# CS 225

**Data Structures**

*Oct. 20 – AVL Applications*

# Every Data Structure So Far

| | Unsorted Array | Sorted Array | Unsorted List | Sorted List | Binary Tree | BST | AVL |
|---|---|---|---|---|---|---|---|
| **Find** | | | | | | | |
| **Insert** | | | | | | | |
| **Remove** | | | | | | | |
| **Traverse** | | | | | | | |

# Share Your #cs225animation

**On Facebook/Twitter/Instagram:**

**#cs225animation**

*…I'll search this tag every few days and like/heart your work!*

**On Piazza:**

See pinned post: "MP4: Animation Sharing"

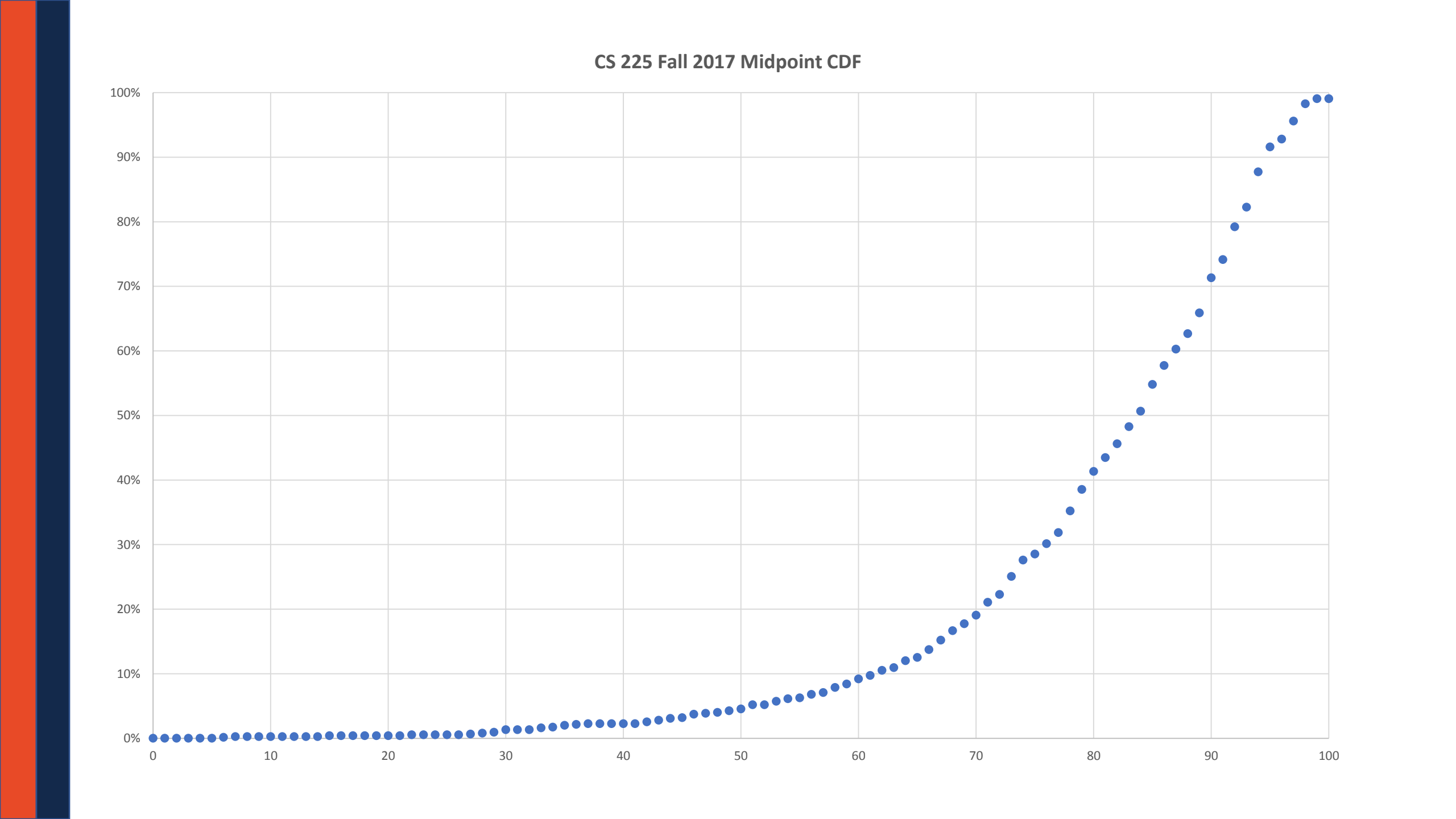
# Mid-Point Grade Updates

**Your mid-point grade update is on Compass 2g!**

- Column: “Midpoint Total (without EC) /314”
- Column: “Midpoint % (without EC)”

**Statistics:**

- 314 total points
- Median Grade: 83%
- Average Grade: 80%


CS 225 Fall 2017 Midpoint CDF
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
0
10
20
30
40
50
60
70
80
90
100

# CS 225 Open Lab Hours

# Range-based Searches

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

…what points fall in [11, 42]?

