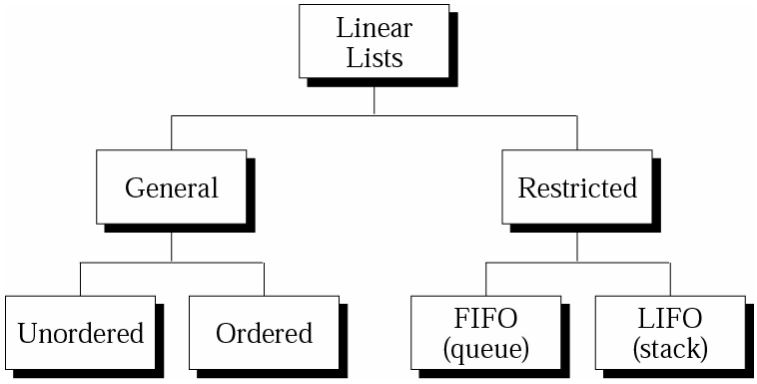


Brooks/Cole Thomson Learning

©Brooks/Cole, 2003

Figure 12-3

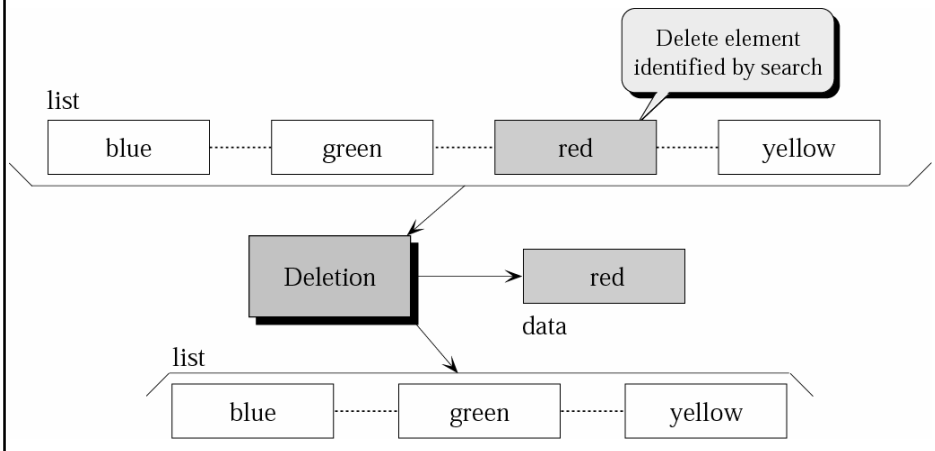
Categories of linear lists



Brooks/Cole Thomson Learning

Figure 12-5

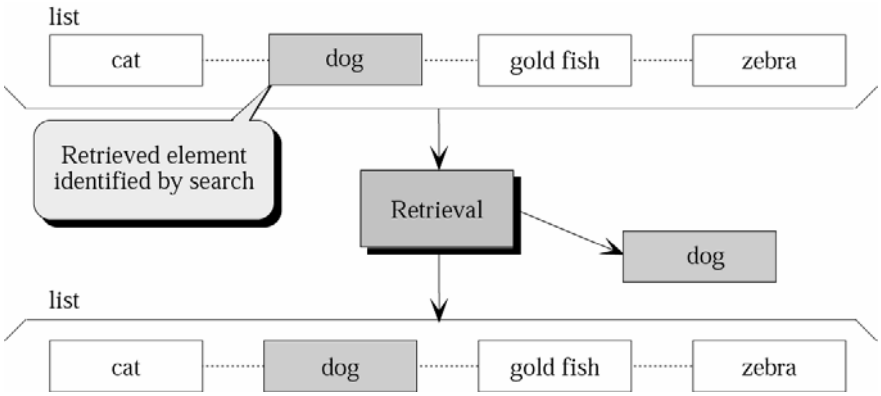
Deletion from a linear list



Brooks/Cole Thomson Learning

Figure 12-6

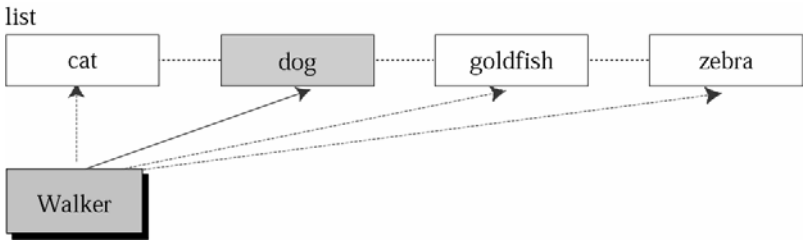
Retrieval from a linear list



Brooks/Cole Thomson Learning

Figure 12-7

Traversal of a linear list



Brooks/Cole Thomson Learning

12.3 STACKS

Figure 12-8

