Eric Yang

304263623

1. a. 50

20 60

10 40 70

30 45 65 80

25 35 75

b. 50

10 60

40 70

25 45 65 80

35 75

c. in-order: 10,20,25,30,35,40,45,50,60,65,70,75,80

pre-order: 50,20,10,40,30,25,35,45,60,70,65,80,75

post-order: 10,25,35,30,45,40,20,65,75,80,70,60,50

1. a. 6

3 5

1 2 4

b. array: [6,3,5,1,2,4] size: 6

c. array: [5,3,4,1,2] size: 5

1. a.

struct Node

{

int value;

Node \*parent, \*leftChild, \*rightChild;

}

b.

allocate a new Node with the value

if the tree is empty

point the root pointer to the new Node

return

create a current Node pointer at the root of the tree

while the Node hasn’t been inserted

if the value of the new Node is equal to the current Node’s value

delete the new Node

return

if the value of the new Node is less than the current Node’s value

if there is a left child, go to the left

else

set the current Node’s left pointer to the new Node

set the new Node’s parent pointer to the current Node

return

if the value of the new Node is greater than the current Node’s value

if there is a right child, go to the right

else

set the current Node’s right pointer to the new Node

set the new Node’s parent pointer to the current Node

return

1. a.

Adjacency Matrix

to

A B C D E F G H I

A 0 1 1 0 0 0 0 0 0

B 0 0 0 0 0 0 0 1 0

C 0 0 0 1 1 0 0 0 0

D 0 1 0 0 0 0 0 1 0

from E 0 0 0 0 0 0 1 0 0

F 0 0 0 0 0 0 1 0 1

G 0 0 1 0 0 0 0 1 0

H 0 0 0 0 0 0 1 0 0

I 0 0 1 0 0 0 0 0 0

0 = false

1 = true

Adjacency List

A🡪B C

B🡪H

C🡪D E

D🡪B H

E🡪G

F🡪G I

G🡪C H

H🡪G

I🡪C

b.

E🡪G🡪H🡪C🡪D🡪B

E🡪G🡪C🡪D🡪H🡪B

E🡪G🡪C🡪D🡪B🡪H