Discussion Questions for "Object Life Cycle"

1. Describe some cases where you want a creator to initialize a field to something other than the value of a parameter. Should this ever be something other that the default zero or null value?
2. Given the following DllNode class:

```
package dll;
class DllNode {
        /* Fields: */    DllNode prev; int value; DllNode next;
        /* Creator */    public DllNode(int value) { this.value=value; }
        public void addTail(int value) {
                if (next == null) {
                        next = new DllNode(value);
                        next.prev=this;
                } else next.addTail(value);
        }
        public String toString() {
                return "[" + ((prev == null) ? "null" : prev.value) +
                        "<-," + value + ",->" +
                        ((next == null) ? "null]" : next.value + "] " + next.toString());
        }
        public static void main(String args[]) {
                DllNode root = new DllNode(Integer.parseInt(args[0]));
                for(int i=1; i<args.length; i++) {
                        root.addTail(Integer.parseInt(args[i]));
                }
                System.out.println("root is: " + root);
                root = null;
        }
}
```

If you compile and invoke this as: java -cp ../../classes dll.DllNode 4 6 9

a. Just before the last "root = null;" line in main, describe all objects and references to those objects. Do all the objects have references? Are all those objects reachable? Are any objects available for garbage collection?
b. Just after the last "root = null;" line in main, what has changed from a.? Do all objects still have references? Are all objects still reachable? Are any objects available for garbage collection?
c. How many "String" objects does the program invocation create? How many of those objects are available for garbage collection?

3. The process of “Garbage Collection” is somewhat controversial in the Java world. There are a branch of Java coders working on real-time embedded systems that have developed an alternative version of Java that does not do garbage collection. Other object oriented languages, such as C++, do not do garbage collection at all, relying on the programmer to delete all the objects they create. What do you think about garbage collection? Would you rather Java did not do automatic garbage collection and ran your program faster?