1. (Multithreading, 10pts)
List all possible outputs from the following program. Indicate next to each possible output whether all threads complete at the end, or, if they do not, which threads remain blocked.

```java
l = new ReentrantLock();
c = l.newCondition()
```

**Thread 1**
l.lock();
System.out.print("a");
c.signalAll();
System.out.print("b");
c.await();
System.out.print("c");
l.unlock();

**Thread 2**
l.lock();
System.out.print("d");
c.await();
System.out.print("e");
c.signalAll();
System.out.print("f");
l.unlock();

**abd**—both threads 1 and 2 are locked
**dabefc**—both threads 1 and 2 complete
2. (Garbage Collection, 6pts)
Consider the following Java code:

```java
class A {
    static Integer a, b, c, d;
    private void foo() {
        a = new Integer(1); // object 1
        b = new Integer(2); // object 2
        c = new Integer(3); // object 3
        c = a;
    }
}
```

What objects are garbage after foo returns? Explain your answer.

**Object 3 is garbage because it is no longer reachable**

3. (Function arguments, 4pts)
What are upward and downward funargs? What is the main difference in their implementation?

**Upward funargs are when a function is returned from another function call and downwards funargs occur when a function is passed as parameter into another function. Upwards funargs require closures and are stored on the heap; downwards funargs can be accessed on the stack.**