Lecture 12: Binary Search Trees

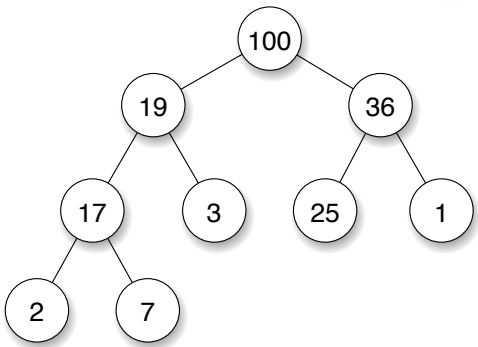
COSC242: Algorithms and Data Structures

Brendan McCane

Department of Computer Science, University of Otago

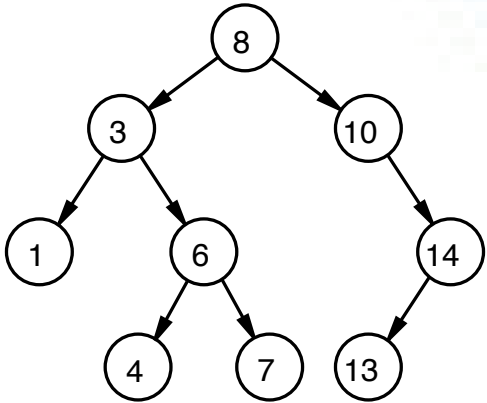
Binary Trees

In COSC241, you were introduced to a data structure called a heap where a tree node was usually bigger than all it's children. Most heaps used are binary trees (maximum of 2 children).



Binary search trees

A binary search tree (BST) is a binary tree such that all children to the left of a node are less than the node and all children to the right are greater than the node:



BSTs or hash tables?

Hash tables are convenient if maximum size of table is known beforehand, actual size does not fluctuate too much or spend too much time at a small fraction of the maximum size, and if the emphasis is on insertion and retrieval.

What if we need to do lots of deletions?

What if we need to do traversals, e.g. to print out items in order of increasing key values?

What if dynamic storage allocation is needed, because maximum tablesize is unknown or size fluctuates a lot?

BSTs are good for very dynamic problems, or if traversals are needed. But they come at a cost: insertion and deletion are $O(\log n)$ on average, and $O(n)$ in the worst case.

BST definition

A binary tree T is either

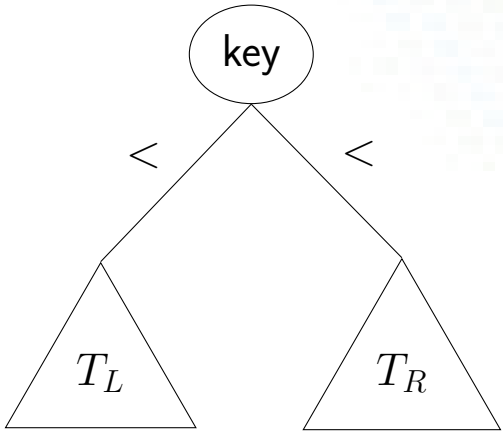
- the empty tree, or
- a root node containing a key field and data fields, a left subtree T_L, and a right subtree T_R.

Nodes with empty left and right subtrees are leaves.

A binary tree T has the BST-property if:

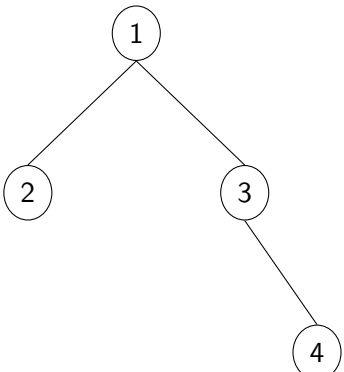
- nodes in T have a key field of ordinal type, so they can be ordered by $<$
- for each node N in T, N's key value is greater than all keys in its left subtree T_L and less than all keys in its right subtree T_R, and T_L and T_R are binary search trees.

