Computer Science 22: Object Oriented Programming

Lecture #14: Polymorphism II

In This Lecture

- Demo: Parametric Polymorphism
- Demo: Subtype/Inclusion Polymorphism
- Demo: Ad hoc Polymorphism
- Typecasting
- **`instanceof`** operator

PARAMETRIC POLYMORPHISM

SUBTYPE/INCLUSION POLYMORPHISM

AD HOC POLYMORPHISM

Typecasting

- An object can be typecast reference into another object reference
- The type of one object is converted to match the type of another object reference
- Example:
  - `String s = new String();`
  - `Object o = (Object) s; // String type cast as Object`
- The cast must be to its **own class** or to **subclass types** or to **superclass types** or **interfaces**

Typecasting

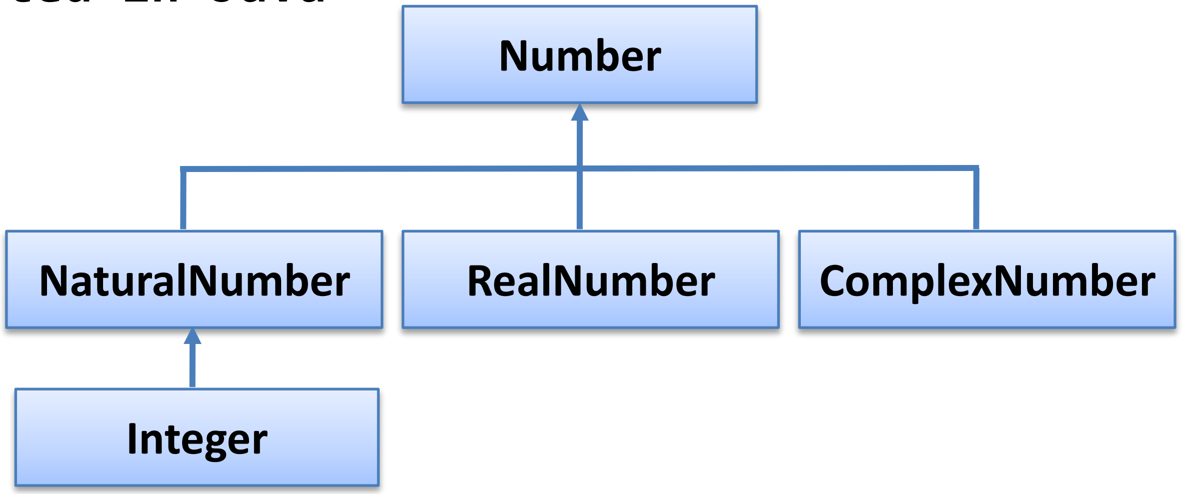
- Downcasting
  - Casting from a base class to its subclasses
- Upcasting
  - Casting from subclass towards superclass (up the hierarchy)
  - No typecast operator required
- Typecasting in Java
  - May produce ClassCastException
  - Can be applied to Primitives
    - i.e., int to char, char to int
    - i.e., float to int, int to float

Typecasting

```
NaturalNumber n = (NaturalNumber)(new Number());
//downcasting, not supported in Java?

Number n = new NaturalNumber();
// upcasting, supported in Java

Number n = new Integer();
// upcasting, supported in Java
```



`instanceof` Operator

- Binary operator requiring an object reference (first operand) and the type (either a Class or Interface) as the second
- Examples:

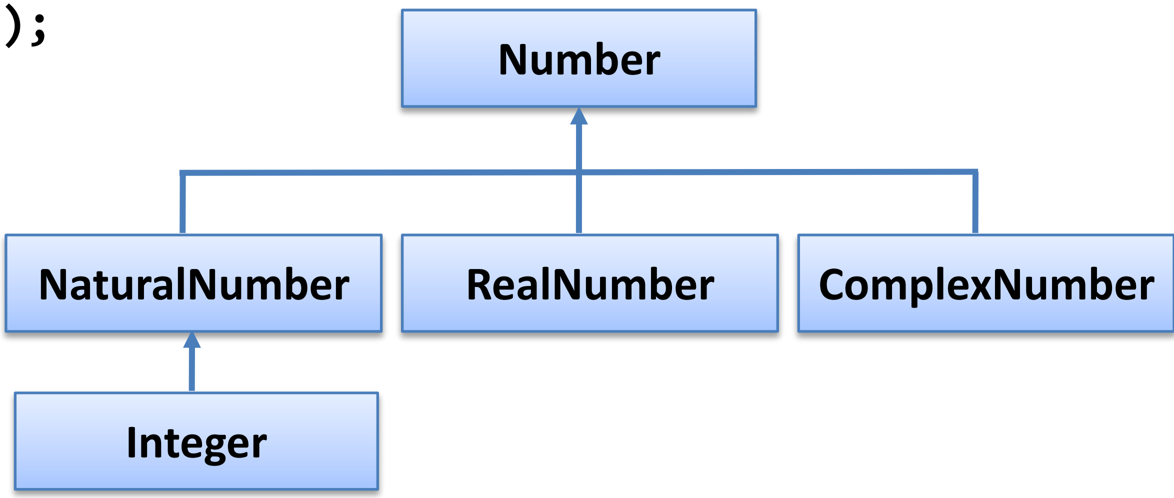
```
String s = new String();
boolean b = s instanceof String;

if (s instanceof CharSequence) {…}
```

`instanceof` Operator

```
Number n = new Number();
NaturalNumber nn = new NaturalNumber();
RealNumber rn = RealNumber();
ComplexNumber cn = new ComplexNumber();
Integer i = new Integer();
```

```
n instanceof Number
nn instanceof Number
rn instanceof Number
cn instanceof Number
i instanceof Number
i instanceof Integer
i instanceof ComplexNumber
```


Number
NaturalNumber
RealNumber
ComplexNumber
Integer