Announcements

Join Code: **225**

Extra Credit Survey!

1. MP Lists due Monday
2. Exam 2 Next Week (M-W)
3. Exam 1 Grades Released
    a. Exam Regrade Form Online
4. Extra Credit Survey Form (2pts, 70% of class must respond)

Warm-Up Question: Using a BST, how could I figure out what points fall between a range? Ex. Find all the values in the tree between (20, 40).

KD Trees

Learning Objectives

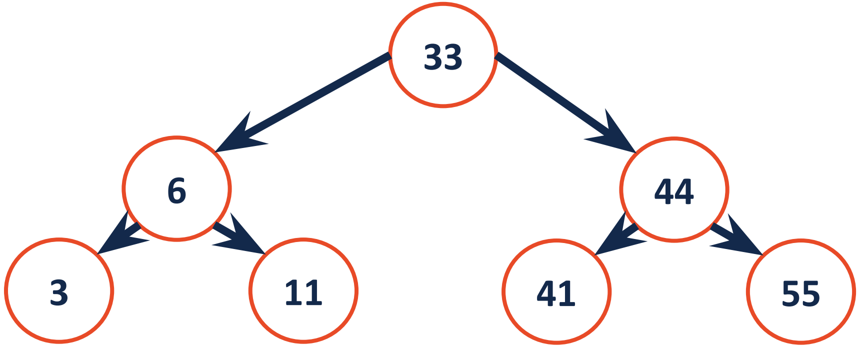
1. Articulate what questions a KD-Tree helps answer
2. Understand the KD-Tree Construction Algorithm
3. Implement the Quick Select Algorithm

Nearest Neighbor and Range Search

Join Code: **225**

What are situations where you would want the nearest point in your database to your search query?

What are situations where you would like a range?

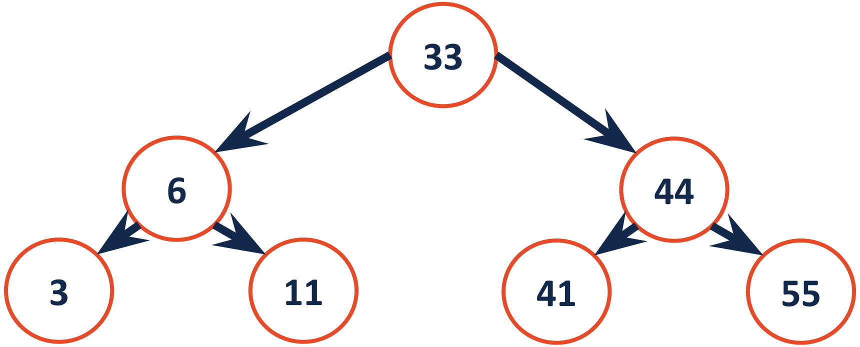


Closest Element

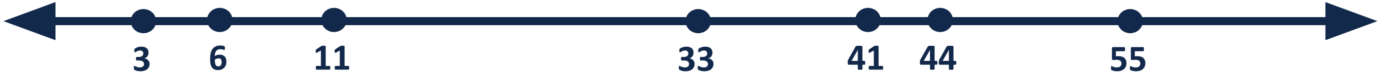
BSTs are useful structures for range-based and nearest-neighbor searches.

Q: Consider points in 1D: **$p = \{p_1, p_2, \ldots, p_n\}$.**

…what point is closest to 16?

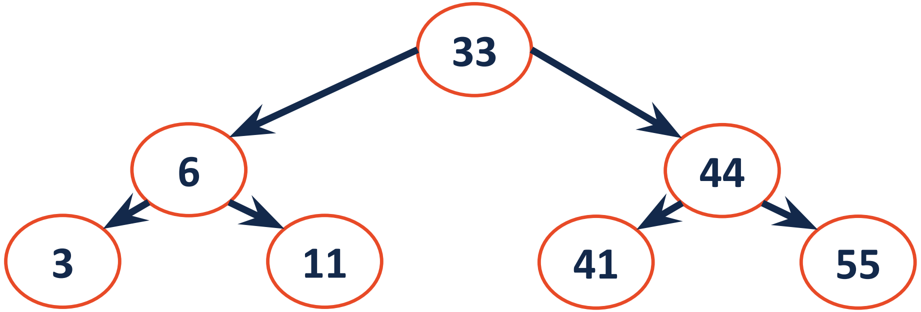


Ex:



Range-based Searches

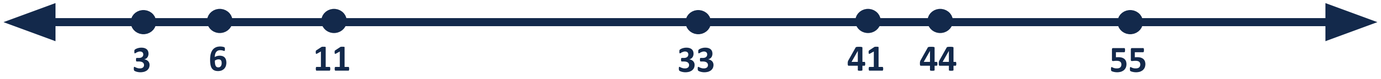
BSTs are useful structures for range-based and nearest-neighbor searches.



