

LEC 15

CSE 122**Collections**

BEFORE WE START

Talk to your neighbors:*What plans do you have for Thanksgiving break? Going anywhere fun?*Music: [122 24au Lecture Tunes](#) 🍂

	Elba Garza and Miya Natsuhara			
Instructors	Ayush	Heon	Harshitha	Aishah
TAs	Andrew	Izak	Marcus	Ben
	Logan	Colin	Carson	Ivory
	Kyle	Jessica	Jack	Cady
	Maggie	Shivani	Connor	Diya
	Nicole H	Ken	Cora	Katharine
	Caleb	Mia	Hannah	
	Nicole W	Ashley	Leo	
	Jacob	Chaafen	Any	


Questions during Class?

Raise hand or send here

sli.do #cse122



Lecture Outline

- **Announcements** 
- Optional
- Recap of Collections
- Dumb Data Structures
- Collections

Announcements

- Resubmission Cycle 5 (R5) out; due Nov 19th by 11:59 PM
- Programming Assignment 3 (P3) out tonight!
 - Due Nov 21st by 11:59 PM
- Quiz 2 Tuesday, Nov 19th
- Thanksgiving Week
 - Optional TA's Choice section on Tuesday, Nov 26th—highly recommend!
 - Some TAs will have section and some not. You can go to any and all!
 - Topics taught will vary widely; we'll let you know ahead of time so you can choose.
 - Optional Lecture on Wednesday, Nov 27th by our amazing TAs!

Lecture Outline

- Announcements
- **Optional** ◀
- Recap of Collections
- Dumb Data Structures
- Collections

Optional

`Optional` is a Java class that is used to handle situations where a value is sometimes there.

- A variable that can *sometimes* be initialized
- `Optional<String> keepPlaying = Optional.empty();`
- `Optional<Integer> maxValue = Optional.of(-1);`

Like a collection, `Optional` uses `<>` to denote the type it contains..

- e.g., `Optional<String>`, `Optional<Integer>`, `Optional<Point>`

Optional Methods

Method	Description
<code>Optional.empty()</code>	Creates an empty <code>Optional</code> object
<code>Optional.of(...)</code>	Creates an <code>Optional</code> object holding the object it's given
<code>isEmpty()</code>	Returns <code>true</code> if there <i>is no</i> value stored, and <code>false</code> otherwise
<code>isPresent()</code>	Returns <code>true</code> if there <i>is a</i> value stored, and <code>false</code> otherwise
<code>get()</code>	Returns the stored object from the <code>Optional</code> (if one is stored; otherwise throws a <code>NoSuchElementException</code>)

The `Optional` class has more than just these methods, but these are what you'll need to focus on for this class!

Note on Optional Methods

`isEmpty()`, `isPresent()`, and `get()` are called like normal instance methods (on an actual instance of `Optional`).

Example: `keepPlaying.isEmpty()`

`Optional.of(...)` and `Optional.empty()` are called differently (Like the `Math` class methods)

Example: `Optional.empty();`

Why Optional?

Using `Optional` can help programmers avoid `NullPointerException`s by making it explicit when a variable may or may not contain a value.

- Remember – `null` refers to the complete absence of an object!

There are other `Optional` methods (that you should explore in your own time if you're interested) that can be really useful to cleanly work with data that may or may not be present.

Student / Course Example one more time...

Let's add two more methods to `Course.java`:


```
public void setCourseEvalLink(String url)
```

```
public Optional<String> getCourseEvalLink()
```

The link to the evaluations for a course doesn't usually exist until the last few weeks of the quarter. What if a client calls `getCourseEvalLink` before one is set up?

`Optional` to the rescue! 

Lecture Outline

- Announcements
- Optional
- **Recap of Collections** 
- Dumb Data Structures
- Collections

Goal for Today

Review some of the data structures we've talked about this quarter

Understand how Java organizes them with *interfaces*

Collections: What *classes* have we seen so far?

...

Array,

ArrayList,

LinkedList,

Stack,

HashSet & HashMap,

TreeSet & TreeMap

Collections: What *interfaces* have we seen so far?

...


Set,

Queue,

List,

Comparable

Lecture Outline

- Announcements
- Optional
- Recap of Collections
- **Dumb Data Structures** 
- Collections


Dumb Data Structures

We're going to create our own versions of these classes so we can dig into how they all relate to each other!

BUT they're going to be real dumb.

If you want to get a sense of how they're *actually* implemented, go take CSE 123!

Lecture Outline

- Announcements
- Optional
- Recap of Collections
- Dumb Data Structures
- **Collections** 

IntCollection Relationships

