Exam Info

Topics

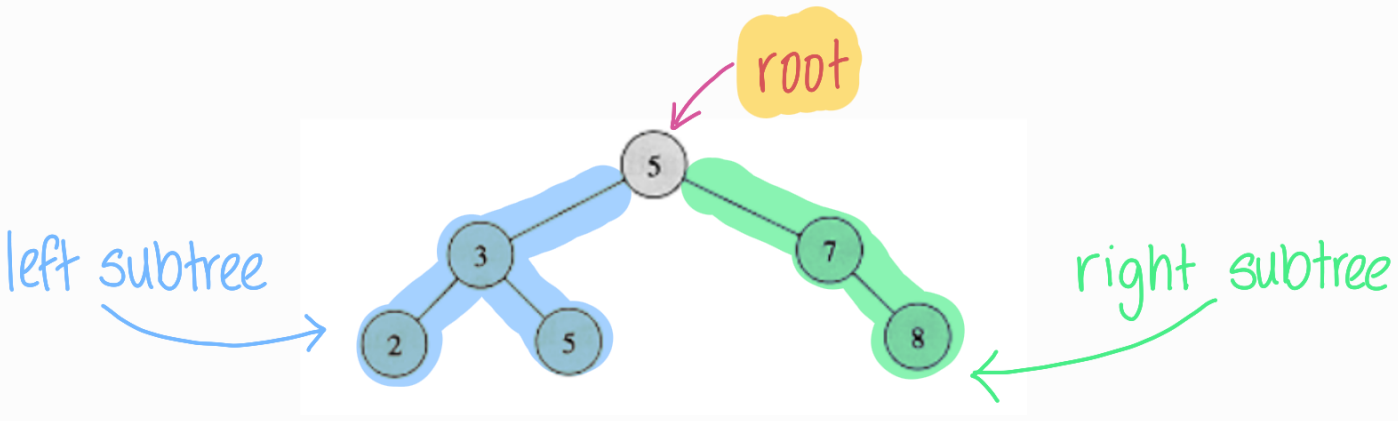
- ⇒ hash tables (1 question)
- ⇒ binary search trees (1 question)
- ⇒ red black trees (1 question)
- ⇒ graph theory (2 questions)

* two exact questions from the examples done in class will be on the final exam (graph theory)

⇒ the binary search tree question will most likely be composed of 3 parts

a) deleting element
b) inserting element
c) modifying pseudocode

Binary Search Trees



⇒ all the keys in the left subtree (2, 3, & 5) are smaller than 5

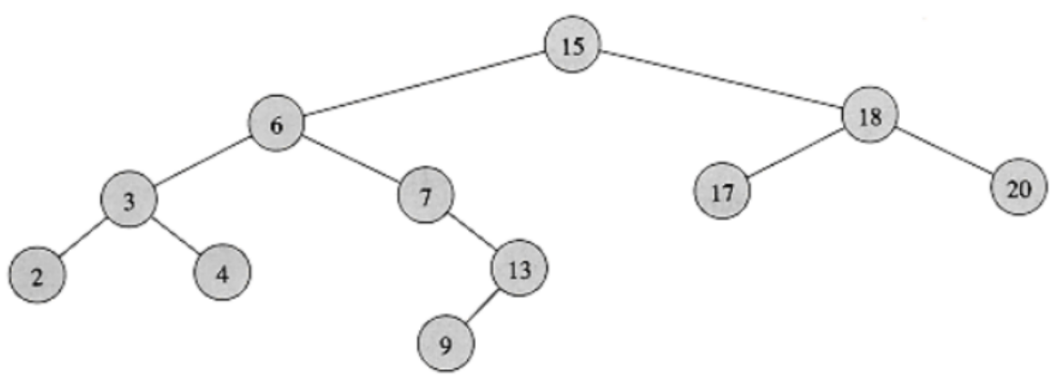
⇒ all the keys in the right subtree (7 & 8) are larger than 5

⇒ this property holds for every node in the tree; for the respective key, the smaller key is placed in the left subtree and the larger key is placed in the right subtree

⇒ 3 common Binary Search Tree operations

- searching
- insertion
- deletion

Searching



•To search for the key 13 in the tree, the path 15 => 6 => 7 => 13 is followed from the root