Q: Consider points in 1D: **$p = \{p_1, p_2, \ldots, p_n\}$.**

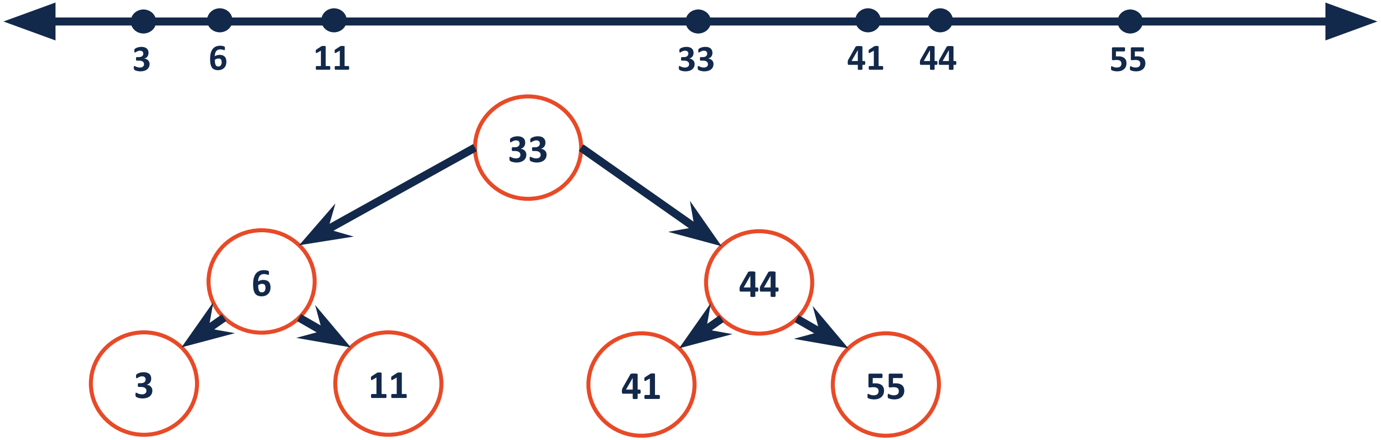
…what points fall in [11, 42]?

Ex:



Range-based Searches

Query: [11,42]

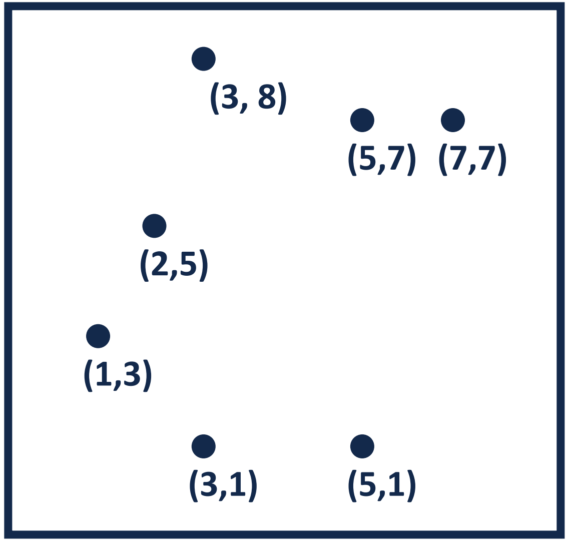


2D Range Based Searches

Consider points in 2D: $\mathbf{p} = \{p_1, p_2, \ldots, p_n\}$

Q: What points are in the rectangle:
[(4, 4), (6, 9)]?

Q: What is the nearest point to **(4,3)**?

