1 a) 50

20 60

10 40 70

15 30 65 80

25 35 75

1 b) 50

25 60

10 40 70

15 35 65 80

75

1 c)

In-order Traversal: 10 15 20 25 30 35 40 50 60 65 70 75 80

Pre-order Traversal: 50 20 10 15 40 30 25 35 60 70 65 80 75

Post-order Traversal: 15 10 25 35 30 40 20 65 75 80 70 60 50

2 a) 7

3 5

1 2 4

2 b)

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

2 c)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 5 | 3 | 4 | 1 | 2 |

3 a)

struct Node

{

Node\* parent;

Node\* left;

Node\* right;

int m\_data;

};

3 b)

insert (Node\* node, Node\* base)

if base is nullptr, set node to the base and node’s parent to nullptr

else if node’s value is less than base’s value

if base has no left child

make base’s left pointer point to node and node’s parent point to base

else call insert, but pass in the base’s left child as the base parameter

else if node’s value is greater than base’s value

if base has no right child

make base’s right pointer point to node and node’s parent point to base

else call insert, but pass in the base’s right child as the base parameter

4

a) O(C + S)

b) O(logC  + S)

c) O(logC + logS)

d) O(logS)

e) O(1)

f ) O(logC + S)

g) O(SlogS)

h) O(C\*logS)