1. a 50

20 60

10 40 70

15 30 80

25 35 65 75

1.b After deleting 30

50

20 60

10 40 70

15 25 80

65 75

35

1.b After deleting 20

50

15 60

10 40 70

25 80

65 75

35

1. c Preorder: 50 20 10 15 40 30 25 35 60 70 65 80 75

Inorder: 10 15 20 25 30 35 40 50 60 65 70 75 80

Postorder: 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 |

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

2. c

3. a struct BSTNode

{

int value;

BSTNode \*left, \*right, \*parent;

};

3. b void insertBST(Node\* parent, int v)

if (parent is nullptr)

Create a new node and set parent to new node

Set parent's value to v

infinite for-loop until we insert the new node

if (v is equal to the parent's value)

return;

else if (v is less than the parent's value)

if (parent's left pointer is not nullptr)

recursively call insertBST passing in parent's left ptr

else //parent's left pointer is nullptr

Create a new node and set parent to new node

Set parent's value to v

return;

else if (v is greater than parent's value)

if (parent's left pointer is not nullptr)

recursively call insertBST, pass in parent's right ptr

else

Create a new node and set parent to new node

Set parent's value to v

return;

4.

a. O(C + S)

b. O(log C + S)

c. O(log C + log S)

d. O(log S)

e. O(1)

f. O(log C + S)

g. O(S log S) => Go through s students, insert them into a set, then print all

h. O(C + log S) => Should be O(C log S)