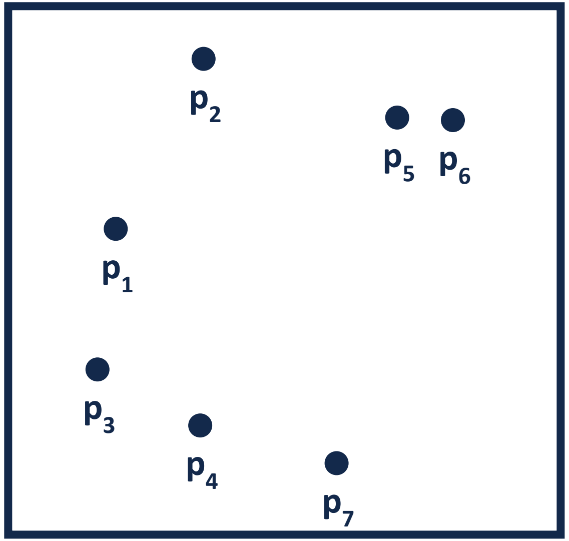


2D Range Based Searches

Consider points in 2D: **$p = \{p_1, p_2, \ldots, p_n\}$**

Tree Construction:

1. Find median point along a dimension and partition nodes
2. Go to next dimension
3. Recursively build left subtree
4. Recursively build right subtree



Quick Select Algorithm

Partitions elements about the median

Smaller Values are on the left of the median

Larger Values are on the right of the median

Faster than sorting O(n log n)

Ex. Input: [11, 6, 44, 41, 33, 57, 2]

Output: [11, 6, 2, 33, 41, 57, 44]

Quick Select Algorithm

1. Select a random pivot
2. Swap the pivot value to the end
3. Partition elements
4. Swap pivot value to proper place

Quick Select Algorithm

Ex. [11, 6, 44, 41, 33, 57, 2]

For a random pivot index, k = 3

[11, 6, 44, 2, 33, 57, 41], i = 0, small = 0

Quick Select Algorithm

Ex. [11, 6, 44, 41, 33, 57, 2]

For a random pivot index, k = 3

[11, 6, 44, 2, 33, 57, 41], i = 0, small = 0

```
// swap pivot and last value
swap(arr[k], arr[length-1]);

for (int i = 0; i < length; i++){
        if (arr[i] < arr[length - 1]){
                swap(arr[i], arr[small]);
                small++;
        }
}
```

Quick Select Algorithm

Join Code: **225**

Ex. [11, 6, 44, 41, 33, 57, 2], For a random pivot index, k = 3

arr	i	small
[11, 6, 44, 2, 33, 57, 41]	0	0
	1	
	2	
	3	
	4	
	5	
	6	

Quick Select Algorithm

Ex. [11, 6, 44, 41, 33, 57, 2], For a random pivot index, k = 3

arr	i	small
[11, 6, 44, 2, 33, 57, 41]	0	0
[11, 6, 44, 2, 33, 57, 41]	1	1
[11, 6, 44, 2, 33, 57, 41]	2	2
[11, 6, 44, 2, 33, 57, 41]	3	2
[11, 6, 2, 44, 33, 57, 41]	4	3
[11, 6, 2, 33, 44, 57, 41]	5	4
[11, 6, 2, 33, 44, 57, 41]	6	4

Quick Select Algorithm

[11, 6, 2, 33, 44, 57, 41], i = 6, small = 4

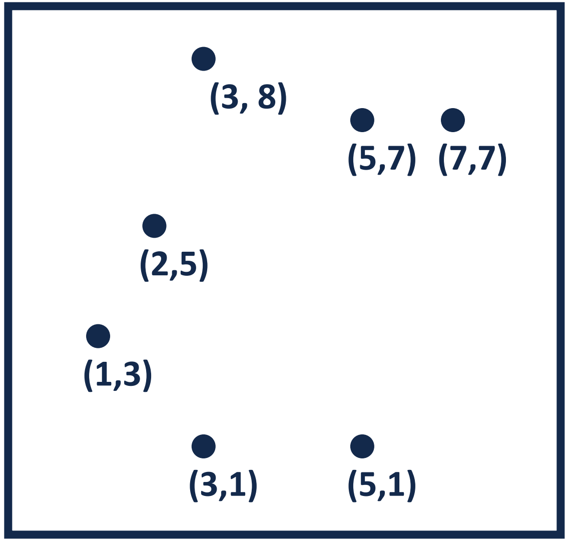
[11, 6, 2, 33, 41, 57, 44], Swap pivot value back to proper place

Constructing a KD Tree

Consider points in 2D: **$p = \{p_1, p_2, \dots, p_n\}$**

Tree Construction:

1. Find median point along a dimension and partition nodes
2. Go to next dimension
3. Recursively build left subtree
4. Recursively build right subtree



Worst Case Analysis

Average Case Analysis