Computer Science 112
Data Structures

Lecture 10:
Binary Search
Binary Search Trees

Midterm Exam

- Sunday, March 1
- 3 – 4:20
- Rooms to be announced
- Know which recitation you are registered for

Review: Big O for Block Search

- Sorted array, size n
- think of it as m blocks of s elements each
- compare target with last element of first block: a[s-1]
  - if target == a[s-1] target has been found
  - else if target < a[s-1] search from start of block to a[2-1]
  - otherwise, redo on next block
- What is worst case cost? What is average cost?

New: Search an ordered array

- Question game
  - You think of a number between 1 and 1,000
  - I will guess a number
  - You tell me if your number is
    - the same as my guess,
    - bigger than my guess,
    - smaller than my guess

Search an ordered array

- In an ordered array, one test can rule out a whole region of the array

```
if (a[10] < target){
    // if we get here target can’t be in a[j] for
    // 0 <= j <= 10
```

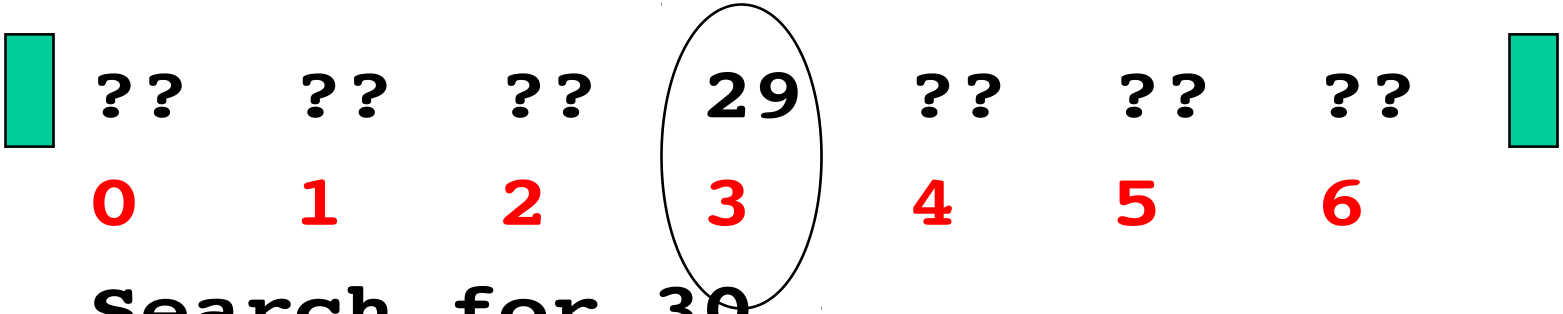
Binary Search

🟩	??	??	??	??	??	??	??	🟩
	0	1	2	3	4	5	6	

Search for 30

left	right	mid
0	6	

Binary Search

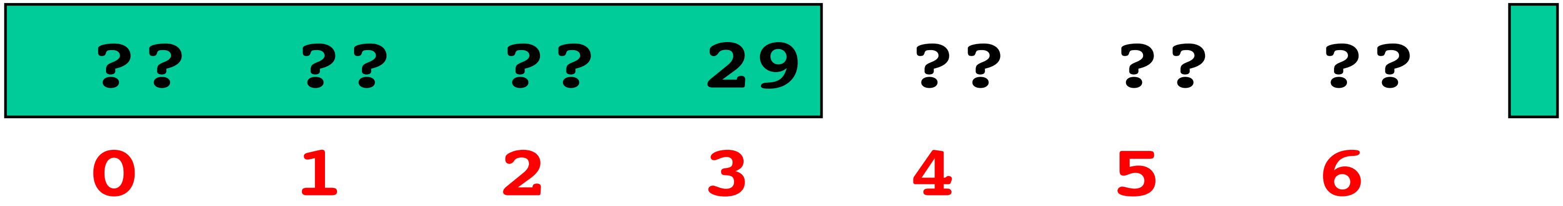


??	??	??	29	??	??	??
0	1	2	3	4	5	6

Search for 30

left	right	mid
0	6	3

Binary Search

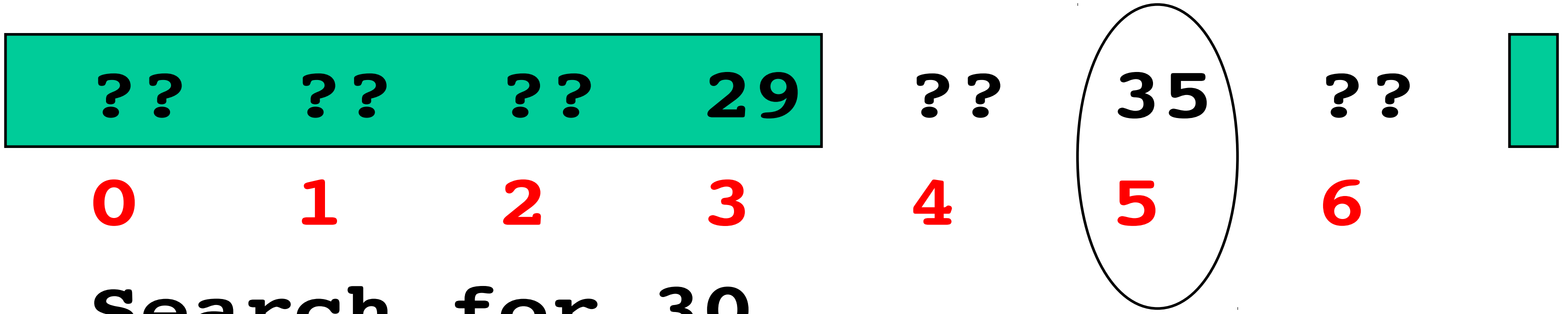


??	??	??	29	??	??	??
0	1	2	3	4	5	6

Search for 30

left	right	mid
4	6	

Binary Search



??	??	??	29	??	35	??	
0	1	2	3	4	5	6	

Search for 30

left	right	mid
4	6	5

Binary Search

??	??	??	29	??	35	??
0	1	2	3	4	5	6

Search for 30

left	right	mid
4	4	

Binary Search

22	25	26	29	30	35	??
