Announcements

• PS 9 is ready on the web

• Try to get some help from me and tutors

• Reading assignment for this slide set: my notes
Subtyping
Interface inheritance
java.lang.Object class

- Any Java class inherits `Object` as its parent class
- `Object` is a superclass of any class that we create
- `Point` is a subclass of `Object`
- We also say `Object` is supertype `Point`; `Point` is a subtype of `Object`
- A variable of supertype can hold a subtype object
- A variable of subtype cannot hold a supertype object
Subtype polymorphism (subtyping)

- In general, we can substitute a subtype object for a supertype variable: *substitution principle* (subtyping is the theoretical basis that makes this possible)

- Declared type vs. actual type
  
  Object o1; // declared type of o1 is Object
  o1 = new Point(1, 2); // actual type of o1 is Point

  - The assignment above is possible because of subtyping

- See Subtype.java and Point.java
Inheritance

• Interface inheritance
  • Subtype inherits only the interface of its supertype

• Implementation inheritance
  • Subtype inherits both interface and implementation of its supertype
Interface inheritance

- An interface is really a new type that we are defining, much like the types that get created when we define classes.
- We can use the interface name, **Shape** here, as the type of a variable. This also means that if a class ‘implements’ this interface, the class MUST implement ALL the methods that are declared in the interface to be able to act as a complete class.
- The methods declared in an interface are declared as *abstract*. These abstract methods become *concrete* when they are implemented in a class that inherits the interface.

See **Shape.java**, **Circle.java**, **Rectangle.java**, **Box.java**, **Point.java**, **UseShape.java**
Sorting and searching (revisited)

- Now that we understand interfaces, let us redo sorting an array of objects using the `Comparable` interface.

- See sorting_objects_2/Point.java
  - Selection.java
  - ArrayTools.java