**Tree construction:**

# Range-based Searches

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

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

…what points fall in [11, 42]?

**Ex:**

# Range-based Searches

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

…what points fall in [11, 42]?

**Ex:**

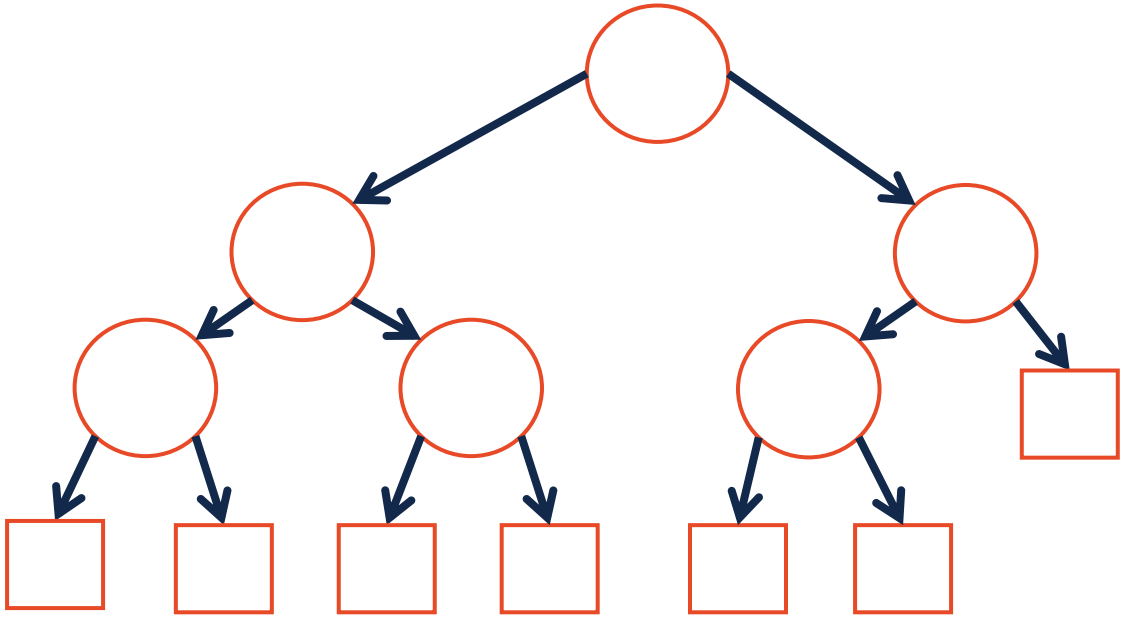
# Range-based Searches

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

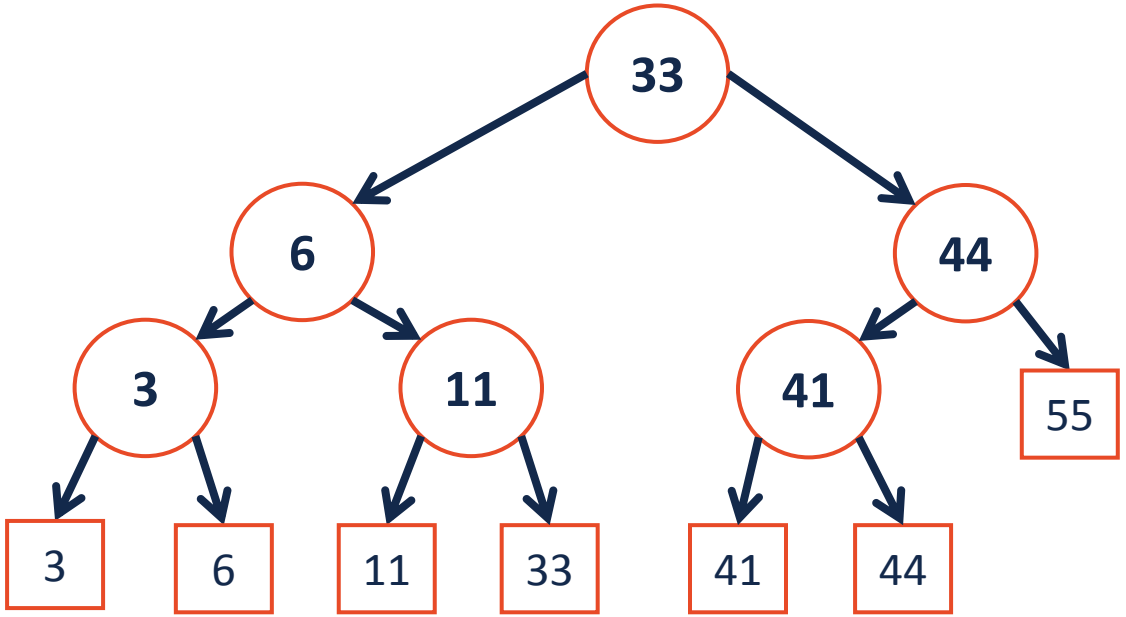
…what points fall in [11, 42]?