0	1	2	3	4	5	6

Search for 30

left	right	mid
4	4	4

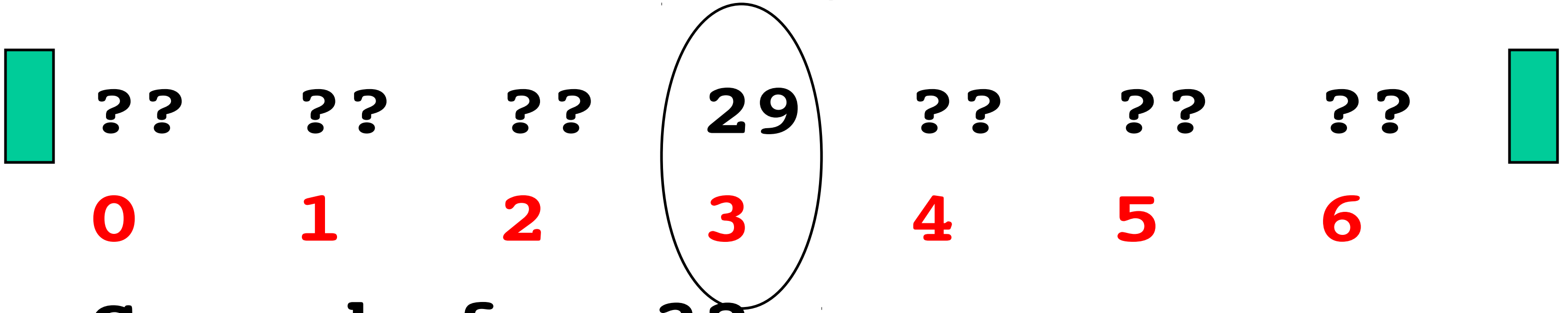
Binary Search

🟩	??	??	??	??	??	??	??	🟩
	0	1	2	3	4	5	6	

Search for 32

left	right	mid
0	6	

Binary Search

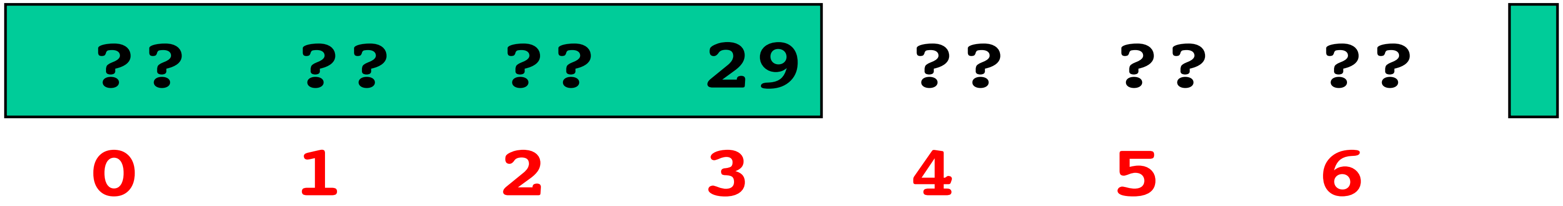


??	??	??	29	??	??	??
0	1	2	3	4	5	6

Search for 32

left	right	mid
0	6	3

Binary Search

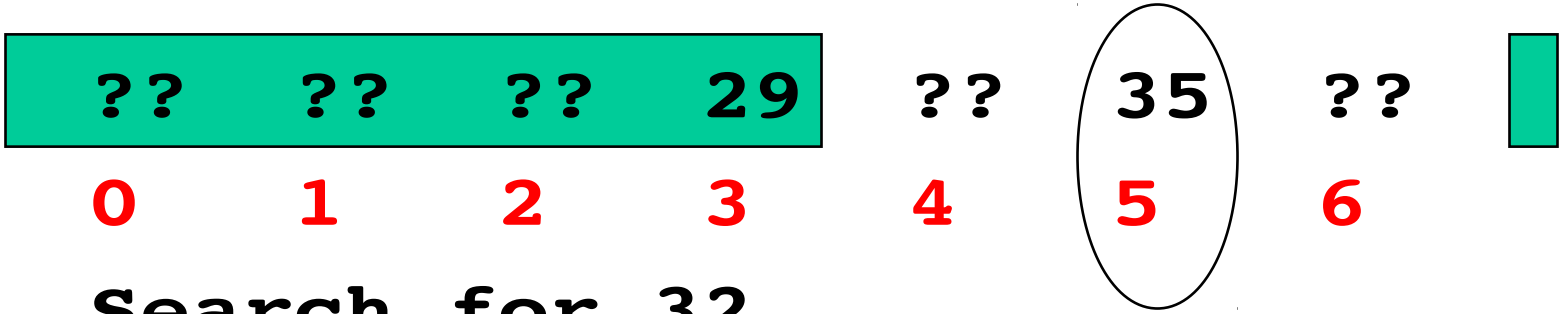


??	??	??	29	??	??	??
0	1	2	3	4	5	6

Search for 32

left	right	mid
4	6	

Binary Search



??	??	??	29	??	35	??
0	1	2	3	4	5	6

Search for 32

left	right	mid
4	6	5

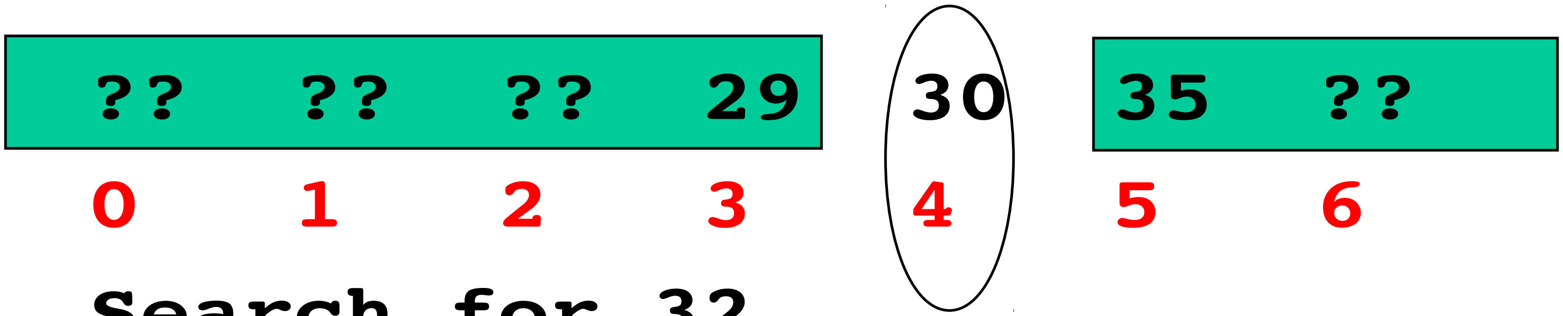
Binary Search

??	??	??	29	??	35	??
0	1	2	3	4	5	6

Search for 32

left	right	mid
4	4	

Binary Search



??	??	??	29	30	35	??
0	1	2	3	4	5	6

```
Search for 32
  left    right    mid
   4        4       4
```

Binary Search

