Advanced OOP

BEFORE WE START

Talk to your neighbors:

Do you listen to any podcasts? If so, what are your favorites?

Instructor
Melissa Lin

TAs
Poojitha Arangam
Darel Gunawan
Colton Harris
Atharva Kashyap
Eesha Kunisetti

Audrey Lin
Di Mao
Steven Nguyen
Ben Wang
Jaylyn Zhang

Questions during Class?
Raise hand or send here
sli.do #cse122
Lecture Outline

• Announcements

• Constructors (cont.)

• Equals

• Larger Example (cont.)
Announcements

• C2 due tomorrow

• P3 released Friday
  - The last part of the assignment involves Junit Testing (we will talk about this in Wednesday’s lecture!)
  - Due August 13
  - No resubmissions

• Final exam reminders
  - Aug 16 and 18 at the normal lecture time and location
  - More logistics on course website
  - Sections from now on are focused on preparing you for the final!
Lecture Outline

• Announcements

• Constructors (cont.)

• Equals

• Larger Example (cont.)
(PCM) Constructor Syntax

```java
public Point(int initialX, int initialY) {
    x = initialX;
    y = initialY;
}
```

All fields should be initialized in the constructor(s)!

If we write any constructors, Java no longer provides one for us.
(PCM) this keyword

The `this` keyword refers to the current object in a method or constructor.

You can use it to refer to an object’s fields

```
this.x, this.y
```

You can use it to refer to an object’s instance methods

```
this.setX(newX)
```

You can use it to call one constructor from another

```
this(0, 0)
```
Lecture Outline

• Announcements

• Constructors (cont.)

• Equals

• Larger Example (cont.)
(PCM) Equals

The equals() method returns true if the given parameter is considered equal to this object, and false otherwise. Used by lots of library methods! (contains, remove for specific elements, etc.)

Each class has one provided by Java, but it checks for reference equality.

If you want equals to check for value equality, you need to write this method yourself.
Object

By taking a parameter of type `Object`, the equals method can be passed *any type of object*.

More to come in CSE 123 on the Java mechanisms that make this work!

We can use the `instanceof` keyword in Java to determine if the parameter is *actually* a `Point`
(PCM) Point’s equals()
Almost there...

This is actually still an imperfect implementation because we would also need to write a `hashCode()` method for our object to work with `HashSet`, `HashMap`, etc. but more to come on that in CSE 331 and beyond 😊
Lecture Outline

• Announcements

• Constructors (cont.)

• Equals

• Larger Example (cont.)
What is the best method comment for `getNewestSong()`?

A. Returns the last song by this Artist that was added to the list
B. Returns the newest song by this artist, or null if the artist has no songs
C. Returns the newest song by this artist
D. Loops through the list of songs by this Artist to find which one has the largest year and returns it
E. Returns the newest song by this artist, with ties being broken by which song was added earlier, or null if the artist has no songs.
What is the best method comment for `getNewestSong()`?

A. Returns the last song by this Artist that was added to the list
B. Returns the newest song by this artist, or null if the artist has no songs
C. Returns the newest song by this artist
D. Loops through the list of songs by this Artist to find which one has the largest year and returns it
E. Returns the newest song by this artist, with ties being broken by which song was added earlier, or null if the artist has no songs.
Think about the design of the Song and Artist classes. Here are guiding questions, but you are welcome to (and encouraged to) deviate from these topics!

• What are “valid” values for each of the fields?
  • Song: name, year
  • Artist: name, songs
• What are additional fields that we could include?
• Would it make sense to include additional constructors? If so, what constructors?
  • Example: a Song constructor that accepts only a name
• What would happen if we used a different data structure to store an Artist’s songs?
  • How would it affect the client perspective of an Artist?
  • What would we need to change about our code?
Design decisions

It’s up to you!

Think carefully about the decisions you make when designing your classes

- What assumptions are you making?
- How does this affect users?