Three representations of a stack

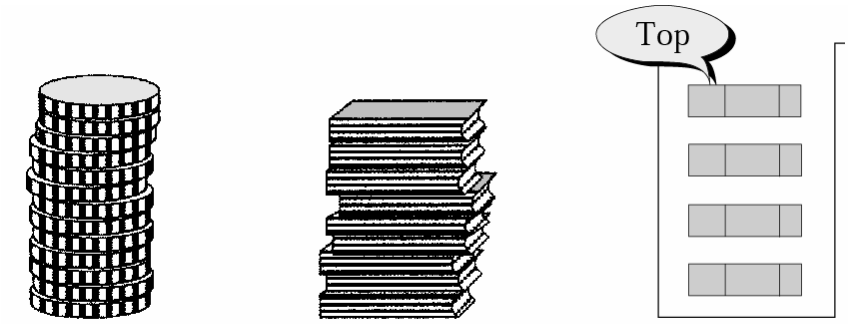
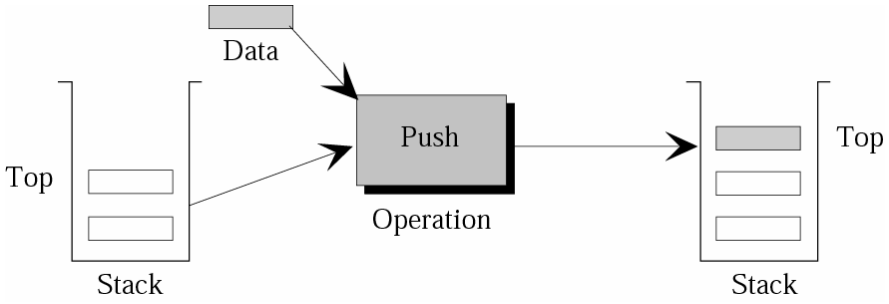


Figure 12-9

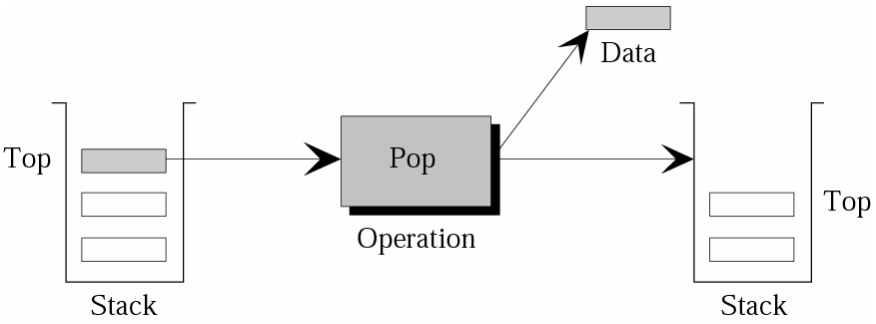
Push operation in a stack



Brooks/Cole Thomson Learning

Figure 12-10

Pop operation in a stack



Brooks/Cole Thomson Learning

Example 1

Show the result of the following operations on a stack S.

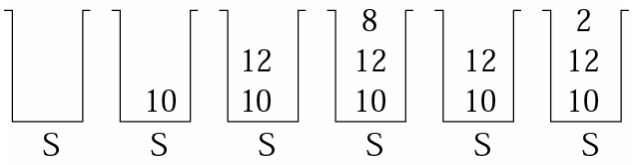
push (S , 10)
push (S , 12)
push (S , 8)
if not empty (S), then pop (S)
push (S , 2)

Solution

See Figure 12.11 (next slide).