??	??	??	29	30	35	??
0	1	2	3	4	5	6

Search for 32

left	right	mid
5	4	

See BinarySearch.java

See RecursiveBinSearch.java

How Many Questions

- **How many questions for 1000 numbers?**
  - **Each question eliminates 1/2 of possibilities**
  - **Therefore each question doubles the size of array you can search**
  - **Therefore size = $2^{\text{questions}}$**
  - **Therefore $\log_2$(size) = questions**
  - **$\text{Log}_2\,(1000) \approx 10$**
  - **Answer: 10 questions**

Searching an array Performance

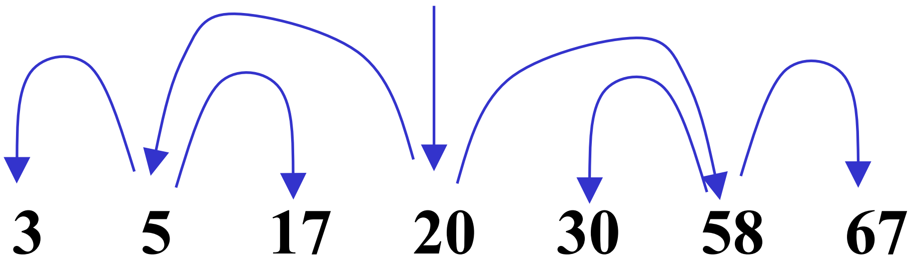
- Search among 1 Million entries
- Check 1 million entries per second
  - Sequential search
    - 1 million operations needed
    - Requires 1 second
  - Binary earch
    - 20 operations needed
    - Requires 20 microseconds
    - 50,000 times faster