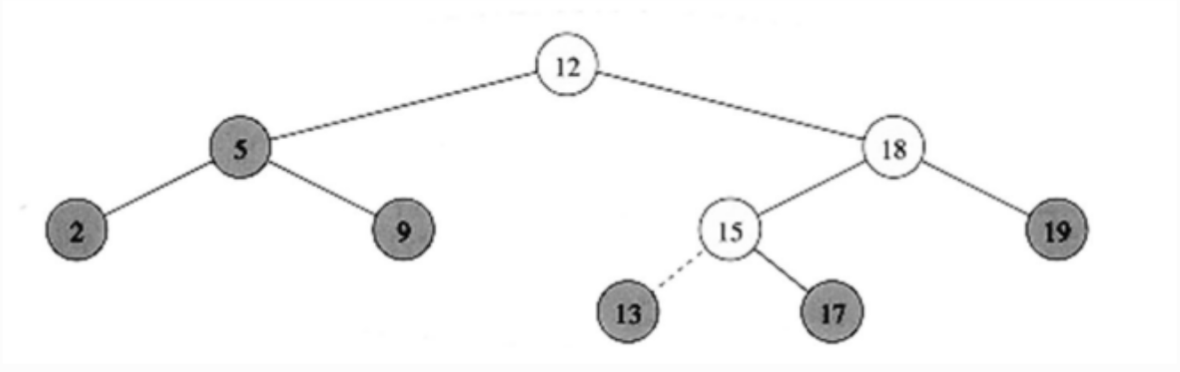
• The **minimum** key in the tree is 2, which can be found by following left pointers from the root

• The **maximum** key 20 is found by following right pointers from the root

• The **successor** of the node with key 15 is the node with key 17, since it is the minimum key in the right subtree of 15

Insertion

Inserting key 13



- The pointer x traces the path, and the pointer y is maintained as the parent of x
- After initialization, the **while** loop in lines 3-7 causes these two pointers to move down the tree, going left or right depending on the comparison of $key[z]$ with $key[x]$, until x is set to NIL
- This NIL occupies the position where we wish to place the input item z. Lines 8-13 set the pointers that cause z to be inserted

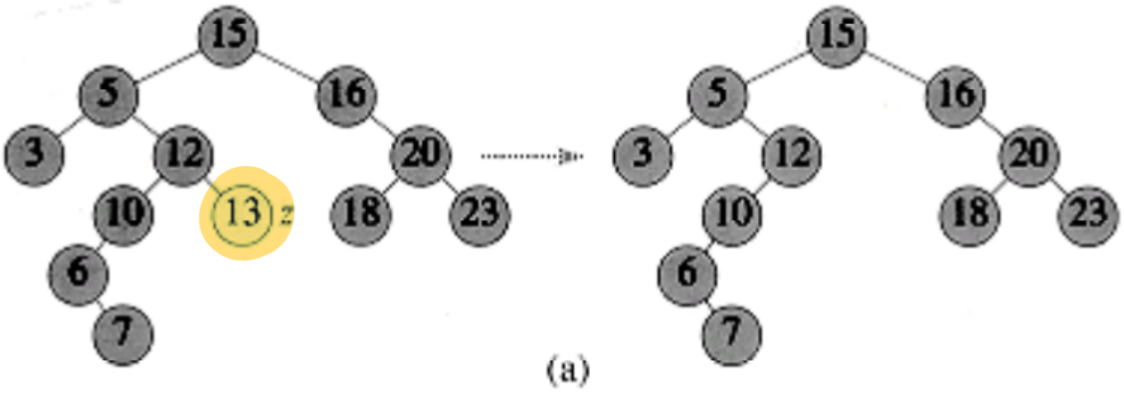
Pseudocode

```
TREE-INSERT(T,z)
1   y ← NIL
2   x ← root[T]
3   while x ≠ NIL
4   do y ← x
5 if key[z] < key[x]
6   then x ← left[x]
7   else x ← right[x]
8   p[z] ← y
9 if y = NIL
10      then root[T] ← z
11      else if key[z] < key[y]
12      then left[y] ← z
13      else right[y] ← z
```

