Homework 1 -- Birds

Due Sep 24 by 11:59pm **Points** 100 **Available** after Sep 11 at 12am

Modern biologists have been classifying life into categories according to the complexity of that life as well as a distinguishable property that can be observed since the 1700s. Birds are defined as warm-blooded, bipedal, vertebrate animals who have two wings instead of arms. They are typically covered in feathers and have a beak instead of a mouth. Bird is one of the largest classes of animals in the bigger Kingdom *Animalia.*

Some of the classifications of birds (derived in part from **BioExplorer.net (https://www.bioexplorer.net/animals/birds/)**) include:

- Birds of prey all have sharp, hooked beaks with visible nostrils. They include hawks, eagles, and osprey.
- Flightless birds live on the ground and have no (or undeveloped) wings. They include the emus, kiwis, and moas. Some (but not all) of these birds are extinct.
- Owls are distinguished by the facial disks that frame the eyes and bill.
- Parrots have a short, curved beak and are known for their intelligence and ability to mimic sounds. Many pet parrots can learn a vocabulary of up to 100 words and often adopt a single "favorite" saying. They include the rose-ring parakeet, gray parrot, and sulfur-crested cockatoo.
- Pigeons (or doves) are known for feeding their young "bird milk" very similar to the milk of mammals. Found all over the world, there are several varieties that are extinct.
- Shorebirds include the great auk, horned puffin, and African Jacana. They live near water sources including wetlands, freshwater and saltwater shorelands, even the ocean.
- Waterfowl are another classification that live near water sources (fresh or salt) and include ducks, swans, and geese.

In this homework you will be asked to design and implement a solution that can be used to keep track of birds according to the above classifications (we are omitting many classifications to keep this assignment relatively confined). For each classification of bird, your solution should be able to track each of the following:

- The type of bird (e.g., duck, horned puffin, etc), their defining characteristic, whether they are extinct, as well as the number of wings they have.
- A description of what 2-4 items they prefer to eat from the following list: berries, seeds, fruit, insects, other birds, eggs, small mammals, fish, buds, larvae, aquatic invertebrates, nuts, and vegetation.

- For birds that live near water, the name of the body of water that they live by.
- In the case of parrots, the number of words in their vocabulary as well as their single "favorite" saying.

Part 1 – Design

Before you start to write code, it is a good idea to design your solution. To do this, you need to understand what your program needs to do, decide what classes you will need, and what methods each class will need. It is a really good idea to think about how each of theses methods and classes could be tested to ensure the correctness of your implementation. Thinking about this early will make the coding process much easier. To help you with your design process, you are required to meet with your professor during the design meeting period listed on Canvas. You must bring your completed design and test plan to this meeting and be ready to explain why your design is the right way to solve the problem. Each design meeting will last approximately 10 minutes.

What to do

Design the data for the above in a way that captures their similarities and accurately represents the relevant data. Create interfaces/classes as you see fit and specify appropriate constructors that allows one to create a question as specified above.

Write a testing plan that thoroughly tests your design. How do your tests ***convince someone else that your code works correctly***? For each test in your design, you should specify what condition you are testing, what example data you will use to test that and what values you might expect a method to produce (known as the *expected value*) when appropriate.

What to submit

Log into the **Handins submission server** **(https://handins.ccs.neu.edu)** and upload a single PDF file to the *Homework 1 Design* assignment. Your PDF file should should include:

- A UML class diagram containing
  - The classes you will need
  - The relationships between the classes
  - What methods and variables those classes have
  - The visibility of those methods and variables
- A testing plan including for each test case
  - Which condition you are testing
  - What example(s) you would use to test that

You do not need any of your implementation or code.

Criteria for grading

Your design will be reviewed during your design meeting. During your meeting, you will be asked to walk us through your design. You should be prepared to discuss:

- How will your design encapsulate the different types of birds?
- What fields you expect to have in each class? What is their access and why?
- What is your strategy for testing your design? How will this convince someone else that your code works correctly?

Part 2 - Development

What to do

Implement the class hierarchy that you specified in Part 1. Create a class that represents a conservatory that houses many different types of birds. The conservatory is broken into various aviaries. This new class should:

- Allow you to rescue new birds and bring them into your conservatory.
- Calculate what food needs to be kept and in what quantities.
- Assign a bird to a given aviary in the conservatory. Assignments must follow the following criteria:
  - There is a maximum of 20 aviaries in the conservatory.
  - Any bird can be inserted into an empty aviary.
  - No aviary can house more than 5 birds.
  - No extinct birds can be added to an aviary.
  - Flightless birds, birds of prey, and waterfowl should not be mixed with other bird types.
- Allow a guest to look up which aviary a bird is in.
- Produce text for any given aviary that gives a description of the birds it houses and any interesting information that it may have about that animal.
- Produce a “directory” that lists all the aviaries by location and the birds they house.
- Produce an index that lists all birds in the conservatory in alphabetical order and their location.

Create a driver class with a main method that creates a conservatory, adds some birds to it and prints the resulting directory and index. This method should serve as a lightweight demo; it should **not** be a comprehensive test of your conservatory.

Documentation

We expect your code to be well-commented using well-formed English sentences. The expectations are:

- Each interface and class contains a comment above it explaining specifically what it represents. This should be in plain language, understandable by anybody wishing to use it. Comment above a class should be specific: it should not merely state that it is an implementation of a particular interface.
- Each public method should have information about what this method accomplishes (purpose), the nature and explanation of any arguments, return values and exceptions thrown by it and whether it changes the calling object in any way (contract).
- If a class implements a method declared in an interface that it implements, **and** the comments in the interface describe this implementation completely and accurately, do not replicate that documentation in the class.
- All comments should be in Javadoc-style.

Create a JAR file of your program

In order to make your application easier to run, you are required to create and submit a `JAR` file:

- Directions for doing this in IntelliJ can be found at **this link (https://www.jetbrains.com/help/idea/packaging-a-module-into-a-jar-file.html)** .
- Directions for doing this in Eclipse can be found at **this link (https://www.codejava.net/ides/eclipse/how-to-create-jar-file-in-eclipse)** .

What to submit

Your `zip` file should contain three folders: src, test and res (even if empty).

- All your code should be in `src/`.
- All your tests should be `test/`.
- Your original and revised design document should be in `res/`.
- Submit a correct `JAR` file in the `res/` folder. We should be able to run your program from this jar file.
- Submit at least two example runs of your program the res/ folder that communicate to us how to specify commands to run your program to verify that it meets all of the above specifications.
- Submit a `README.md` file that documents how to use your program, which parts of the program are complete, and design changes and justifications. The file should also include any necessary citations (see syllabus).

Criteria for grading

You will be graded on:

- Whether your code is well-structured and clean (i.e. not unnecessarily complicating things or using unwieldy logic).
- Correctness of your implementations including whether your example runs contain enough information for us to run your program from the provided `JAR` file and to see that all of the required features are working correctly.
- Whether you have written enough comments for your classes and methods, and whether they are in proper `Javadoc` style.
- Whether your code is formatted correctly (according to the style grader).
- Self-assessment to be completed on the Handins submission server.