GEOG 178/258: Conceptual Modeling and Programming for the Geo-Sciences

TA: Rui Zhu
Week 4: Inheritance and Delegation
Review on Inheritance

• Hierarchical classes: super-class and subclass
  (e.g. watercourse is the super-class;
   rivers, stream, and channels are subclass)

• Keywords:
  1. `extends`: the keyword used to inherit the properties of a class;

```java
class Super{
    
    
}
class Sub extends Super{
    
    
}
```
Review on Inheritance

• Note:

A subclass inherits all the members (variables, methods, and nested classes) from its superclass.

Constructors are not members, so they are not inherited by subclasses, but the constructor of the superclass can be invoked from the subclass.
Review on Inheritance

• Keywords (cont.):
  2. super: the keyword used to inherit the properties of a class;

→ invoke the superclass constructor from subclass.

  super(values);

→ differentiate the members of superclass from the members of subclass, if they have same names;

  super.variable
  super.method();
Review on Delegation

By using delegation, you are simply calling up some class which knows what must be done. You do not really care how it does it, all you care about is that the class you are calling knows what needs doing.  


Delegate Methods

• Use the ArrayList in your models

```java
import java.util.ArrayList;
import java.util.Iterator;

public class Polyline {
    private ArrayList<Point> points;

    public boolean add(Point e) { return points.add(e); } // Delegate methods
    public Polyline() { points = new ArrayList<Point>(); }
    public void printAll(){
        Iterator<Point> iterator = points.iterator();
        while(iterator.hasNext())
            System.out.println(iterator.next());

        /* This is possible because Java understands * that it will always get a Point object */
        // System.out.println(iterator.next().getX());
    }

    public static void main(String[] args) {
        Polyline poly = new Polyline();
    }
}
Wrap up!
Hints for the HW4!

ToDo (Due Tuesday 9am)

- Implement an evacuation application.
  - Create 3 building classes, e.g., Hospital, Farm, Store
  - Create a Flood class in addition to the Wildfire class
  - Create a common super class for the building classes and one for the disaster classes
  - Your application should be able to check which buildings should be evacuated in case of different disasters
  - Read chapter 6 and 7; explain (as a text) the difference between call-by-reference vs. call-by-value

- Upload a zip file [LN1LN2W4.zip] with the * .java files to GauchoSpace
Wrap up!
Hints for the HW4!

<table>
<thead>
<tr>
<th>Inheritance</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Super-class</strong></td>
<td>Building</td>
<td>Disaster</td>
</tr>
<tr>
<td><strong>Sub-class</strong></td>
<td>Hospital; Farm; Store</td>
<td>Flood; Wildfire</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Delegation</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Building</td>
<td>ArrayList&lt;Points&gt;</td>
<td></td>
</tr>
<tr>
<td>Disaster</td>
<td>ArrayList&lt;Points&gt;/BoundingBox</td>
<td></td>
</tr>
</tbody>
</table>
Practice

Task:
Watercourse has three types: creek, river and channel.

They have common characteristics like:
(1). they are all represented as connections of several straight line segments, which are represented by polylines; → ArrayList<Polyline> watercourse= new ArrayList<Polyline>();
(2). they have average width and average depth; → double width; double depth
(3). the total length of the watercourse is the sum of each segment's length.
→ double totalLength(){...}
(4). their water volume could be calculated by using the formula:
watervolume=total_length* average_width*average_depth
→ double waterVolume(){...}

They are different in terms of:
The width of creek ranges from (0, 25); the width of river ranges from [25, 60); the width of channel ranges from [60, Positive infinity).
→ the constructors are different for these three!
Using Java to build this watercourse model!

**In the test class:**
(1). create three instances for each category,
(2). calculate and print out both total length and the water volume for each.

Note: this is also the workflow for working out your HW4!