Searching an array
Performance

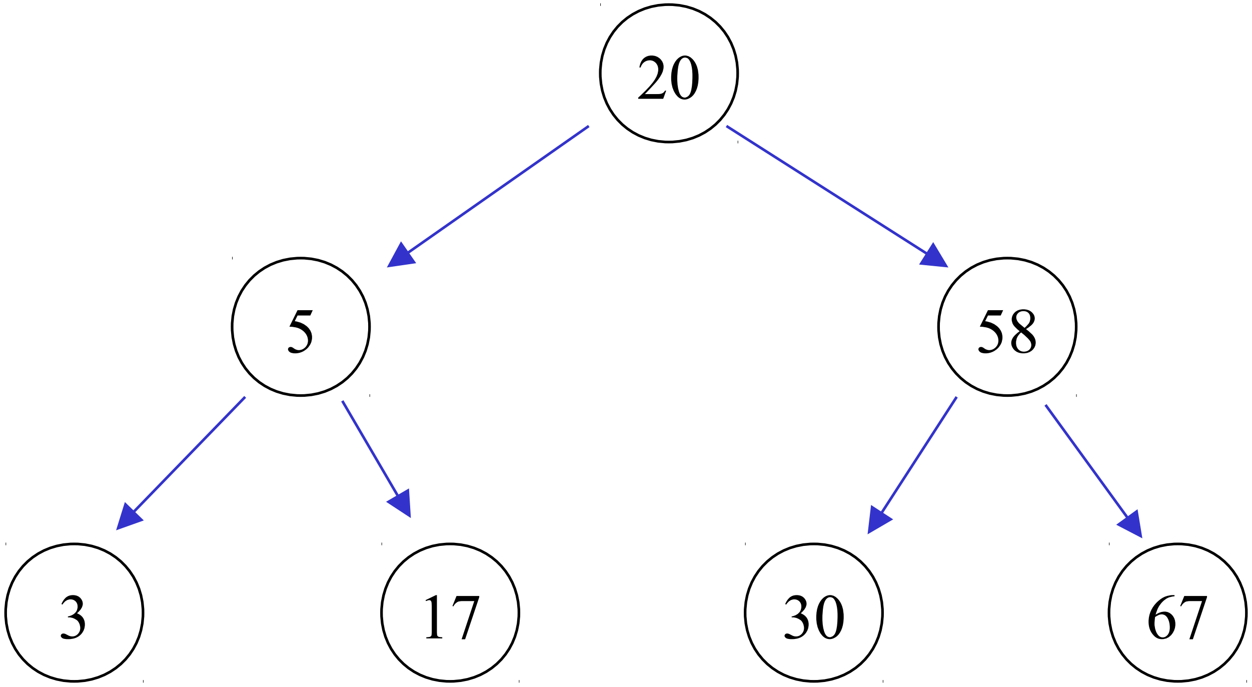
- Search among 1 Billion entries
- Check 1 million entries per second
  - Sequential search
    - 1 billion operations needed
    - Requires 1000 seconds - about 20 minutes
  - Binary search
    - 30 operations needed
    - Requires 30 microseconds
    - 30 million times faster

Binary Search Trees

- Why can't we do binary search on a linked list?
  - can't jump to middle
- Lets keep a pointer to the middle!
- Then what?



This looks like a binary tree!



Binary Tree Nodes

- **Each node has**
  - **a data field**
  - **a left subtree, which is a Binary Tree Node**
  - **a right subtree, which is a Binary Tree Node**

Binary Search Tree

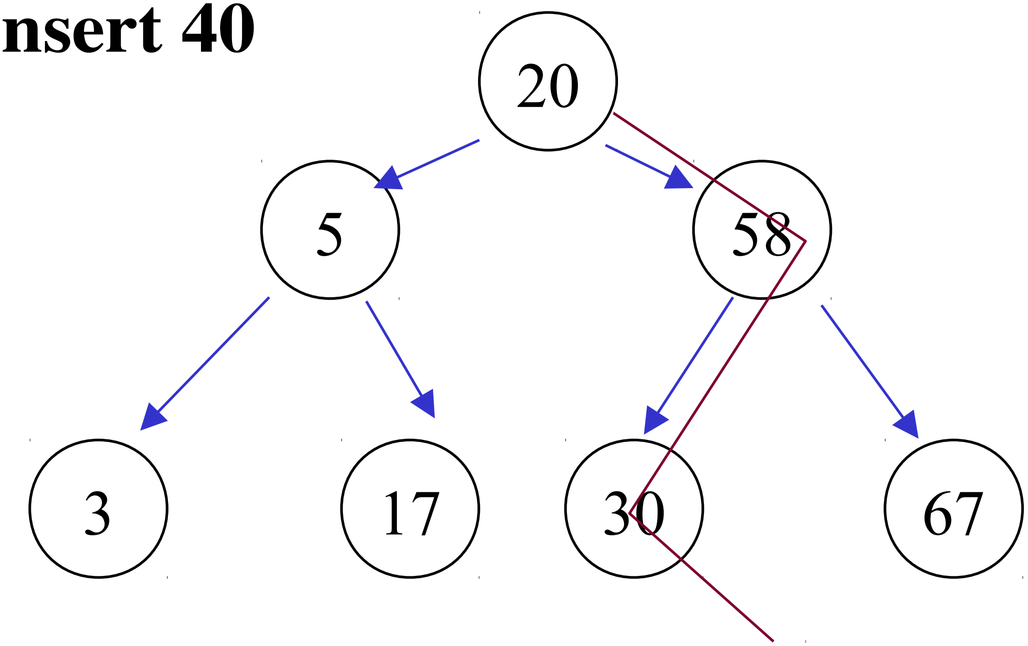
- data at a node is > any data in left subtree
- data at a node is < any data in right subtree
- Therefore, to print a BST in data order:
  - Print left subtree in data order
  - Print data
  - Print right subtree in data order

