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: _____

Notes Regarding this Examination

Open Book(s) You may consult any printed textbooks in your immediate possession during the course of this examination.

Open Notes You may consult any printed notes in your immediate possession during the course of this examination.

No Electronic Devices Permitted You may not use any electronic devices during the course of this examination, including but not limited to calculators, computers, and cellular phones. All electronic devices in the student's possession must be turned off and placed out of sight (for example, in the student's own pocket or backpack) for the duration of the examination.

Violations Copying another's work, or possession of electronic computing or communication devices in the testing area, is cheating and grounds for penalties in accordance with school policies.

Question 1: Polymorphism (15 points)

List three types of polymorphism and briefly explain one benefit of each.

A. One type of polymorphism.

B. A second type of polymorphism.

C. A third type of polymorphism.

Question 2: Object-Oriented Design (10 points)

Javier is designing the Class structure for software that will maintain and use a database. Because he has been told that this software will need to print the database as a table of information (among other functions it will need to carry out), he decides to add a **PrintAsTable** Class.

A. What common design mistake has Javier just made? *Explain why* this is a mistake.

B. What would be a better way to add this functionality to the software? *Explain why* this is a better way to add this functionality.

Question 3: Object-Oriented Design Redux (20 points)

You are asked to develop a piece of bioinformatics software. (Note that "bio" means life and "informatics" is the science of information. So, you are being asked to write software that contains information about living things.) This software will keep track of the chromosomes and other characteristics of individual elephants, zebras, ostriches, finches, baobab trees, and papyrus plants.

A. Draw a simplified UML diagram for this software that shows that elephants and zebras are types of mammals, ostriches and finches are types of birds, and baobab trees and papyrus plants are types of plants. Your diagram should further show that mammals and birds are types of animals, and that animals and plants are types of organisms. (Note that you don't need to show variables or methods in this diagram.)

B. Add to your UML diagram the fact that each organism has one or more chromosomes and that those chromosomes are ordered sequences of genes. For this software, you may use **Strings** for genes. (Note that you will need to add some variables to the diagram.)

Question 4: Abstract Classes & Interfaces (15 points)

Consider your design from Question 2. If, to answer any of these questions you would need additional information, please say what that information is.

A. Would it be appropriate to make any of the "types" (elephants, mammals, animals, organisms, etc.) interfaces? If so, which ones and *why?* If not, *why not?*

B. Would it be appropriate to make any of the "types" (elephants, mammals, animals, organisms, etc.) abstract classes? If so, which ones and *why?* If not, *why not?*

C. Would it be appropriate to make any of the "types" (elephants, mammals, animals, organisms, etc.) concrete classes? If so, which ones and *why?* If not, *why not?*

Question 5: Generics (20 points)

A. What is the primary advantage of using generics for variables? *Explain why* this is an advantage.

B. What is the primary advantage of using generics for methods? *Explain why* this is an advantage.

Question 6: Java Collections Framework & Inheritance (20 points)

A. What is common about all the method signatures added by the **List** interface that were not in the **Collection** interface? *Why* doesn't the **Collection** interface have these method signatures?

B. Since only method signatures (not method bodies) are inherited by **List** from **Collection**, give one good reason *why* it is valuable (worthwhile) to have this inheritance at all.