Figure 12-11

Example 1

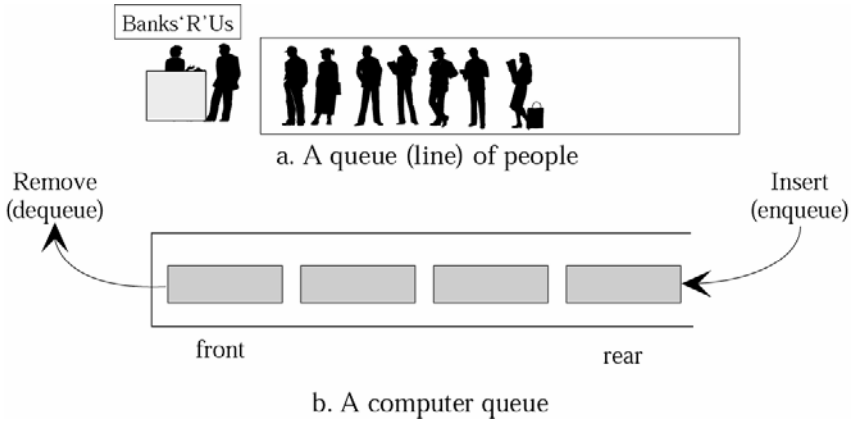


12.4 QUEUES

Brooks/Cole Thomson Learning

Figure 12-12

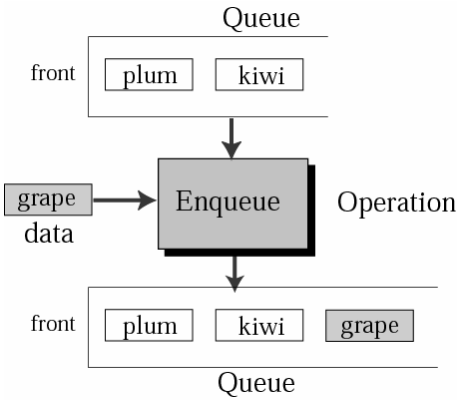
Queue representation



Brooks/Cole Thomson Learning

Figure 12-13

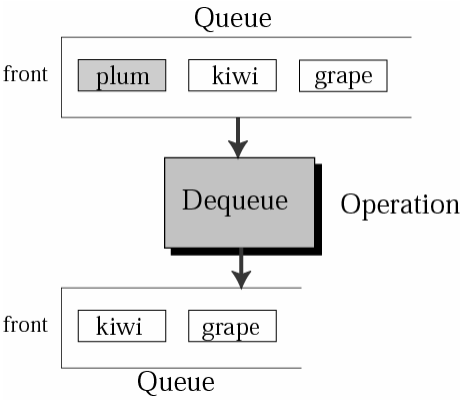
Enqueue operation



Brooks/Cole Thomson Learning

Figure 12-14

Dequeue operation



Brooks/Cole Thomson Learning

Example 2

Show the result of the following operations on a queue Q.

enqueue (Q , 23)
if not empty (Q), dequeue (Q)
enqueue (Q , 20)
enqueue (Q , 19)
if not empty (Q), dequeue (Q)

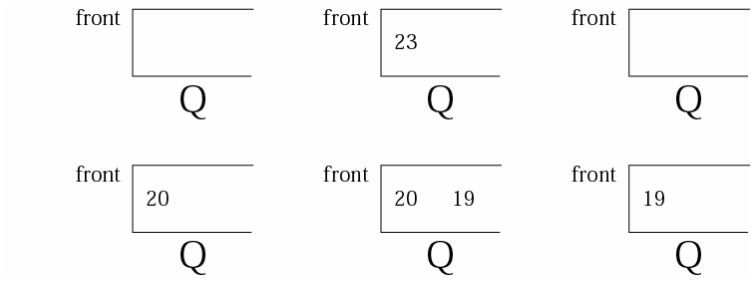
Solution

See Figure 12.15 (next slide).