⇒ memorize the pseudocode for Insertion ; may possibly be asked to modify the pseudocode on the final exam

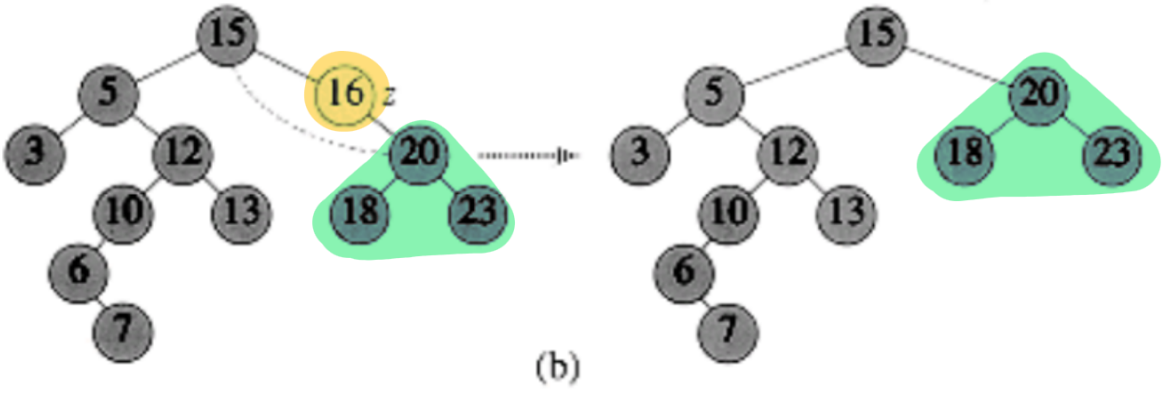
Deletion

- **Deletion**
  - Deleting a node z from a binary search tree. the node actually removed is lightly shaded
  - 1) If z has no children, we just remove it



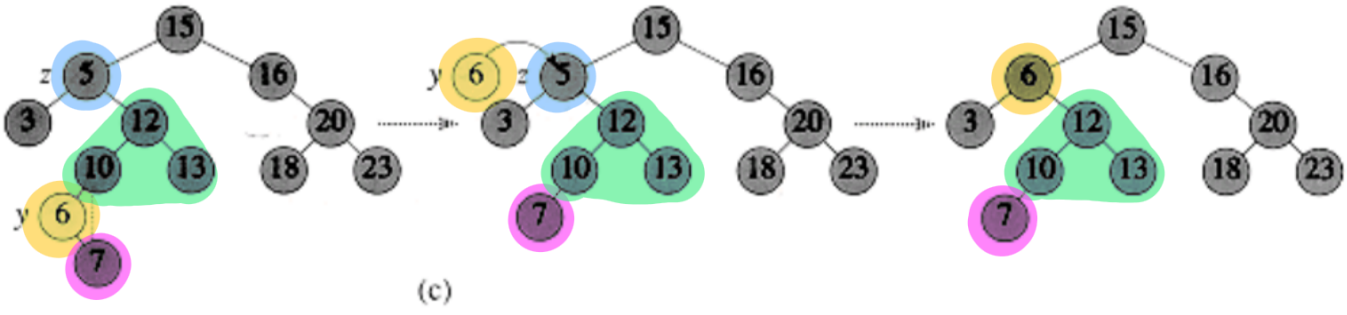
(a)

- **Deletion**
  - Deleting a node z from a binary search tree. the node actually removed is lightly shaded
  - 2) If z has only one child, we splice out z



(b)

- **Deletion**
  - Deleting a node z from a binary search tree. the node actually removed is lightly shaded
  - 3) If z has two children, we splice out its successor y, which has at most one child, and then replace the contents of z with the contents of y



(c)

⇒ Key 5 is being deleted

⇒ Key 5 is being replaced with key 6 because its the successor for key 5

⇒ the key whose numerical value is closest to the key being deleted and is on the same tree (left or right) from the root is selected to replace the position of the key being deleted

⇒ Key 7 replaces (the position of) key 6

Assignment

•Binary Search Tree

- Build a BST with the given array; 15, 5, 16, 3, 12, 20, 6, 13, 18, 23, 7