- **Previous lecture:**
  - Review cell and struct arrays
  - Introduce objects
- **Today’s lecture:**
  - Introduction to Matlab handle objects—plots
  - Introduction to objects and classes
- **Announcements:**
  - Discussion in computer lab this week
  - Project 5 due Friday at 11pm; extra consulting hours Friday 3:45-6:30pm in ACCEL Green Rm (Carpenter Hall)
  - Prelim 2 on Tues, Apr 26 at 7:30pm
  - Prelim 2 topics: end with Project 5 and Lecture and discussion of previous week; will NOT include OOP

---

**Object-Oriented Programming**

- First design and define the **classes** (of the objects)
  - Identify the properties (data) and actions (methods, i.e., functions) of each class
- Then create the **objects** (from the classes) that are then used, that interact with one another

---

**Example class: Rectangle**

- **Properties:**
  - xLL, yLL, width, height
- **Methods (actions):**
  - Calculate area
  - Calculate perimeter
  - Draw
  - Intersect (the intersection between two rectangles is a rectangle!)

---

**Example class: Time**

- **Properties:**
  - Hour, minute, second
- **Methods (actions):**
  - Show (e.g., display in hh:mm:ss format)
  - Advance (e.g., advance current time by some amount)

---

Matlab supports procedural and object-oriented programming

- We have been writing **procedural programs**—focusing on the algorithm, implemented as a set of functions
- We have used objects in Matlab as well, e.g., graphics
- A plot is a “handle graphics” object
  - Can produce plots without knowing about objects
  - Knowing about objects gives more possibilities

---

**The plot handle graphics object in Matlab**

\[
x = ...; y = ...; 
\text{plot}(x, y)
\]

- **In the past we focused on the visual produced by that command. If we want the visual to look different we make another plot.**
- **We can actually hang on to the graphics object—store its “handle”—so that we can later make changes to that object.**
A “movie” of a ball tracing a curve—3 frames

Without considering objects: draw the three “frames" independently, each time drawing both line and ball.

Make use of objects: make the line and ball two independent objects that we can hold on, i.e., we hold on to their individual handles. Leave the line alone, but at each iteration give the ball new x, y-values.

Objects of the same class have the same properties

- Both objects have some x-data, some y-data, some line style, and some marker style. These are the properties of one kind, or class, of the objects (plots).
- The values of the properties are different for the individual objects.

```matlab
x = 1:10;
% Two separate graphics objects:
plot(x, sin(x), 'k-')
plot(x(1:5), 2.^x, 'm-*')
```

Given class Interval (file Interval.m) ...

```matlab
% Create 2 Intervals, call them
A= Interval(2,4.5)
B= Interval(-3,1)
% Assign another right end point
A.right= 14
% Half the width of A (scale by 0.5)
A.scale(.5)
% See the result
disp(A.right) % show value in right property in A
disp(A) % show all property values in A
```

Class Interval

- An interval has two properties:
  - left, right
- Actions—methods—of an interval include
  - Scale, i.e., expand
  - Shift
  - Add one interval to another
  - Check if one interval is in another
  - Check if one interval overlaps with another

See demoInterval0.m

Class Interval

- An interval has two properties:
  - left, right
- Actions—methods—of an interval include
  - Scale, i.e., expand
  - Shift
  - Add one interval to another
  - Check if one interval is in another
  - Check if one interval overlaps with another

```matlab
classdef Interval < handle
    properties
        left
        right
    end
    methods
        function  scale(self, f)
            . . .
        end
        function  shift(self, s)
            . . .
        end
        function  Inter = overlap(self, other)
            . . .
        end
        function  Inter = add(self, other)
            . . .
        end
        . . .
    end
end
```

To specify the properties and actions of an object is to define its class.

An Interval object

The “handle” or “reference” of the object.

- The “constructor” method
  - The object is also called an “instance” of a class. It contains every property, “instance variable,” and every “instance method” defined in the class.
Multiple Interval objects

```matlab
classdef Interval < handle
    properties
        left
        right
    end
    methods
        function Inter = Interval(lt, rt)
            Inter.left = lt;
            Inter.right = rt;
        end
    end
end
```

Every object instance contains every "instance variable" and every "instance method" defined in the class. Every object has a unique handle.

A handle object is referenced by its handle

```matlab
p = Interval(3,7); % p references an Interval object
r = Interval(4,6); % r references an Interval object
p = Interval(3,7); % p references an Interval object
r = Interval(4,6); % r references an Interval object
```

What is the effect of referencing?

```matlab
p = Interval(3,7); % p references an Interval object
s = p; % s stores the same reference as p
s.left = 2; % change value inside object
disp(p.left) % 2 is displayed
```

The constructor method

To create an Interval object, use its class name as a function call:

```matlab
p = Interval(3,7)
```

A handle, also called a reference, is like an address; it indicates the memory location where the object is stored.

What is the effect of referencing?

```matlab
p = Interval(3,7); % p references an Interval object
s = p; % s stores the same reference as p
s.left = 2; % change value inside object
disp(p.left) % 2 is displayed
clear p % get rid of p from memory
```

The object is not copied—no new object is created! s and p both reference the same object.