120 Concepts and Common Programming Issues
Common Issues

- You can’t instantiate an interface
  - Interfaces describe methods
  - Classes implement those methods
- Recursive functions need base cases
  - If you get a stack overflow error, you probably forgot a base case
  - Recursion is powerful but requires a “leap of faith” in the recursive step
- Objects can be null, primitives can’t
  - Watch out for NPE’s
  - Make sure you are testing equality correctly
  - Careful with pointers!
- Pointers, primitives, and new
Static variables are shared between *all instances* of an object

- If one instance updates a static field, all instances feel the effect

Static methods are similar—this means they cannot reference/modify non-static fields or call non-static methods
Static vs. Dynamic

Static:
- Can be called without an instance of the class (e.g. Math.random(), Collections.sort())
- Cannot reference non-static fields or call non-static methods

Dynamic:
- Must have an instance of the class
- Can call any static or non-static methods
- Can modify the internal state of the calling object
Parameterization

Can parameterize classes to make them more generic

Rather than have separate classes for an IntList, a StringList, a DoubleList, etc., can create a generic class List<E>

E stands for an arbitrary type

- Good because it’s generic
- Bad because you can’t assume anything about it (except that it is an Object)
- A specific type will be provided upon instantiation (e.g. List<Integer> l = new LinkedList<Integer>();)
- In this case, everywhere you see an “E”, replace with “Integer” to understand the behavior