• Previous lecture:
  – Why use OOP?
  – Attributes for properties and methods

• Today’s lecture:
  – Inheritance: extending a superclass
  – Overriding methods in superclass
  – Next topic: Recursion (read §14.1 before next lecture)

• Announcements:
  – Prelim 2 feedback on Canvas
  – Project 6 coming this week: OOP & recursion
Can we get all the functionality of Die in TrickDie without rewriting all the Die code in class TrickDie?

```matlab
classdef Die < handle
    properties (Access=private)
        sides=6;
        top
    end
    methods
        function D = Die(...) ...
        function roll(...) ...
        function disp(...) ...
        function s = getSides(...) ...
        function t = getTop(...) ...
    end
    methods (Access=private)
        function setTop(...) ...
    end
end

% "Inherit" the components of class Die

classdef TrickDie < handle
    properties (Access=private)
        favoredFace
        weight=1;
    end
    methods
        function D = TrickDie(...) ...
        function f = getFavoredFace(...) ...
        function w = getWeight(...) ...
    end
end
```
Yes! Make TrickDie a **subclass** of Die

```matlab
classdef Die < handle
    properties (Access=private)
        sides=6;
        top
    end
methods
    function D = Die(...)  ...
    function roll(...)  ...
    function disp(...)  ...
    function s = getSides(...)  ...
    function t = getTop(...)  ...
end
methods (Access=protected)
    function setTop(...)  ...
end
end

classdef TrickDie < Die
    properties (Access=private)
        favoredFace
        weight=1;
    end
methods
    function D = TrickDie(...)  ...
    function f=getFavoredFace(...)  ...
    function w = getWeight(...)  ...
end
end
```
Inheritance

Inheritance relationships are shown in a *class diagram*, with the arrow pointing to the parent class.

An *is-a* relationship: the child *is a* more specific version of the parent. Eg., a trick die *is a* die.

*Multiple* inheritance: can have multiple (direct) parents ← e.g., Matlab

*Single* inheritance: can have one (direct) parent only ← e.g., Java

If relationship is “has a” or “can do”, prefer *composition* to inheritance
Inheritance vocabulary

• Allows programmer to derive a class from an existing one

• Existing class is called the parent class, or superclass

• Derived class is called the child class or subclass

• The child class inherits the (public and protected) members defined for the parent class

• Inherited trait can be accessed as though it was locally defined
Which components get “inherited”?

• **public** components get inherited

• **private** components exist in object of child class, but cannot be **directly accessed** in child class ⇒ we say they are **not** inherited

• Note the difference between **inheritance** and **existence**!
Which components get “inherited”?

• **public** components get inherited

• **private** components **exist** in object of child class, but cannot be **directly accessed** in child class ⇒ we say they are **not inherited**

• Note the difference between **inheritance** and **existence**!
**protected attribute**

- Attributes dictate which members get inherited

- **private**
  - Not inherited, can be *accessed* by *local* class only

- **public**
  - Inherited, can be *accessed* by *all* classes

- **protected**
  - Inherited, can be *accessed* by *sub*classes

- **Access**: access as though defined locally

- All members from a superclass *exist* in the subclass, but the *private* ones cannot be *accessed* directly—can be accessed through inherited (public or protected) methods
>> d = Die(6);
>> td = TrickDie(2, 10, 6);
>> %... more code in Command Window ...

A. d.setTop(3) and td.setTop(3) both work

B. Neither d.setTop(3) nor td.setTop(3) works

C. d.setTop(3) works but td.setTop(3) doesn’t
Must call the superclass’ constructor

• In a subclass’ constructor, call the superclass’ constructor before assigning values to the subclass’ properties.

• Calling the superclass’ constructor cannot be conditional: explicitly make one call to superclass’ constructor

See constructor in TrickDie.m
Overriding methods

- Subclass can *override* definition of inherited method
- New method in subclass has the same name (but has different method body)

See method `roll` in `TrickDie.m`
Overridden methods: which version gets invoked?
To create a TrickDie: call the TrickDie constructor, which calls the Die constructor, which calls the roll method. Which roll method gets invoked?
Overriding methods

• Subclass can override definition of inherited method
• New method in subclass has the same name (but has different method body)
• Which method gets used??

*The object that is used to invoke a method determines which version is used*
• Since a TrickDie object is calling method `roll()`, the TrickDie’s version of `roll()` is executed
• In other words, the method most specific to the type (class) of the object is used
Accessing superclass’ version of a method

- Subclass can override superclass’ methods
- Subclass can access superclass’ version of the method

```matlab
classdef Child < Parent
    properties
        propC
    end
    methods
        ...
        function x = method(arg)
            y = method@Parent(arg);
            x = ... y ...;
        end
        ...
    end
end
```

See method `disp` in `TrickDie.m`
Important ideas in inheritance

• Keep common features as high in the hierarchy as reasonably possible
• Use the superclasses’ features as much as possible
• “Inherited” $\Rightarrow$ “can be accessed as though declared locally”
  (private member in superclass exists in subclasses; they just cannot be accessed directly)
• Inherited features are continually passed down the line
A cell array can reference objects of different classes

```matlab
A{1} = Die();
A{2} = TrickDie(2,10);  % OK
```

A simple array can reference objects of only one single class

```matlab
B(1) = Die();
B(2) = TrickDie(2,10);  % ERROR
```
Anatomy of object array syntax

A = {};  % Optional

Array initialization
A{1} = Die();
A{2} = TrickDie();

A{1}.sides  % Error: private property

Array indexing  Property access
A{2}.roll()

Array indexing  Method invocation
End of Matlab OOP in CS1112

OOP is a concept; in different languages it is expressed differently.

In CS (ENGRD) 2110 you will see Java OOP
OOP in computing culture