

CISC 3115

Polymorphism and “Generic” Methods

Hui Chen

Department of Computer & Information Science

CUNY Brooklyn College

Outline

- Discussed
 - Inheritance
 - Superclass/supertype, subclass/subtype
 - Inheritance and constructors in Java
 - Inheritance and instance methods in Java
 - The Object class in Java
- Polymorphism
 - What is it? What benefits are there?
 - How do we design programs using it?
 - Writing generic method
 - Actual type vs. declared type

Polymorphism

- A variable of a supertype can refer to a subtype object
- Example: what would be the output?

```
GeometricObject object;
```

```
.....
```

```
System.out.println("Created on "  
    + object.getDateCreated()  
    + ". Color is " + object.getColor());
```

Question: What would be the Output?

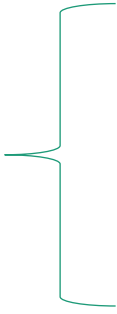
```
GeometricObject object;
```

```
.....
```

```
System.out.println("Created on "
```

```
    + object.getDateCreated()
```

```
    + ". Color is " + object.getColor());
```



We don't know yet
before we know
this!

Example: What would be the Output?

```
GeometricObject object;
```

```
object = new Circle(100, "red", true);
```

```
System.out.println("Created on "
```

```
    + object.getDateCreated()
```

```
    + ". Color is " + object.getColor());
```

```
object = new Rectangle(100, 100, "blue", true);
```

```
System.out.println("Created on "
```

```
    + object.getDateCreated()
```

```
    + ". Color is " + object.getColor());
```

Now we know.

Now we know.

Writing “Generic” Method

- Since a subclass “is-a” a superclass, we can write a method with a parameter of the superclass type.
- The method can take argument of any subclass, thus we say this method is “generic”
- But a superclass “is-not-a” subclass!

Example: ShapeClient

- What's the output?

```
public class ShapeClient {  
    /** Main method */  
    public static void main(String[] args) {  
        // Display circle and rectangle properties  
        displayShapeObject(new Circle(1, "red", false));  
        displayShapeObject(new Rectangle(1, 1, "black", true));  
    }  
  
    /** Display geometric object properties */  
    public static void displayShapeObject(GeometricObject object) {  
        System.out.println("Created on " + object.getDateCreated() +  
            ". Color is " + object.getColor());  
    }  
}
```

Actual Type and Declared Type

- Declared type: data type known at compilation time
- Actual type: data type known at runtime
 - A variable may refer to an object of different type at runtime
 - Example: what are actual and declared types of “ben”, and “adam”?

```
Person ben = new Person("Ben Franklin", "00124", "2901 Bedford Ave");  
Person adam = new Student("Adam Smith", "00248", "2902 Bedford Ave")
```


Questions?

- Concept of polymorphism
- Writing generic methods
- Declared type and actual type

Exercise

- Complete the Shape hierarchy with the following classes
 - GeometricObject, Circle, and Rectangle as shown in the examples discussed
 - Add a GeometricObjectClient class where you implement the following “generic methods”:
 - `public static void displayShapeObject(GeometricObject object)`
 - `public static void printArea(GeometricObject object)`
 - Revise the GeometricObjectClient’s main method to create a circle and a rectangle object, and display the objects along with their areas.