Brooks/Cole Thomson Learning

Figure 12-15

Example 2



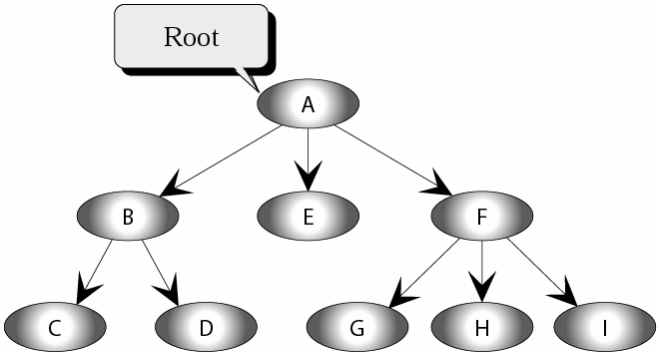
Brooks/Cole Thomson Learning

12.5 TREES

Brooks/Cole Thomson Learning

Figure 12-16

Representation of a tree

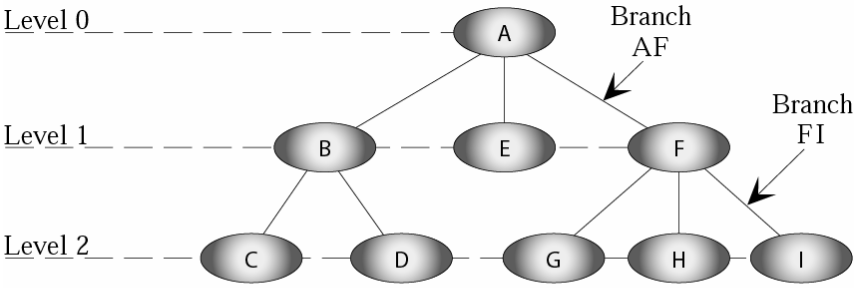


Brooks/Cole Thomson Learning

©Brooks/Cole, 2003

Figure 12-17

Tree terminology

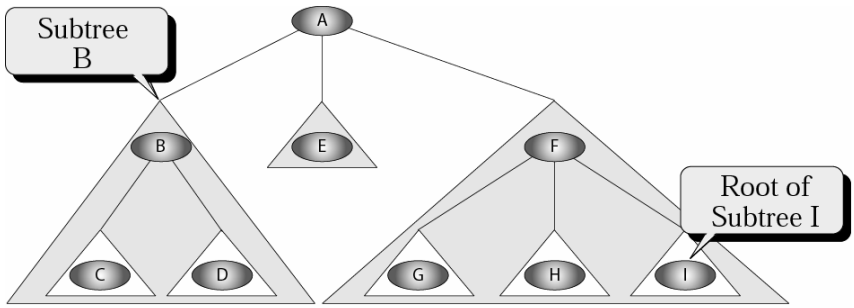


Parents:	A, B, F	Leaves	C, D, E, G, H, I
Children:	B, E, F, C, D, G, H, I	Internal nodes	B, F
Siblings:	{B,E,F}, {C,D}, {G,H,I}		

Brooks/Cole
Thomson Learning

Figure 12-18

Subtrees



Brooks/Cole
Thomson Learning

12.6 BINARY TREES

Figure 12-19

Binary tree

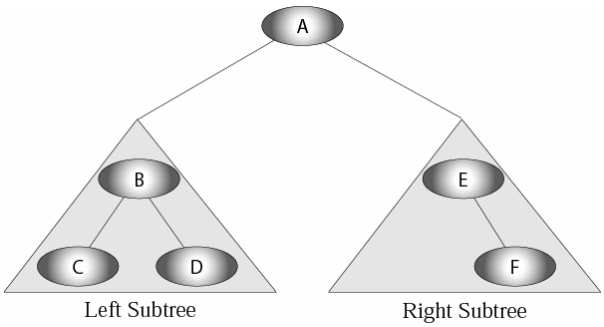
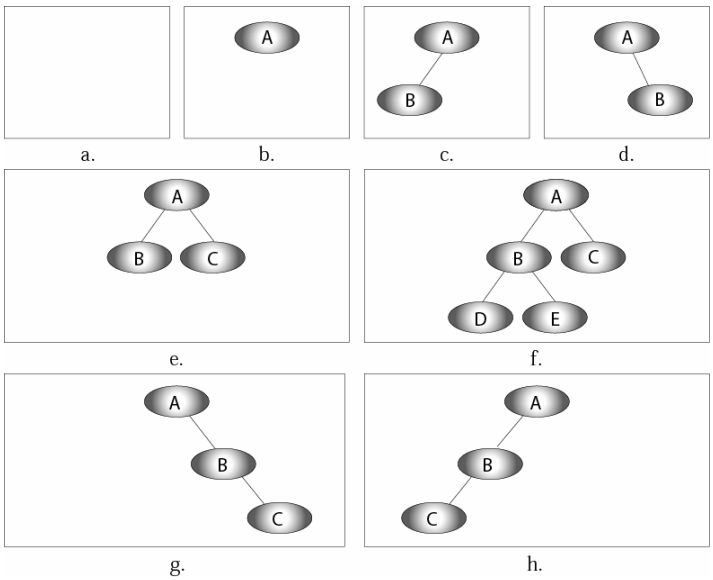