**Tree construction:**

# Range-based Searches



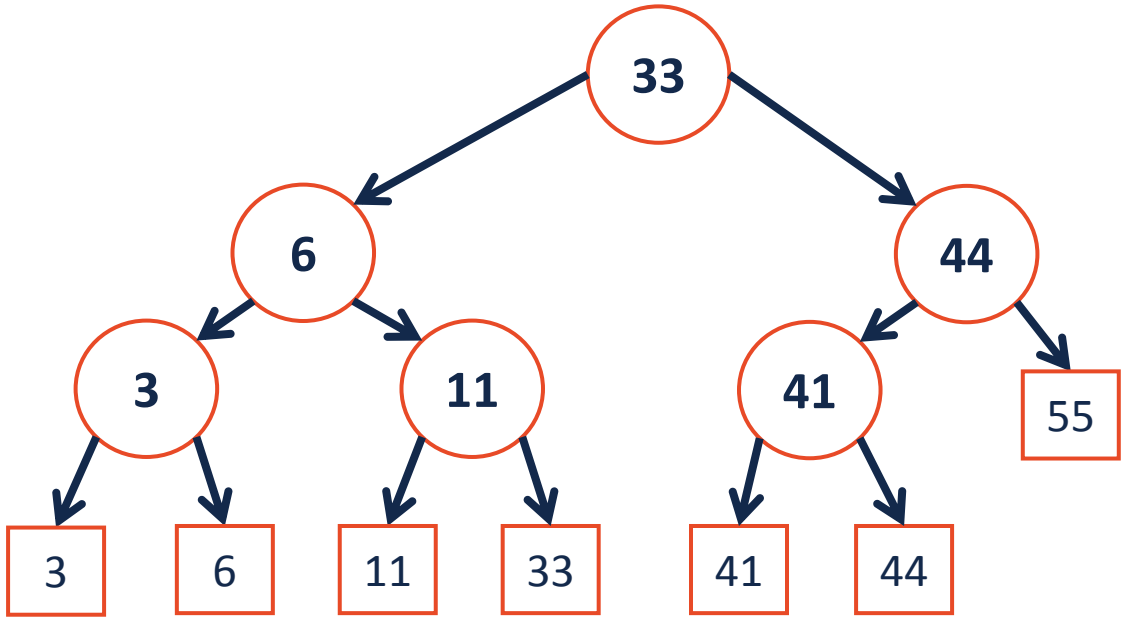
# Range-based Searches



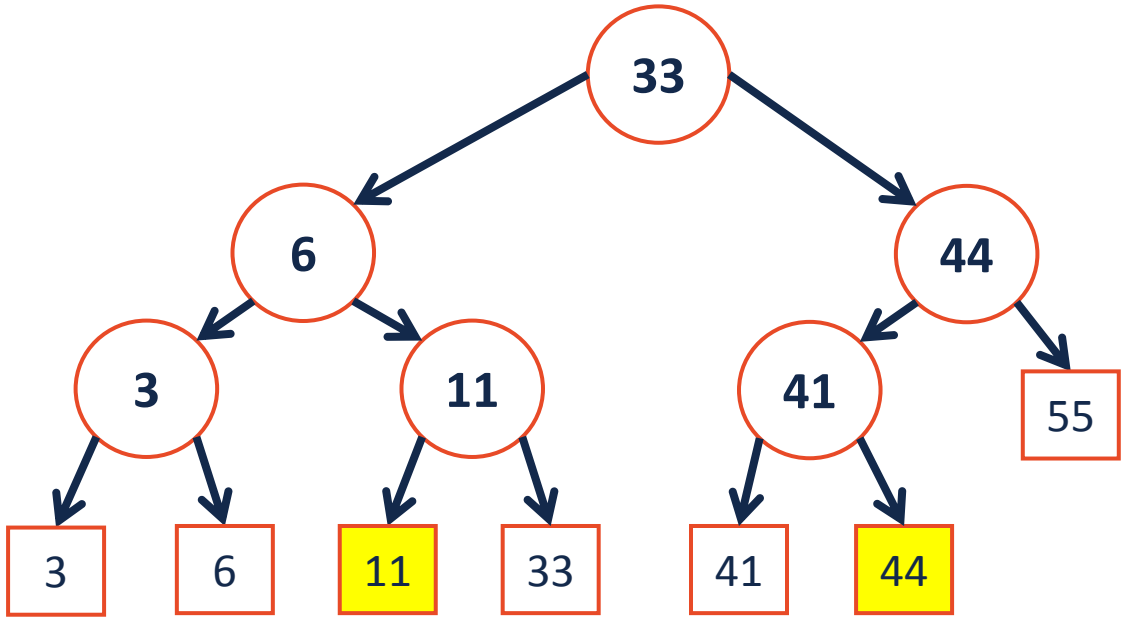
# Range-based Searches

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