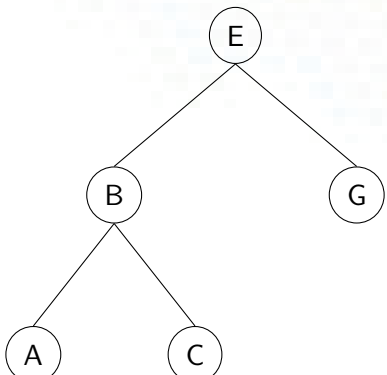
BST



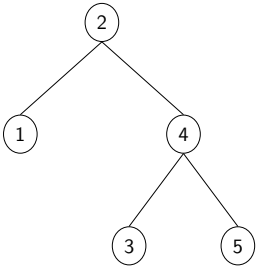
Examples and counterexamples

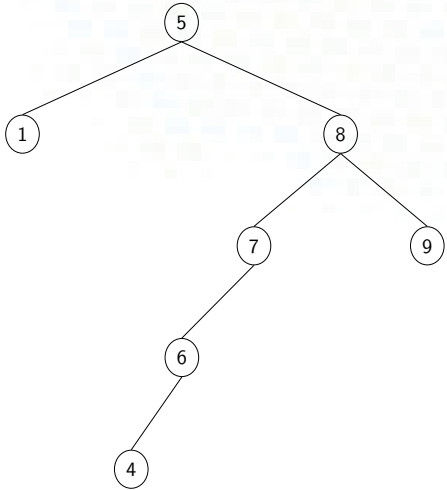
Which of these are BSTs?





More examples and counterexamples





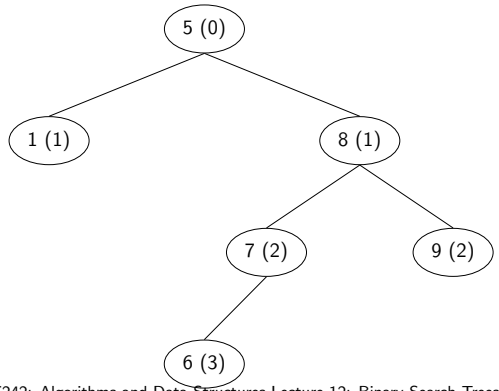
Definitions

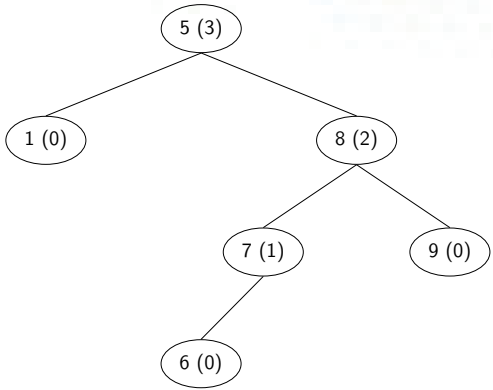
Definition (Depth of a node)

The depth of a node is the number of edges from that node to the root of the tree.

Definition (Height of a BST)

The height of a BST is the number of edges from the root to the deepest leaf.

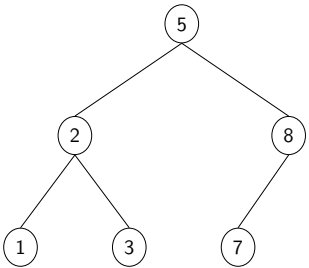




More definitions

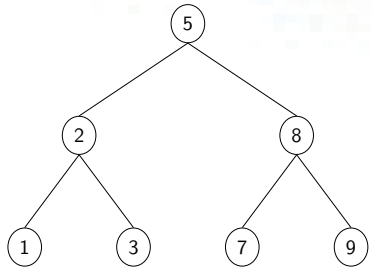
Definition (Complete BST)

A complete BST is a BST where each level, except possibly the last, is completely filled, and all nodes are as far left as possible.



Definition (Fully Complete BST)

A fully complete BST is a BST where each level is completely filled.



Properties

Theorem

The number of nodes in a fully complete binary tree of height h is $2^{h+1} - 1$.

Proof.

By induction - in class.

Corollary

The height of a complete tree of n nodes is $\lfloor \log_2 n \rfloor$.

C data structure

```
/* should live in bst.h */
typedef struct bst_node *BST;

/* should live in bst.c */
struct bst_node {
  KeyType key;
  BST left;
  BST right;
};
```

KeyType can be any comparable type (e.g. int, float, char, char *, etc).

Inserting

```
1:  function BST_INSERT(BST T, KeyType key)
2:      if T == NIL then
3:          return new bst_node(key)
4:      else
5:          if key < T→key then
6:              T→left = Insert(T→left, key)
7:          else
8:              T→right = Insert(T→right, key)
9:          end if
10:         return T
11:     end if
12: end function
```


Relevant parts of the textbook

Chapter 12 is all about binary search trees.

We're looking at things in a different order to the textbook. Insertion into a BST is in section 12.3, but the textbook uses an iterative version of insertion.