

LEC 09

CSE 122

Introduction to Objects

Questions during Class?
Raise hand or send here

sli.do #cse122



BEFORE WE START

Talk to your neighbors:

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** 
- Review: OOP
- Abstraction
- Building a Shopping Cart

Announcements

- CQ1 has begun
 - *Sign up early– slots fill!*
 - *Same-day changes **not allowed**; email Brett if an emergency*
- CQ0 grades out; PQ0 grades coming next week

Mid-Quarter Feedback

- Good things!
 - PCM/Lecture/Section triad
 - TAs
 - IPL
 - AI policy

Mid-Quarter Feedback

- Areas for improvement
 - Computer Quiz length
 - Already working on it— see note in CQ0 grade post
 - More practice quizzes
 - Unlikely to get more than 1/quiz this quarter, but will build over time
 - Make sure to use them well!
 - Gradebook
 - On the way

Mid-Quarter Feedback

- Mixed results
 - Pace of lecture
 - Also amount of review/recap
 - Quiz/Practice alignment
 - Spec clarity

Lecture Outline

- Announcements
- **Review: OOP** 
- Abstraction
- Building a Shopping Cart

(PCM) Object Oriented Programming (OOP)

- **Procedural programming:** Programs that perform their behavior as a series of steps to be carried out
 - Classes that do things
- **Object-oriented programming (OOP):** Programs that perform their behavior as interactions between objects
 - Classes that represent things
 - We're going to start writing our own objects!

(PCM) Classes & Objects

- **Classes** can define the template for an object
 -  Like the blueprint for a house!
“What does it mean to be this thing?”
- **Objects** are the actual instances of the class
 -  Like the actual house built from the blueprint!
“It is an example of this thing!”

We create a new instance of a class with the **new** keyword
e.g., `Scanner console = new Scanner(System.in);`

(PCM) Object State & Behavior

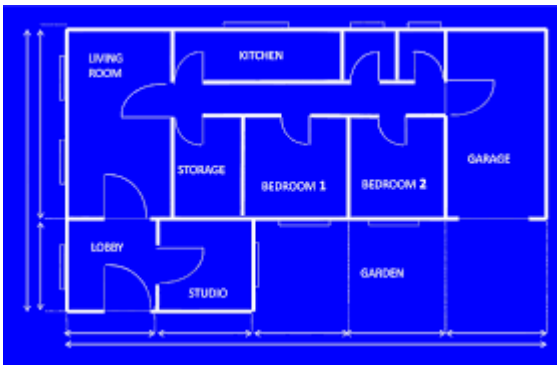
- **Objects** can tie related *state* and *behavior* together
- **State** is defined by the object's *fields* (aka *instance variables*)
 - *Scanner's state may include what it's scanning, where it is in the input, etc.*
- **Behavior** is defined by the object's *instance methods*
 - *Scanner's behavior includes "getting the next token and returning it as an int", "returning whether there is a next token or not", etc.*
 - *...Next time!*

(PCM) Object Class Syntax

```
public class MyObject {  
    // fields (or instance variables)  
    type1 fieldName1;  
    type2 fieldName2;  
    ...  
  
    // instance methods  
    public returnType methodName(...) {  
        ...  
    }  
    ...  
}
```

(PCM) Instance Variables

- Fields are also referred to as **instance variables**
- Fields are defined in a class, where each instance of the class has their own copy of the fields
 - Hence *instance* variable! It's a variable tied to a **specific** instance of the class!