Search

- Searching a BST is easy
  - if node = null, search fails
  - if node.data equals target, found
  - if target < node.data, search on left subtree
  - else search on right subtree
- See BST.java, BSTApp.java

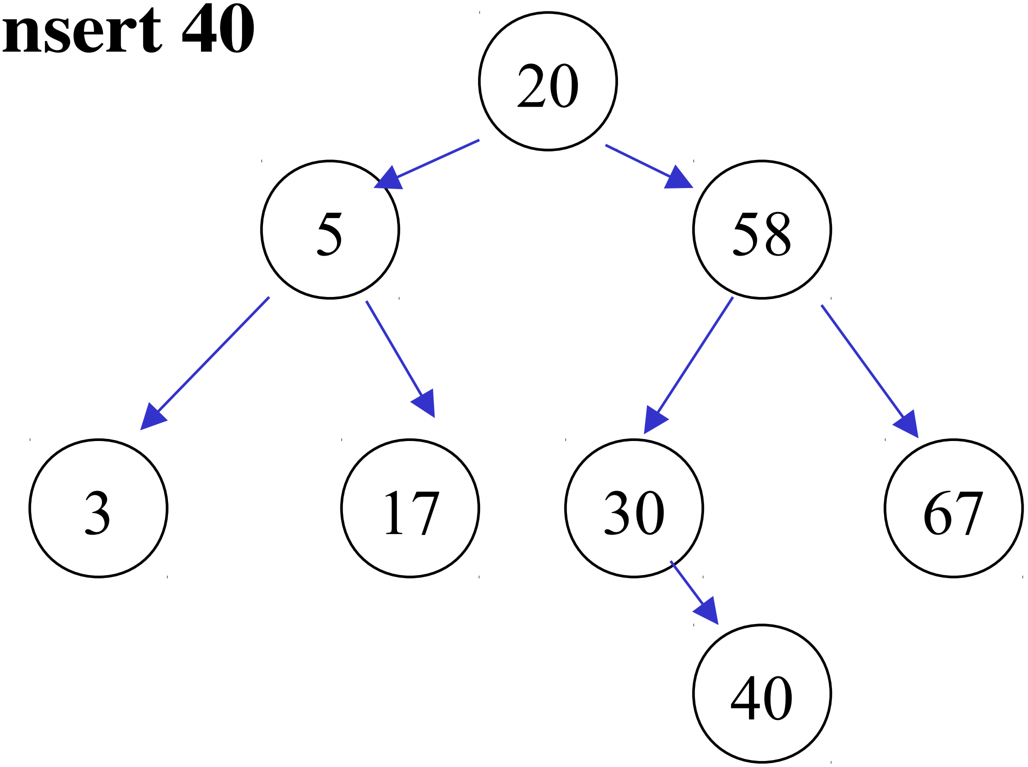
Insert

- **Search, fail, insert where failed**
  - **Insert 40**



Insert

- Search, fail, insert where failed
  - Insert 40


20
5
58
3
17
30
67
40