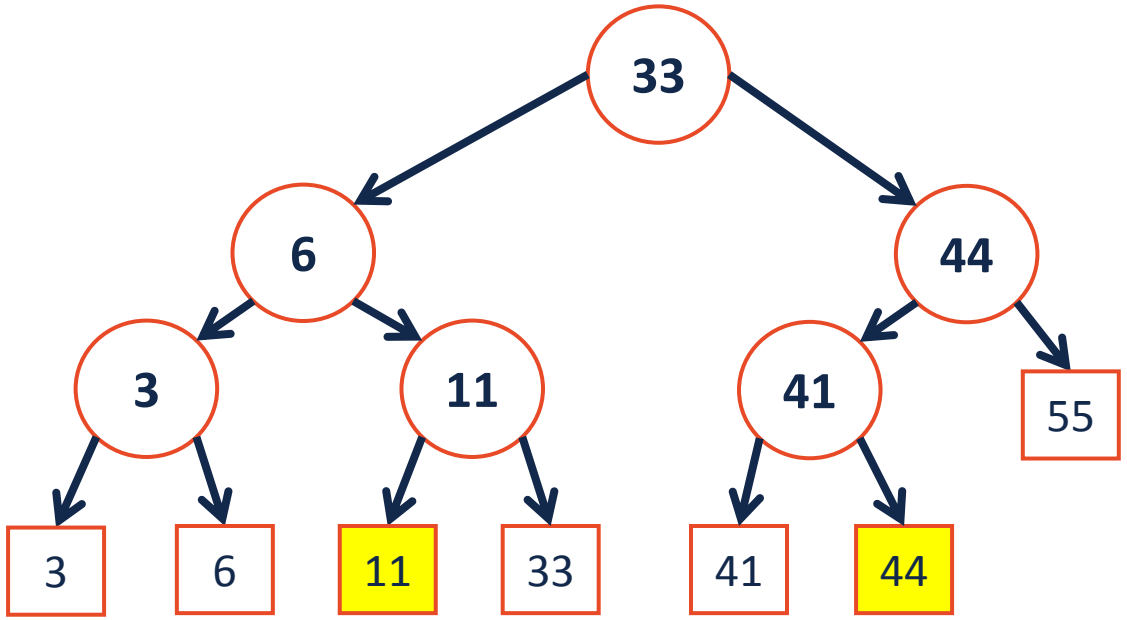
…what points fall in [11, 42]?



# Range-based Searches



# Running Time



# Range-based Searches

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

…what points fall in [11, 42]?