Figure 12-20

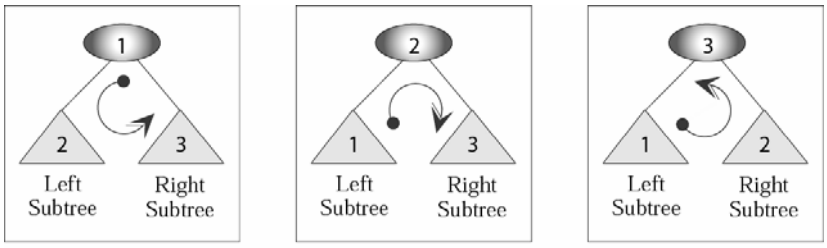
Examples of binary trees



Brooks/Cole Thomson Learning

Figure 12-21

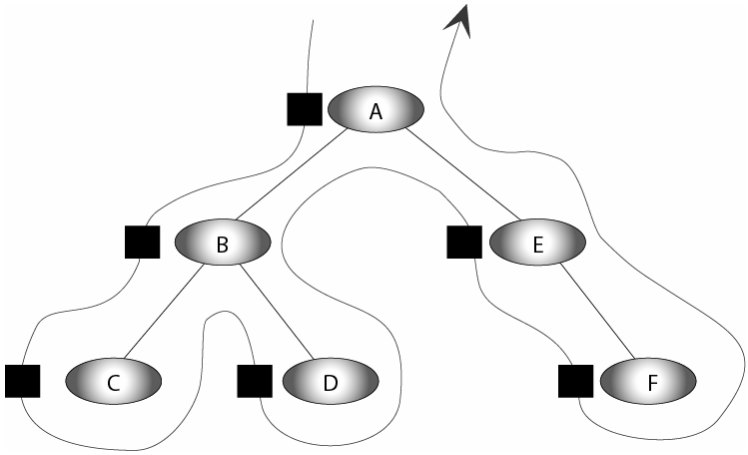
Depth-first traversal of a binary tree



Brooks/Cole Thomson Learning

Figure 12-22

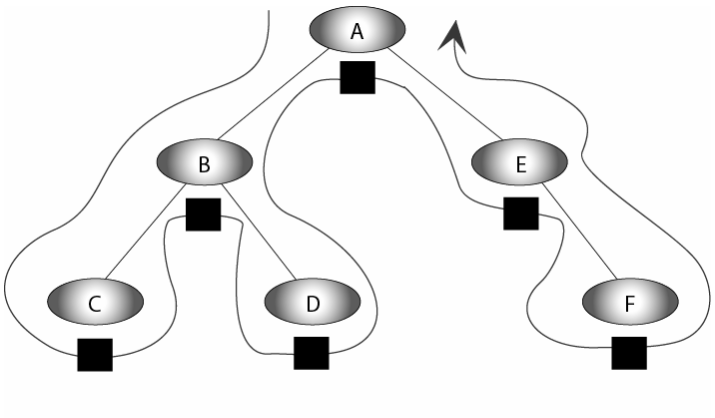
Preorder traversal of a binary tree



Brooks/Cole
Thomson Learning

Figure 12-23

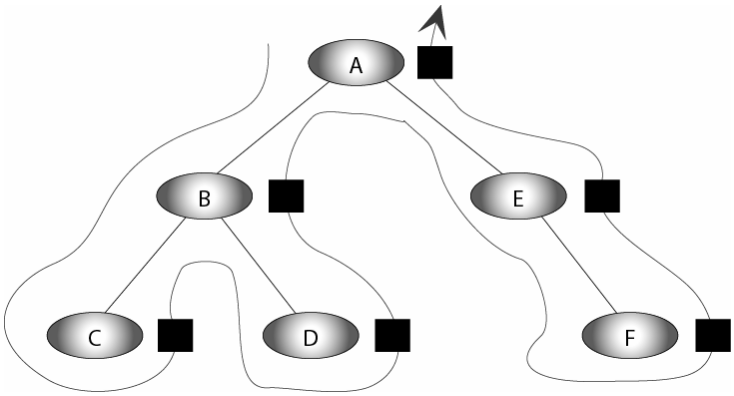
Inorder traversal of a binary tree



Brooks/Cole
Thomson Learning

Figure 12-24

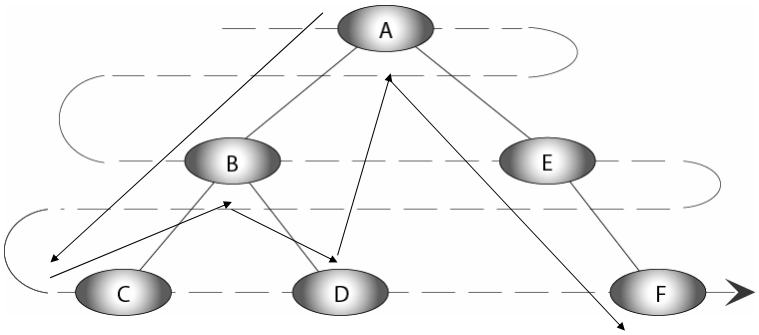
Postorder traversal of a binary tree



Brooks/Cole
Thomson Learning

Figure 12-25

Breadth-first traversal & Depth-first traversal of a binary tree

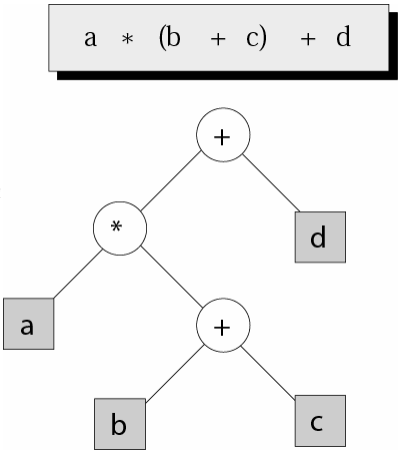


Brooks/Cole
Thomson Learning

Figure 12-26

Expression tree

Leaf node: operand
Internal: operators
Subtree: subexpression
Use inorder traversal



