Craig Zilles   CS125 Lecture 19   "Encapsulation" 10/11/13

```
public class MedicalImage
{
  public Picture picture;
  public Date date;
  public Location where;
```

```
public class Link {
  public int value;
  public Link  next;
}
```

```
public class Simulation {
  public Atom[] atoms;
  public double temp;
```

```
public class Atom {
  public double x;
  public double y;
  public double vx;
  public double vy;
```

Create a new Atom and set its position to (5,8)

```
// Inside Universe.java
public static void main(String[] args) {




}
```

How would we compare two atoms, and you cannot edit Atom,java?
1) Create a class (static) method that takes two parameters 'a1, a2' of type Atom that returns true if two Atoms have exactly the same x and y. Hint static => no <u>this pointer!</u>

What if you can edit Atom.java? Hint - think of comparing two string objects,   s1.equals(s2)
Hint: You will need to use 'this.x'
2) Create an instance (non-static) method in Atom 'equals' that takes a pointer to an atom returns true if the atom has the same position as the given atom. Then (ii) write some code that uses your method h1.equals(h2)
(iii) Where would your code fail if the atom parameter value was null?

```
// Atom.java continued...
```

Create an instance method "moving" in Atom that takes no parameters and returns true iff the atom's vx or vy values are non-zero.

Create an instance method 'init' in Simulation that initializes the atom array with 100 slots and creates 100 atoms at random locations.

```
// Simulation.java continued
```

```
public class Xtring { // HOMEWORK (due in section)

  // Each Xtring needs to store its characters, so use a char array can call it 'array'-

    _______________________;

  public char charAt(int i) {

  }
  public int indexOf(char key) {
    for (int i = 0; i < array.length; i ++)
        if( ___________________________________
    return -1;
 }

  public boolean equals(Xtring s) {
//Hint: Do quick checks first, before comparing all characters







  }
  public Xtring toUpperCase() { // Character.toUpperCase(char) will be useful here
// Hint return a new String object. Don't change the original Xtring. The new object will need a new array.







  }
  public Xtring substring(int start, int end) { // start index is inclusive, end index is exclusive






  }
  public int indexOf(Xtring s) {
// Hint: Use nested loops (check for starting offset, inner loop checks all characters of s match this string)
```