**Ex:**

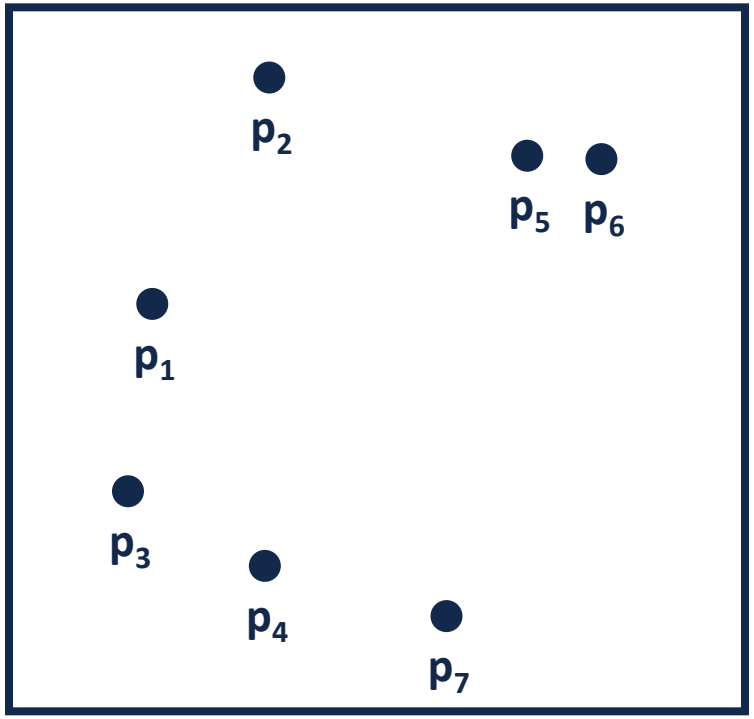
# Range-based Searches

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

**Q:** What points are in the rectangle:

$\mathbf{[\ (x_1, y_1),\ (x_2, y_2)\ ]}$?

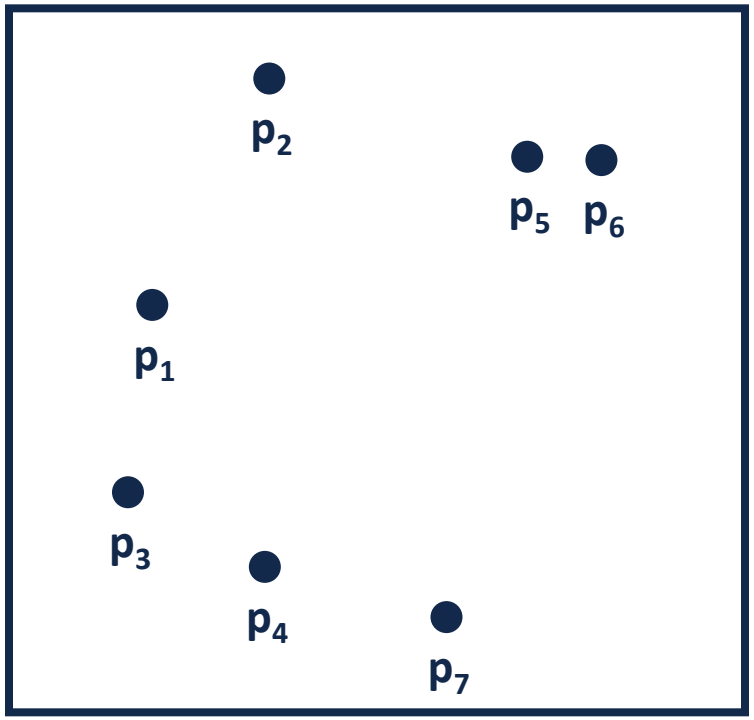
**Q:** What is the nearest point to $\mathbf{(x_1, y_1)}$?