Brooks/Cole Thomson Learning

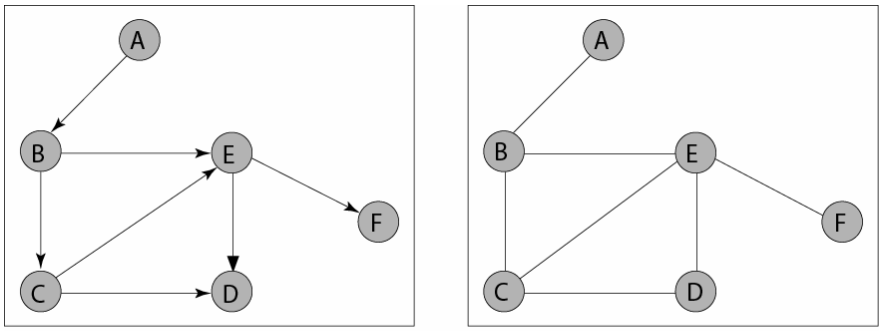
12.7 GRAPHS

Brooks/Cole Thomson Learning

Figure 12-27

Directed and undirected graphs

Graph: G={V,E} V: vertices, E: edges



a. Directed Graph b. Undirected Graph

Figure 12-33

Depth-first traversal of a graph

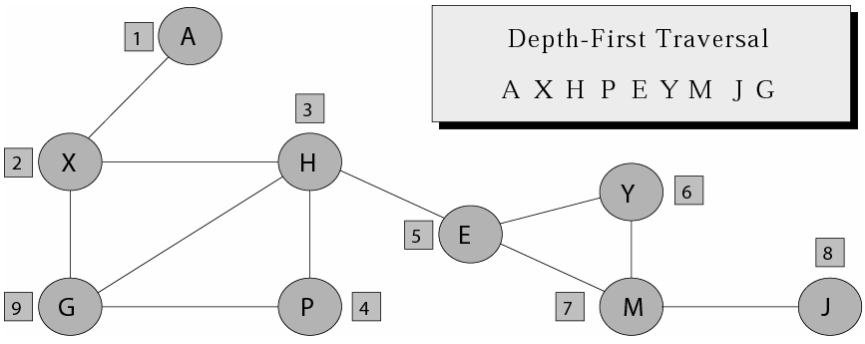
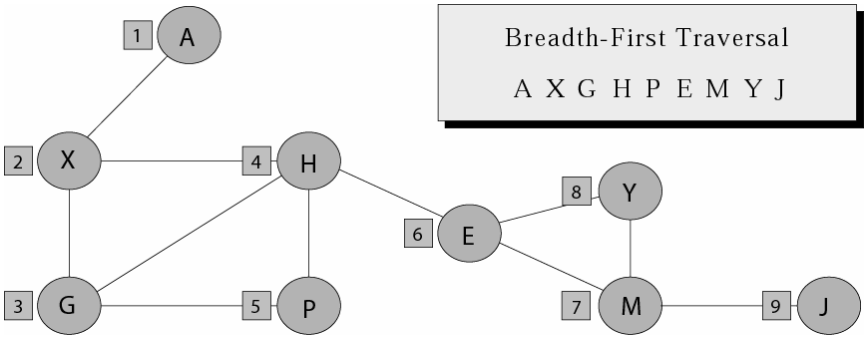


Figure 12-34

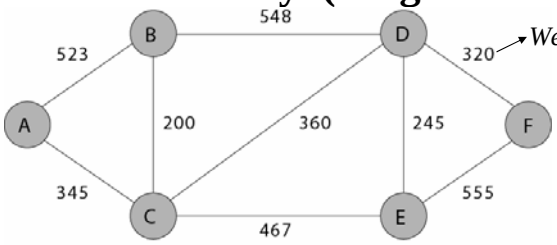
Breadth-first traversal of a graph



Brooks/Cole Thomson Learning

Figure 12-35: Part I

Graph implementations
Use array (#edges is large)



	A	B	C	D	E	F
A	0	523	345	0	0	0
