Student Name: ______________________ Student ID # ______________________

UOSA Statement of Academic Integrity

On my honor I affirm that I have neither given nor received inappropriate aid in the completion of this exercise.

Signature: ______________________ Date: ______________________

Question 1: Inheritance (20 points)

You are asked to write software that will deal with two types of objects, A and B. Objects of types A and B are similar in that they contain data of exactly the same type and in the same quantities. However, they are different in that the actions that these objects can perform is not entirely the same. Of the 20 methods you would need for objects of type A, only 18 would be applicable to objects of type B. Of the 24 methods you would need for type B, only 18 are applicable to objects of type A.

A. Would it be better to create a single class for objects of these two types, two different (unrelated) classes, a superclass and a subclass, or some other class relationship? *Explain your answer thoroughly, including both* ***how*** *you would set up the class relationship and* ***why*** *you would set it up this way.*

B. What additional information would you find helpful in making your decision for part A? *Why?*

Question 2: Abstract Classes & Interfaces (20 points)

A. When does it make sense to create an abstract class, rather than an interface?

B. When does it make sense to create an interface, rather than an abstract class?

Question 3: Object-Oriented Design (20 points)

Describe the classes you would create and how they are related to one another for the following simple problem:

> You are to create a program that handles records about people at a university. The types of people that your program will deal with are undergraduate students, graduate students, faculty, staff, and graduate assistants (who are graduate students employed to help with teaching and/or research).

You may draw this with a simplified UML class diagram (showing just the class names and the relations between them) or write this out in words.

Question 4: Generics (20 points)

The **Collection** interface is implemented using generics yet the **Collections** (note the ‘s’) class, which is used to manipulate objects of subclasses of **Collection** (no ‘s’), does not use generics.

A. *Why does* the **Collection** interface use generics? (I.e., what is a good reason for designing the interface this way?)

B. *Why doesn’t* the **Collections** class use generics? (Hint: It can’t. So this question is really *why can’t* it?)

Question 5: Java Collections Framework (20 points)

A. Why are the map classes (such as **HashMap**) not derived from the **Collection** interface?

B. If you were to cast a collection object of type **ArrayList** containing a list giving the birth year for each of your classmates to an object of type **HashSet**, what would you expect to change about the contents of the collection object?