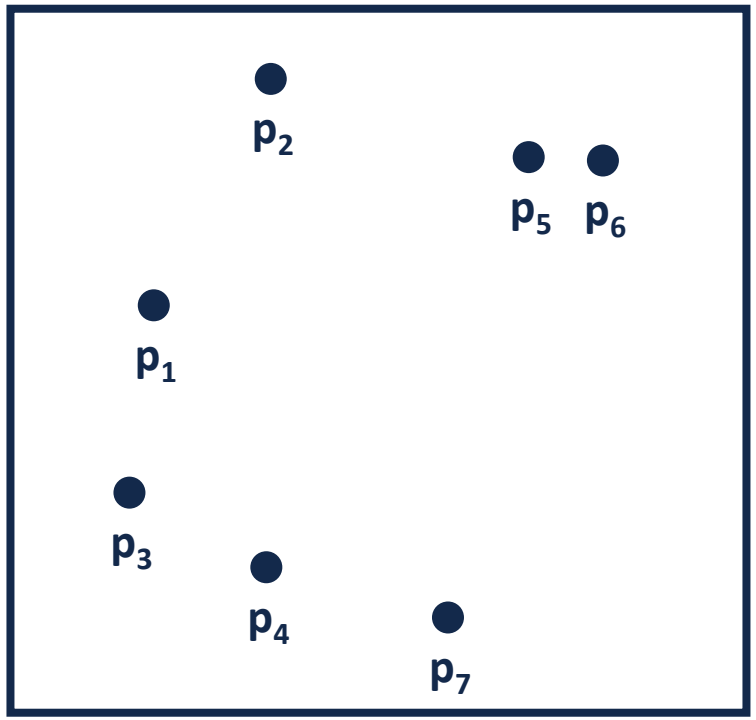
# Range-based Searches

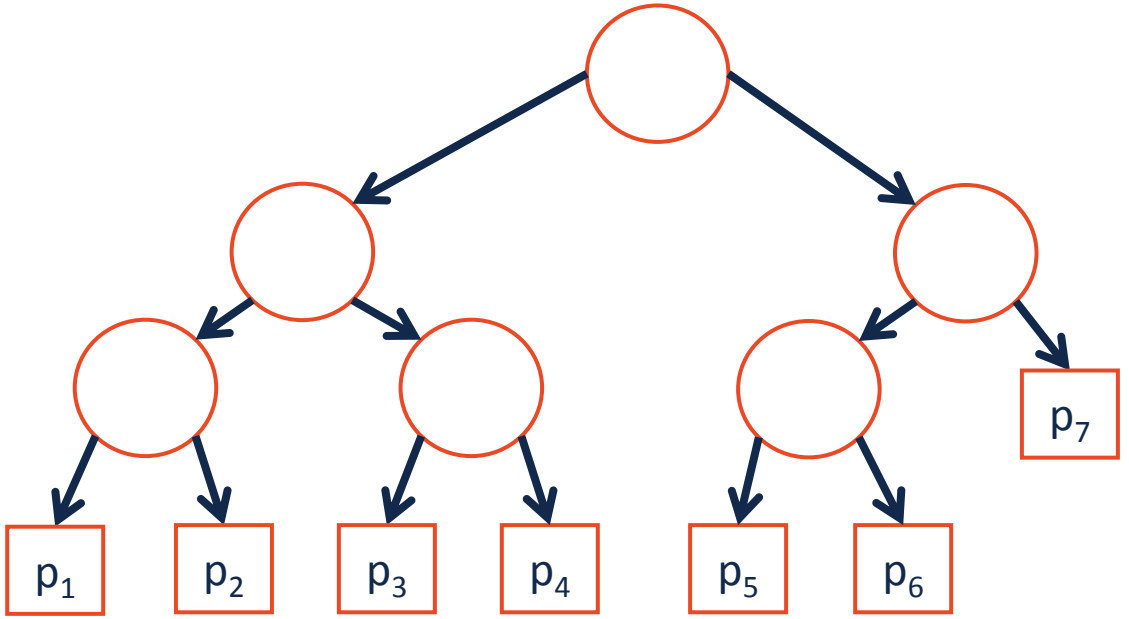
Consider points in 2D: $\mathbf{p = \{p_1, p_2, \ldots, p_n\}}$.