B	523	0	200	548	0	0
C	345	200	0	360	467	0
D	0	548	360	0	245	320
E	0	0	467	245	0	555
F	0	0	0	320	555	0

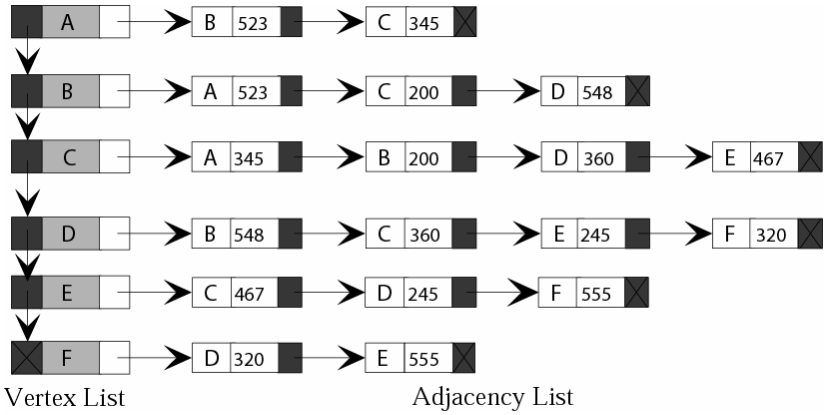
Vertex Array

Adjacency Matrix

Brooks/Cole Thomson Learning

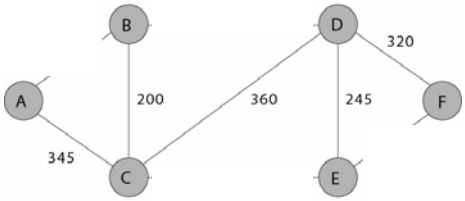
Figure 12-35: Part 2

Graph implementations
Link list (#edges is large)



Brooks/Cole Thomson Learning

Minimum Spanning Tree



Spanning tree: A tree that contains all the vertices in the graph.
Min. spanning tree: A spanning tree with min. weight.
Applications:
Example:The shortest length of cable to connect all computers.

Brooks/Cole Thomson Learning