Polymorphism

Polymorphism

- The Essence of good Object Oriented Programming
- polymorphism (from Greek πολύς, polys, "many, much" and μορφή, morphē, "form, shape")

Object Oriented Programming

- Encapsulation
  - All the information about an object is kept together (and some of it is private)
- Inheritance
  - Can reuse objects and tailor their behaviour (inherit the parent's functions and data and modify the ones you want)
- Polymorphism
  - Treat different items of data in the same way

Inheritance

- Example
  - Dog inherits from Animal

```
class Animal
{
    void sleep()
    {
        System.out.println("zzz");
    }
}

class Dog extends Animal
{
    void talk()
    {
        System.out.println("woof!");
    }
}
```

Inheritance

- Overriding a method

```
class Animal
{
   void sleep()
   {
      System.out.println("zzz");
   }
}

class Dog extends Animal
{
   void sleep()
   {
      System.out.println("snore!");
   }
}
```

Understanding Polymorphism

- From Examples
  - Graphics Program
    - Many different type of Object, Rectangle, Circle, etc.
    - All these objects are subclasses of Shape
      - Shape has a move method
    - All the other objects override the move method.
    - When the user wants to move a set of selected objects together, the code is
      - for(int i = 0; i < length; i++)
        selected[i].move();
      - The objects all move correctly

Even though the items in the selected array could be quite different, the program doesn't care. It just calls the move method and polymorphism ensures it all works

Polymorphism

- Consider

```
Animal ani = new Animal();
Animal fido = new Dog();

Animal [] menage = {ani, fido};

for(int i = -; i < ani.length; i++)
   menage[i].sleep();
```

This code prints:

zzz
snore!

- Even though fido is an element in an array of animals, he is still a Dog and when the sleep() method is called, the Dog's sleep method is called.

That's it!

- You know know all you need to know about Polymorphism!

Except for one small thing

- Sometimes the base class may not make any objects
  - It might only be used as a basis to make other classes.
  - In this case, it is usually declared abstract
    - E.g.

```
public abstract class Animal
{
    public sleep()
    {
        System.out.println("zzz");
    }
}
```

Summary

- Object Oriented Programming
  - Encapsulation, inheritance, Polymorphism
- Treat a set of different objects in the same way
- Abstract classes