**Tree construction:**

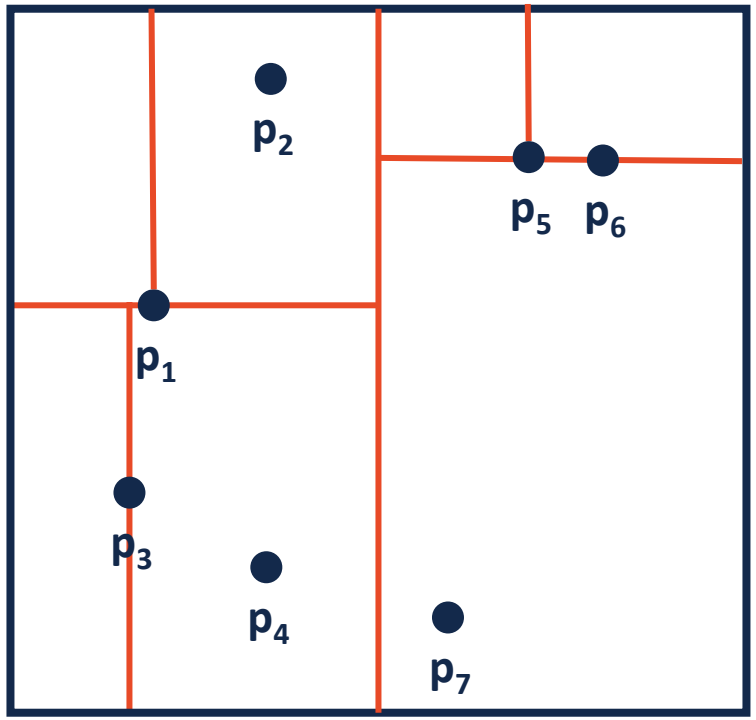


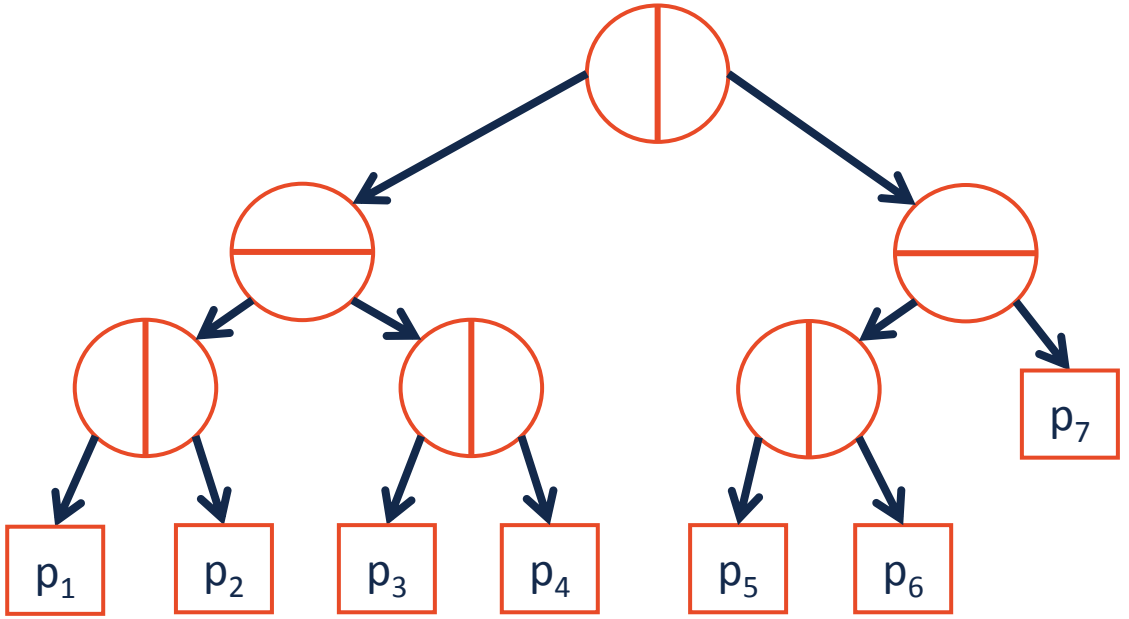
# Range-based Searches



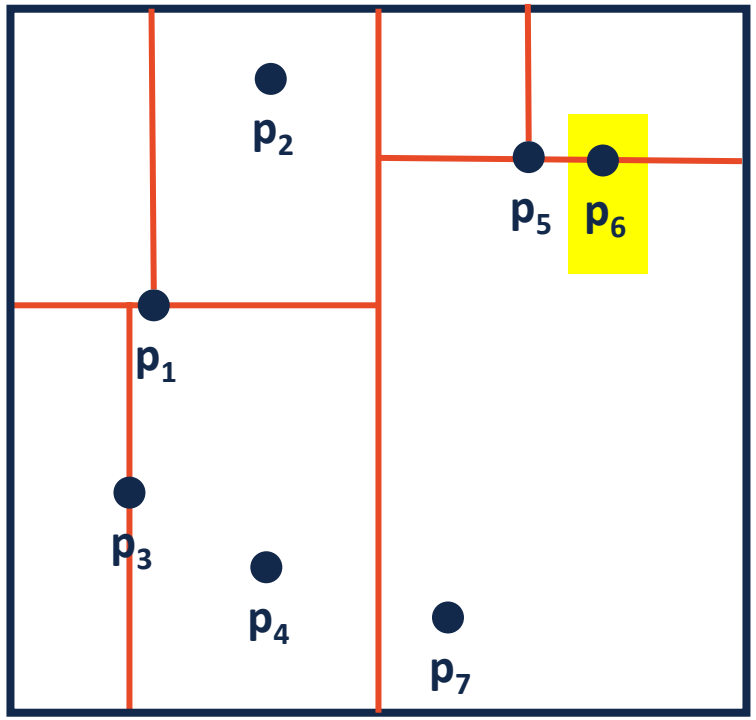



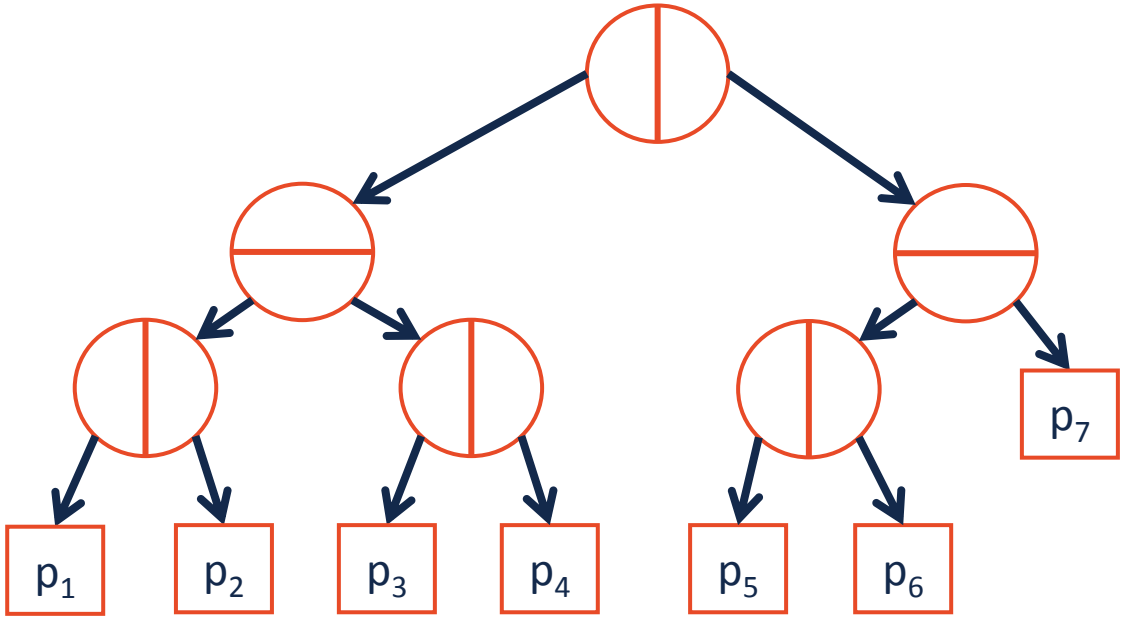
# kD-Trees



# kD-Trees



# MP5

# Iterators

## Why do we care?

# Iterators

## Why do we care?

```
1  DFS dfs(...);
2  for ( ImageTraversal::Iterator it = dfs.begin(); it != dfs.end(); ++it ) {
3    std::cout << (*it) << std::endl;
4  }
```

# Iterators

## Why do we care?

```
1  DFS dfs(...);
2  for ( ImageTraversal::Iterator it = dfs.begin(); it != dfs.end(); ++it ) {
3    std::cout << (*it) << std::endl;
4  }
```

```
1  DFS dfs(...);
2  for ( const Point & p : dfs ) {
3    std::cout << p << std::endl;
4  }
```

# Iterators

## Why do we care?

```
1  DFS dfs(...);
2  for ( ImageTraversal::Iterator it = dfs.begin(); it != dfs.end(); ++it ) {
3    std::cout << (*it) << std::endl;
4  }
```

```
1  DFS dfs(...);
2  for ( const Point & p : dfs ) {
3    std::cout << p << std::endl;
4  }
```

```
1  ImageTraversal & traversal = /* ... */;
2  for ( const Point & p : traversal ) {
3    std::cout << p << std::endl;
4  }
```

# Iterators

```
1  ImageTraversal *traversal = /* ... */;
2  for ( const Point & p : traversal ) {
3    std::cout << p << std::endl;
4  }
```

# Iterators

```
std::list<Sphere> sphereList;
...
for (const Sphere & s : sphereList) {
  ...
}
```

```
std::vector<Sphere> sphereList;
...
for (const Sphere & s : sphereList) {
  ...
}
```

```
std::map<std::string, Sphere> sphereMap;
...
for (const std::pair<std::string, Sphere> & kv : sphereMap) {
  ...
}
```

# CS 225 – Things To Be Doing

## Exam 7 (theory) starts Monday!

**Review Document: On Piazza**
**Review Session: 7pm, 1404 SC**
**More Info:** https://courses.engr.illinois.edu/cs225/fa2017/exams/

## MP4: Due Monday

*Due: Monday, Oct. 23 at 11:59pm*

## Lab: lab_avl

*Due Sunday, Oct. 22 at 11:59pm*

## POTD

Every Monday-Friday – *Worth +1 Extra Credit /problem (up to +40 total)*