

# CS210

# Discussion

Week 5

# Attendance



# Today

- Comparable vs Comparator
- Exercise hints
- Finish through exercise 3
  - Show us passing Gradescope tests to leave early

# Comparable vs. Comparator

## Comparable

- `compareTo()`
- An instance of the thing you're comparing
- Typically represents a 'natural' order

## Comparator

- `compare()`
- A separate object whose purpose is to compare two others
- Typically represents an alternative order
- Must be created before it can be used

# Comparable vs. Comparator

- For comparable, an object compares itself to another
- Comparison results are integers
  - $< 0$  if it's smaller
  - $0$  if they're equal
  - $> 0$  if it's bigger

```
public class Dog implements Comparable<Dog>{  
    3 usages  
    private int age;  
  
    public Dog(int age) {  
        this.age = age;  
    }  
  
    public int compareTo(Dog other) {  
        return this.age - other.age;  
    }  
}
```

```
Dog sparkles = new Dog( name: "Sparkles", humanAge: 3);  
Dog champ = new Dog( name: "Champ", humanAge: 5);  
StdOut.println(sparkles.compareTo(champ));
```

# Comparable vs. Comparator

```
private class NameComparator implements Comparator<Dog> {  
    public int compare(Dog firstDog, Dog secondDog) {  
        return firstDog.name.compareTo(secondDog.name);  
    }  
}
```

- Comparator compares two separate objects
- Comparison results are integers
  - < 0 if the first is smaller than the second
  - 0 if they're equal
  - > 0 if the first is bigger than the second

```
Comparator<Dog> nameComp = new NameComparator();  
StdOut.println(nameComp.compare(sparkles, champ));
```

# Questions?



# Six-Sided Dice

```
>_ ~/workspace/project3
```

```
$ java Die 5 3 4  
Dice a, b, and c:
```

```
*   *
```

```
  *
```

```
*   *
```

```
*
```

```
  *
```

```
   *
```

```
*   *
```

```
*   *
```

```
a.equals(b)      = false
```

```
b.equals(c)      = false
```

```
a.compareTo(b)   = 2
```

```
b.compareTo(c)   = -1
```

- A Die object to represent the roll of a die
- Can print the die face
- Is comparable based on face value



# Six-Sided Dice


- equals may look familiar
- Need to cast 'other' to Die

```
// Returns true if this die is the same as other, a
public boolean equals(Object other) {
    if (other == this) {
        return true;
    }
    if (other == null) {
        return false;
    }
    if (other.getClass() != this.getClass()) {
        return false;
    }
    // ...
}
```

```
Die otherDie = ((Die) other);
return otherDie.value == this.value;
```


# Six-Sided Dice

- We need to return an integer
  - $< 0$  if 'this' value is less than 'that' value
  - 0 if equal
  - $> 0$  if 'this' value is greater than 'that' value
- Remember that 'value' here is an integer
  - $5 - 2 = 3$
  - $2 - 5 = -3$

```
// Returns a comparison of this d
public int compareTo(Die that) {
    
}
```

# Six-Sided Dice

- Switch statement
  - 1 case for each number 1 – 6
- 0 Means Die hasn't been rolled yet
  - Should print "Not rolled yet"
  - Default case

```
// Returns a string representation of the die face  
public String toString() {  
      
}  

```

5 =

*						*	\n
			*				\n
*						*	\n



*						*	\n				*				\n	*						*	\n
---	--	--	--	--	--	---	----	--	--	--	---	--	--	--	----	---	--	--	--	--	--	---	----

# Location

- Similar to Die
- 'equal' needs to check all three: name, lat, and lon
- 'compareTo' is based on great circle distance to Greece
  - Use 'distanceTo' and a new Greece Location object

