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

TA: Rui Zhu
Week 3: Dubug, ArrayList and Iterator (Wildfire Model as example)
Review on debug

- **Create break points:** A breakpoint is a location in your source code that will halt execution so that you can view the information about a program's state.

- **Watch expressions:** Use it to keep track of complex expressions.

- **Inspect variables:** Use it to track the value of variables.
Review on ArrayList

- Dynamic arrays that can grow as needed
- Import via: `import java.util.ArrayList;` (or Ctrl+Shift+O)
- `ArrayList<Building> b = new ArrayList<Building>();`
- Methods:
  - `void add(int index, Object element)`
  - `boolean add(Object o)`
  - `Object remove(int index)`
  - `void clear()`
  - `boolean contains(Object o)`
  - `Object get(int index)`
  - `int indexOf(Object o)`
  - `Iterator iterator()`
  - ...

(http://docs.oracle.com/javase/6/docs/api/java/util/ArrayList.html)
Review on Iterator

- Iterator enables you to cycle through a collection (e.g. ArrayList), obtaining or removing elements.
- Import via: `import java.util.Iterator`
- `Iterator<Point> iterator = points.iterator()` (this iterator starts from the beginning of the collection)
- **Methods:**
  - `boolean hasNext()`
  - `Object next()`
  - `void remove()`

(https://docs.oracle.com/javase/7/docs/api/java/util/Iterator.html)
Review on Iterator (continue)

- How to use Iterator?
  
  In general, follow three steps to cycle through the contents of a collection:

  (1). generate an iterator to the start of the collection by calling the collection's `iterator()` method.

  (2). Set up a loop that makes a call to `hasNext()`. Have the loop iterate as long as `hasNext()` return true.

  (3). Within the loop, obtain each element by calling `next()`.

  http://www.tutorialspoint.com/java/java_using_iterator.htm
Practice:

- Wildfire Model

Wildfire Evacuation Example

...Your turn, which buildings have to be evacuated? (use your BoundingBox class)
public class TestWildFireModel {
    
    /**
     * @param args
     */
    public static void main(String[] args) {
        // generate a wild fire model using the constructor in class WildFireModel (use WildfireModel(int numberOfBuildings)).
        
        // Extract the Wild Fire Polygons from the model (use getWildFirePolygon())
        
        // Extract the buildings from the model (use getBuildings())
        
        // Determine the bounding box of the Wild Fire Polygons (Use iterator to determine upper left and lower right points)
        // step 1: generate an iterator
        // step 2: set up a loop
        // step 3: in the loop, use next() to obtain each element and find the maximum x as lower right x, minimum x as upper left x,
        // maximum y as upper left y and minimum y as lower right y.
        
        // create the upper right and lower left points of the bounding box
        
        // create the bounding box
        
        // determine which building is within the bounding box (use iterator)
        // step 1: generate an iterator
        // step 2: set up a loop
        // step 3: in the loop, use next() to obtain each element and check whether this element is within the bounding box (use isInside() from BoundingBox)
    }
}