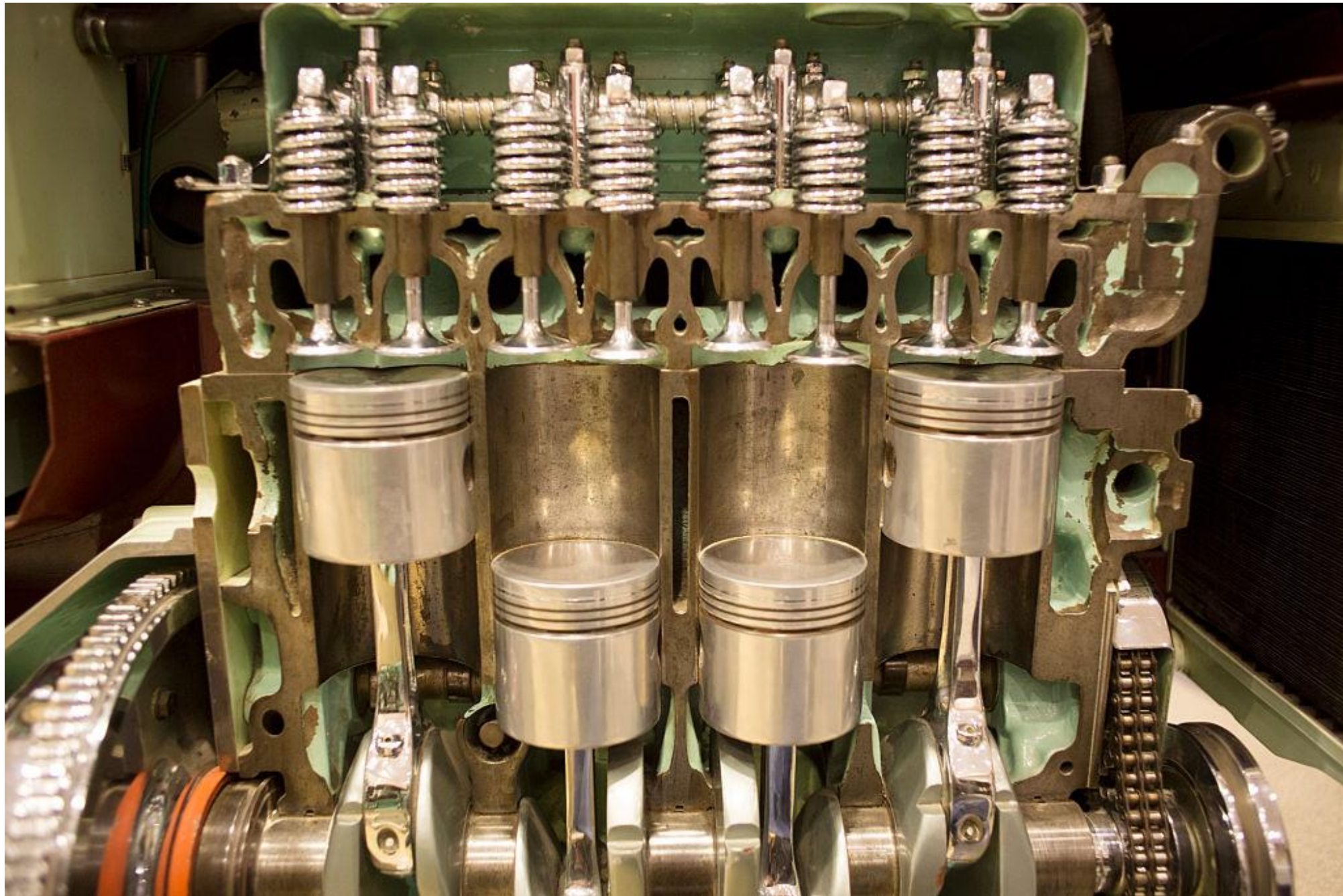
(PCM) Instance Methods

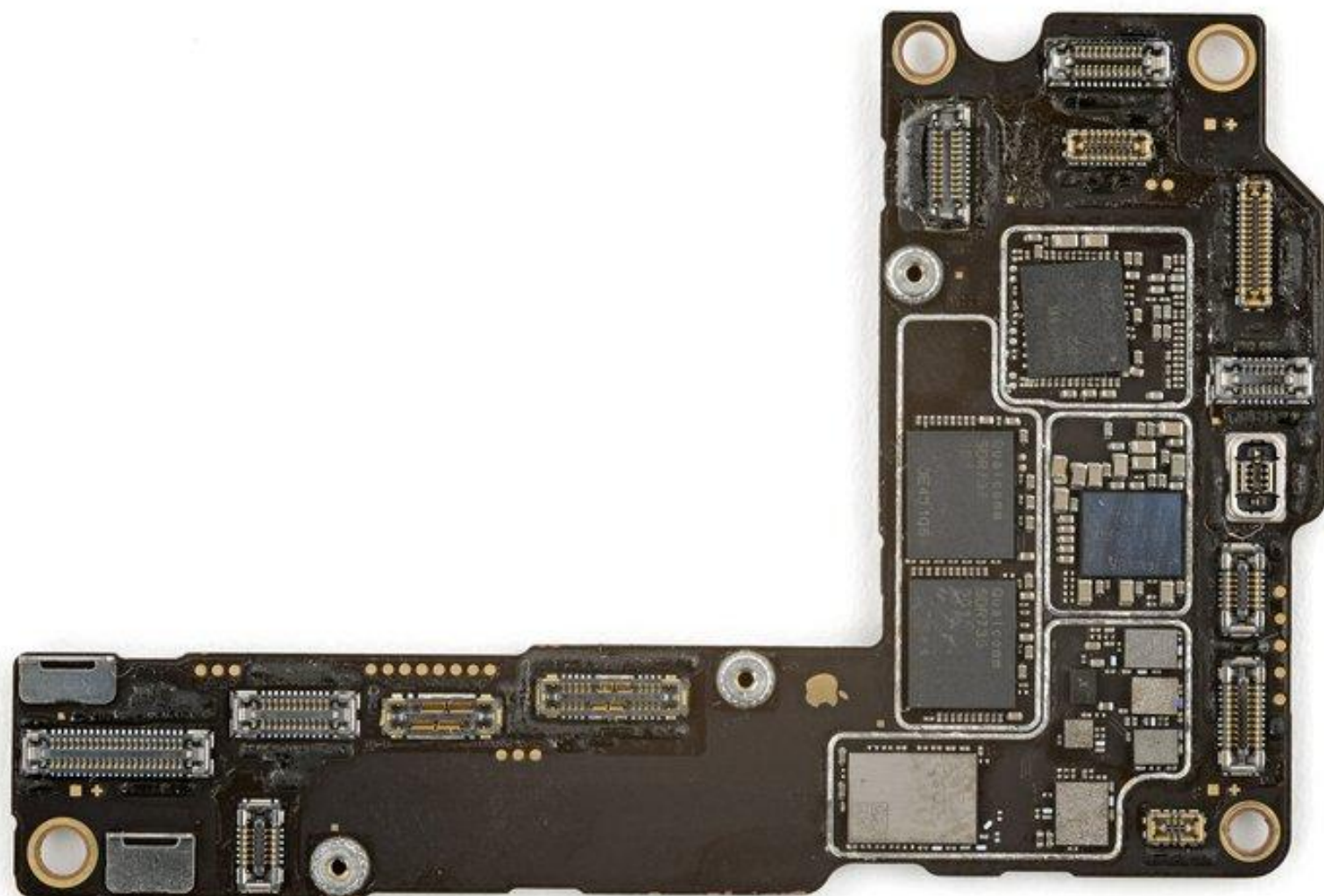
- **Instance methods** are defined in a class
- Calling an instance method on a particular *instance* of the class will have effects only on that instance
- *More on these later...*



Lecture Outline

- Announcements
- Review: OOP
- **Abstraction** ◀
- Building a Shopping Cart





Abstraction

The separation of ideas from details, meaning that we can use something without knowing exactly how it works.

You were able use the Scanner class without understanding how it works internally!

Client v. Implementor

We have been the clients of many objects this quarter!

Now we will become the implementors of our own objects!

Lecture Outline

- Announcements
- Review: OOP
- Abstraction
- **Building a Shopping Cart** ◀