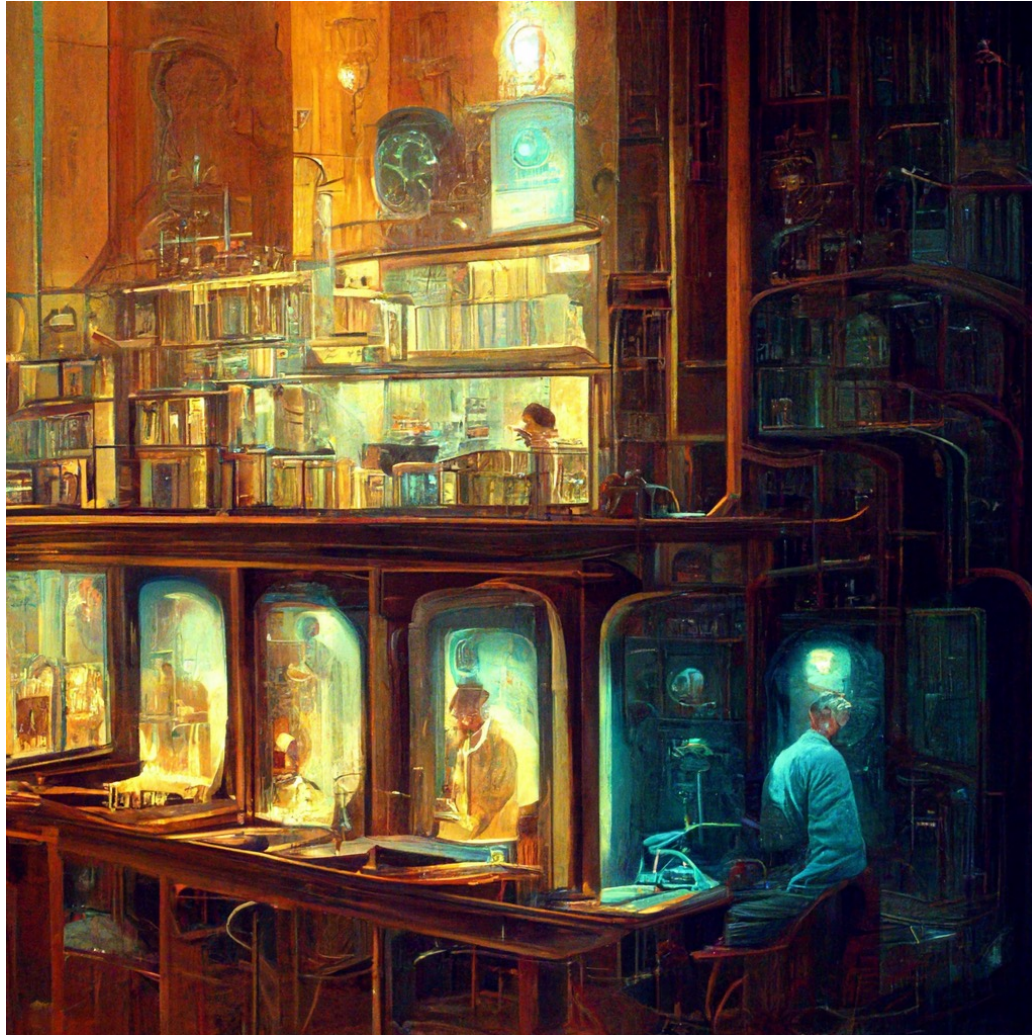
$$d = 6359.83 \arccos(\sin(x_1) \sin(x_2) + \cos(x_1) \cos(x_2) \cos(y_1 - y_2)).$$

# Point3D

- A data type to represent a point in 3D space
  - (x, y, z)
- Similar to Location and Die
- Alternate comparators for each axis

```
$ java Point3D
How many points? 3
Enter 9 doubles, separated by whitespace: -3 1 6 0 5 8 -5 -7 -3
Here are the points in the order entered:
  (-3.0, 1.0, 6.0)
  (0.0, 5.0, 8.0)
  (-5.0, -7.0, -3.0)
Sorted by their natural ordering (compareTo)
  (-3.0, 1.0, 6.0)
  (-5.0, -7.0, -3.0)
  (0.0, 5.0, 8.0)
Sorted by their x coordinate (xOrder)
  (-5.0, -7.0, -3.0)
  (-3.0, 1.0, 6.0)
  (0.0, 5.0, 8.0)
Sorted by their y coordinate (yOrder)
  (-5.0, -7.0, -3.0)
  (-3.0, 1.0, 6.0)
  (0.0, 5.0, 8.0)
Sorted by their z coordinate (zOrder)
  (-5.0, -7.0, -3.0)
  (-3.0, 1.0, 6.0)
  (0.0, 5.0, 8.0)
```

# Questions?



# Exercise Hints

- Casting to a Die object `Die otherDie = ((Die) other);`
- Die toString method should print 3 rows of 7 characters, each separated by a newline
  - i.e `5 = "* * \n * \n * *";`
  - `0 = "Not rolled yet";`
- Remember, you can access the instance variables of another object (so long as they're accessible)

```
public double distanceTo(Location other) {  
    double otherLat = other.lat;
```



- Great circle distance for Location.java

$$d = 6359.83 \arccos(\sin(x_1) \sin(x_2) + \cos(x_1) \cos(x_2) \cos(y_1 - y_2)).$$

- Euclidean distance for Point3D.java

$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}$$