While you wait

The following nodes are added to a binary search tree (BST) in order. Draw the resulting BST.

6, 8, 2, 4, 1, 7, 5, 3, 9

```java
public void add(int newValue){
    if(root == null)
        root = new TreeNode(newValue);
    else
        add(newValue, root);
}
public void add(int newValue, TreeNode current) {
    if (newValue < current.myValue) {
        if (current.myLeft == null)
            current.myLeft = new TreeNode(newValue);
        else
            add(newValue, current.myLeft);
    }
    else if (current.myRight == null)
        current.myRight = new TreeNode(newValue);
    else
        add(newValue, current.myRight);
}
```

MARKETING + FINANCE + PUBLIC RELATIONS
TECHNOLOGY + CONSULTING + BUSINESS +
ENTREPRENEURSHIP + VENTURE CAPITAL +
SALES + HUMAN RESOURCES + PROGRAMS

Start your future. Join ours.
BST

• My answer
  • 6, 8, 2, 4, 1, 7, 5, 3, 9

Today

• Trees
  • The importance of balanced trees
  • Tree traversals
  • Heaps – a tree
    • Priority Queues
• The following nodes are added to a binary search tree (BST) in order. Draw the resulting BST.

• 1, 2, 3, 4, 5, 6, 7, 8, 9
• Snarf today’s code.
• With a partner, run the code
  • Make a graph of
    • x-axis – number of nodes
    • y-axis – tree height
• Run the code replacing
  • `tree.add(i);` with `tree.add(r.nextInt());`
  • Make another graph
**BST**

- Order matters!
  - random input $\Rightarrow$ height $O(\log N)$
  - ordered input $\Rightarrow$ height $O(N)$

---

**Today**

- Trees
  - The importance of balanced trees
  - Traversals
  - Heaps
    - Priority Queues
**Tree traversals**

- Given a BST, how would you print the nodes In Order?

```
6
2 8
1 4 7 9
3 5
```

**Tree traversals**

- Given a BST, how would you print the nodes In Order?

  - recursiveCall(go left)
  - current
  - recursiveCall(go right)
Tree traversals

• Given a BST, how would you print the nodes in Pre Order?
  
  • recursiveCall(go left)
  • current
  • recursiveCall(go right)

Tree traversals

• Given a BST, how would you print the nodes in Pre Order?
  
  • current
  • recursiveCall(go left)
  • recursiveCall(go right)

• Duplicate a tree
Tree traversals

- Given a BST, how would you print the nodes in Post Order?
  - recursiveCall(go left)
  - recursiveCall(go right)
  - current

- Delete a tree

Trees

- Applications
  - Computer graphics
  - Database
  - File storage on your computer
  - Internet protocols
  - Data visualizations
Today

- Trees
  - The importance of balanced trees
  - Traversals
  - Heaps
    - Priority Queues

Queues and Stacks

```java
public static void main(String[] args) {
    Queue aQueue = new LinkedList();
    Stack aStack = new Stack();
    String[] wordsToAdd = {"compsci", "201", "is", "great"};

    for(String s: wordsToAdd){
        aQueue.add(s); //enqueue
        aStack.push(s);
    }

    while(!aQueue.isEmpty())
        System.out.print(aQueue.remove() + " "); //dequeue

    System.out.println();

    while(!aStack.isEmpty())
        System.out.print(aStack.pop() + " ");
}
```
Priority Queue

• Airport queue
  • First class?

```
public static void main(String[] args) {
    PriorityQueue<String> aQueue = new PriorityQueue<String>();
    String[] wordsToAdd = {"compsci", "201", "is", "great"];
    for(String s: wordsToAdd) {
        aQueue.add(s);
        aStack.push(s);
    }
    while(!aQueue.isEmpty())
        System.out.print(aQueue.remove() + " ");
}
```

1. compsci 201 is great
2. great is compsci 201
3. is great compsci 201
4. 201 compsci great is
Priority Queue

• What is the output?

```java
PriorityQueue<Integer> ex = new PriorityQueue<Integer>();
ex.add(2);
ex.add(13);
ex.add(9);
ex.add(75);
ex.add(4);
while(!ex.isEmpty()) {
    System.out.println(ex.remove());
}
```

• Add in any order
• Remove by natural ordering (compareTo)

Heaps

• Common implementation of priority queues
• A tree-like structure
• Almost completely filled
  • All nodes filled expect last level (left to right)
• Max-Heap - Descendants have values <= to parent
• Min-Heap - Descendants have values >= to parent
Heaps

• Why is a heap implemented with a priority queue?
  • Where is the min value?

• Add 55 to heap
  • add node to first open slot
Heaps

- Add 55 to heap
  - If parent is larger, swap
**Heaps**

- Add 55 to heap
  - If parent is larger, swap

```
Heaps as Arrays
```

```
Layer 1  Layer 2  Layer 3  Layer 4
20  75  43  84  90  57  71  96  91  93
```

```
Layer 1  Layer 2  Layer 3  Layer 4
20  75  43  84  90  57  71  96  91  93
```
public void add(double d) {
    mySize++;
    myMinHeap[mySize] = d;
    int index = mySize;
    int parentIndex = index/2;
    while((myMinHeap[parentIndex] > myMinHeap[index]) &
        parentIndex != 0)
    {
        swap(index, parentIndex);
        index = parentIndex;
        parentIndex = index/2;
    }
}

private void swap(int i, int j){
    double temp = myMinHeap[i];
    myMinHeap[i] = myMinHeap[j];
    myMinHeap[j] = temp;
}
public void add(double d){
    mySize++;
    myMinHeap[mySize] = d;
    int index = mySize;
    int parentIndex = index/2;
    while((myMinHeap[parentIndex] > myMinHeap[index]) &
        parentIndex != 0){
        swap(index, parentIndex);
        index = parentIndex;
        parentIndex = index/2;
    }
}

private void swap(int i, int j){
    double temp = myMinHeap[i];
    myMinHeap[i] = myMinHeap[j];
    myMinHeap[j] = temp;
}
Heaps as Arrays

```java
public void add(double d) {
    mySize++;
    myMinHeap[mySize] = d;
    int index = mySize;
    int parentIndex = index / 2;
    while ((myMinHeap[parentIndex] > myMinHeap[index]) &&
        parentIndex != 0) {
        swap(index, parentIndex);
        index = parentIndex;
        parentIndex = index / 2;
    }
}

private void swap(int i, int j) {
    double temp = myMinHeap[i];
    myMinHeap[i] = myMinHeap[j];
    myMinHeap[j] = temp;
}
```

Remove

- Remove the root
- Move last value into root
- If a child is smaller than root
- promote the smallest child

- What would the array look like if I called remove()?
Practice

• Draw the array for the following heap

• Add the value 27
  • redraw the array

Code

• People are waiting in line for coffee and the barista wants to help as many people as possible in the shortest amount of time.

• Complete the snarfed code CofeeCulture, (which uses a PriorityQueue) to calculate the maximum number of people to get coffee in 5 minutes?
Today

IT'S A CHRISTMAS TREE WITH A HEAP OF PRESENTS UNDERNEATH!

... WE'RE NOT INVITING YOU HOME NEXT YEAR.