LEC 14

CSE 122

Comparable

Questions during Class?

Raise hand or send here

sli.do #cse122

BEFORE WE START

Talk to your neighbors:

What is your favorite way to eat a potato? 🥔

Respond on sli.do!

Instructor: Brett Wortzman

TAs:

Archit	Daniel	Naomi	Sreshta
Blake P.	David	Neha	Suyash
Blake R.	Hanna	Nicole	Trisha
Cady	Isis	Rio	Wesley
Carson	Jun	Rohan	Yash
Cole	Landon	Shreyank	
Connor	Michelle	Somya	

Music: 🎷 CSE 122 26au Lecture Tunes 🎶

Lecture Outline

- Announcements
- `Comparable` Review

Announcements

- CQ2 ends Friday
  - Only 30 slots left (and almost none Friday)
- P3 out Friday*, due **12/3**
  - May slip to the weekend
- No class next week
- PQ2 Tuesday, 12/1
- Gradebooks should be published this week (!!)

End-of-Quarter Logistics

- "Final exam" slot: **Tuesday, December 15, 12:30pm – 2:20pm**
  - Two parts:
    - PQ3 (normal-length paper quiz)
    - Paper Quiz clobbers: one problem per previous Paper Quiz (including PQ3)
      - Will replace lowest problem grade on the corresponding quiz (if improvement)
- Computer Quiz retakes: **Monday 12/14 through Wednesday 12/16**
  - Times TBA
    - Will need to sign up ahead of time
  - Can retake **one** full computer quiz
    - Must also resubmit corresponding project
  - Will *fully* replace previous CQ grades
  - CQ3 *may not* be graded before CQ retakes
    - Will still be allowed to retake CQ3

Lecture Outline

- Announcements
- `Comparable` Review

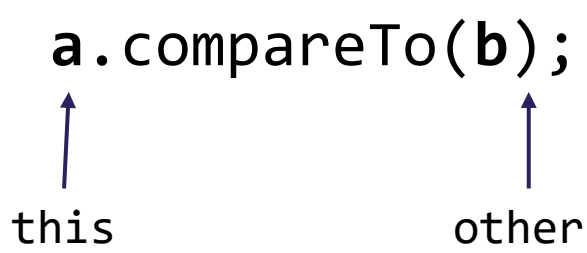
(PCM) Comparable

- `Comparable<T>` is an interface that allows implementers to define an ordering between two objects
  - Used by `TreeSet`, `TreeMap`, `Collections.sort`, etc.

- One required method:
  `public int compareTo(T other)`

- Returned integer falls into 1 of 3 categories
  - `< 0`: `this` is "less than" `other`
  - `= 0`: `this` is "equal to" `other`
  - `> 0`: `this` is "greater than" `other`

- Actual value returned ***does not matter***
  - Only ever test sign, not actual value



(PCM) Subtraction Trick

- `compareTo` implementation when comparing two integers ascending:

```
if (this.field < other.field)       -> negative number
else if (this.field > other.field)  -> positive number
else                                -> 0
```

- This is just subtraction!

```
this.field - other.field
```

- What if we wanted to sort descending?

```
other.field - this.field
```

- **Warning**: this only works for integers! Doubles have issues with truncation.

Reminder: `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! e.g. `contains`, `remove` for specific elements, etc.

Each class has a default implementation that checks for **reference equality**.

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

Guidelines for `compareTo`

- if `a.compareTo(b)` is positive, then `b.compareTo(a)` is negative (and vice versa)
- if `a.compareTo(b)` returns `0`, then `b.compareTo(a)` returns `0`
- if `a.compareTo(b)` returns `0`, then `a.equals(b)` returns `true` (and vice versa)

Classes can Implement Multiple Interfaces

A class can implement multiple interfaces – it's like one person signing multiple contracts!

If a class implements an interface A and an interface B, it must include *all* of A's required methods **and** *all* of B's required methods