Drew Beck
Homework 5

Object Orientation

1. c1 = i2, Java disallows this assignment because, though c2 extends c1, only c2 implements i2. There is no connection between i2 and c1.
   i1 = c1, Java allows this assignment because c1 is a class which implements i1.
   c1 = c2, Java allows this assignment with a downcast because c2 is a subclass of c1.
   c3 = i1, Java disallows this assignment because c1 only implements I1.
   c2 = c3, Java disallows this assignment because c2 and c3 are not related in any way.

2. Covariance is used in the Java language when overriding methods. The overriding method is covariant in the return type only. Everything else about the method must be the same including argument types. Therefore, if the above example was written in Java, A1 would have to be a subclass of A2, but B1 and B2 would need to be the same type. The advantage of this rule is it ensures no type errors can occur when using the overriding method. Source: https://briangordon.github.io/2014/09/covariance-and-contravariance.html

   Contravariance is used in C#. This is essentially the opposite of covariance, which means in the example, A1 could be a more generic type of A2, i.e. A2 inherits from A1. Source: https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/covariance-contravariance/

Parameters

1. By Value: A[0] = 0, A[1] = 2
   By Reference: A[0] = 3, A[1] = 2
   By Value-Result: A[0] = 1 or A[0] = 3 depending on the order in which the result assignment is performed, A[1] = 2
   By Macro Expansion: A[0] = 1, A[1] = 3
   By Name: A[0] = 1, A[1] = 3