1.

a.

25

30

15

10

60

40

20

50

70

80

65

78

32

b. Inorder: 10 15 20 25 30 32 40 50 60 65 70 78 80

Preorder: 50 20 10 15 40 30 25 32 60 70 65 80 78

Postorder: 15 10 25 32 30 40 20 65 78 80 70 60 50

c. After deleting the node 30 then deleting the node 20:

15

10

60

40

50

25

70

32

80

65

78

2.

a. struct Node

{

Node\* parent\_pointer;

Node\* left\_child;

Node\* right\_child;

int data;

}

b.

if the tree is empty

point the root pointer to the inserting node, then return

create current pointer points to root

repeatedly

if the node’s value is equal to current node’s value, return

if the node’s value is less than current node’s value

if left\_child is nullptr, point it to the inserting node and point the inserting node’s parent pointer to current node, return

else point current node to the left\_child

if the node’s value is greater than current node’s value

if right\_child is nullptr, point it to the inserting node and point the inserting node’s parent pointer to current node, return

else point current node to the right\_child

7

3. a.

2

0

4

6

5

b. array: first line: index, second line: value.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 |
| 7 | 5 | 6 | 4 | 0 | 2 |

c.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 |
| 6 | 5 | 2 | 4 | 0 |

4. a. C + S

b. logC + S

c. logS + logC

d. logS

e. 1

f. logC + S

g. SlogS

h. ClogS