Previous Lecture:
  - static variables and methods
  - Polymorphism

Today’s Lecture:
  - Class Object
  - abstract class

Reading:
  - Sec 4.3
Recap—extending a class

- Subclass is a more specific version of the superclass
- Subclass inherits public and protected members from superclass
- You can declare new variables and methods in subclass
- You can override inherited methods with new definition
- Do not re-declare variables in subclass
  Called shadowing—not good practice in general
- Do not re-declare static components
Accessing methods/variables through polymorphic references

Ask two questions:

1. What determines whether a method/variable can be accessed?
   - reference type

2. For an overridden method, what determines which version gets invoked?
   - object type
The Object class

- If a class is not explicitly defined to be the child of an existing class, it is assumed to be the child of the Object class
  ⇒ All classes are derived from the Object class

```
class Room
  is the same as

class Room extends Object
```
The **Object class**

- If a class is not explicitly defined to be the child of an existing class, it is assumed to be the child of the **Object class**

⇒ All classes are derived from the **Object class**
The Object class

- If a class is not explicitly defined to be the child of an existing class, it is assumed to be the child of the Object class
  ⇒ All classes are derived from the Object class

- `.toString`: “default” instance method defined in the Object class
- Arrays are Objects, literally!
abstract class

- A placeholder in a class hierarchy that represents a generic concept
- Cannot be instantiated
- Modifier: abstract

```
public abstract class Geometry
```

- Can contain abstract methods

```
public abstract double Area();
```

- Subclasses of abstract classes will “fill out” these abstract methods
What we learned...

- Develop/implement *algorithms* for problems
- Develop programming skills
  - Design, test, debug, document, demonstrate
- Apply the Java programming language
  - Fundamentals of object oriented programming
  - Inheritance
  - Control structures
  - Methods for reducing redundancy
  - Data structure
What we learned... (cont’d)

- Specific tasks
  - Sorting
  - Simulation of systems
  - Text and string processing