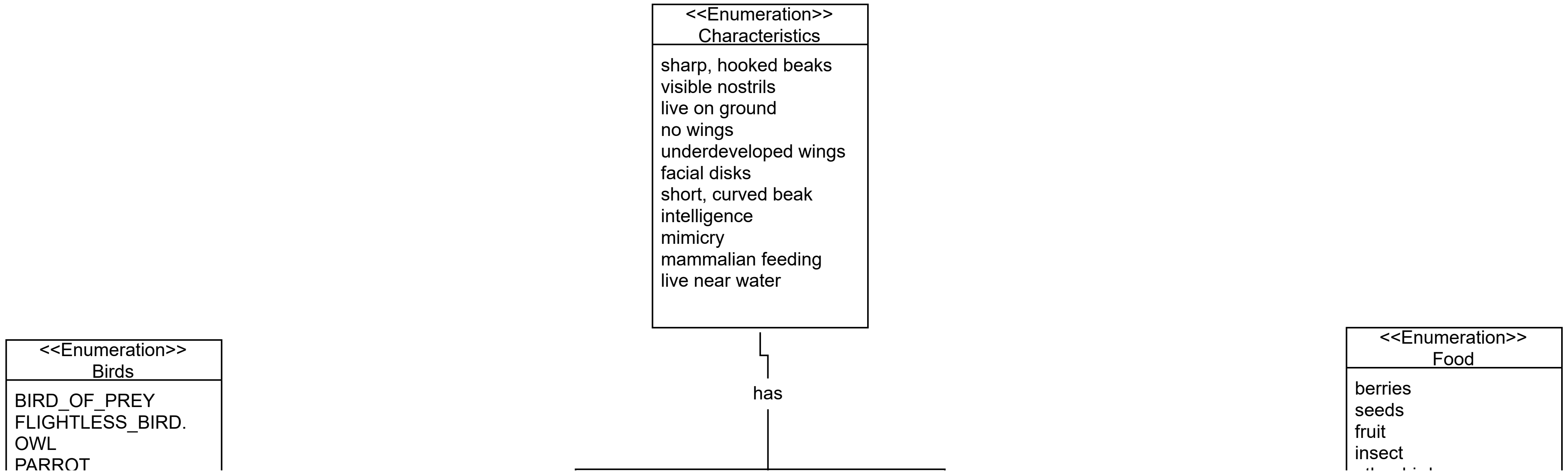
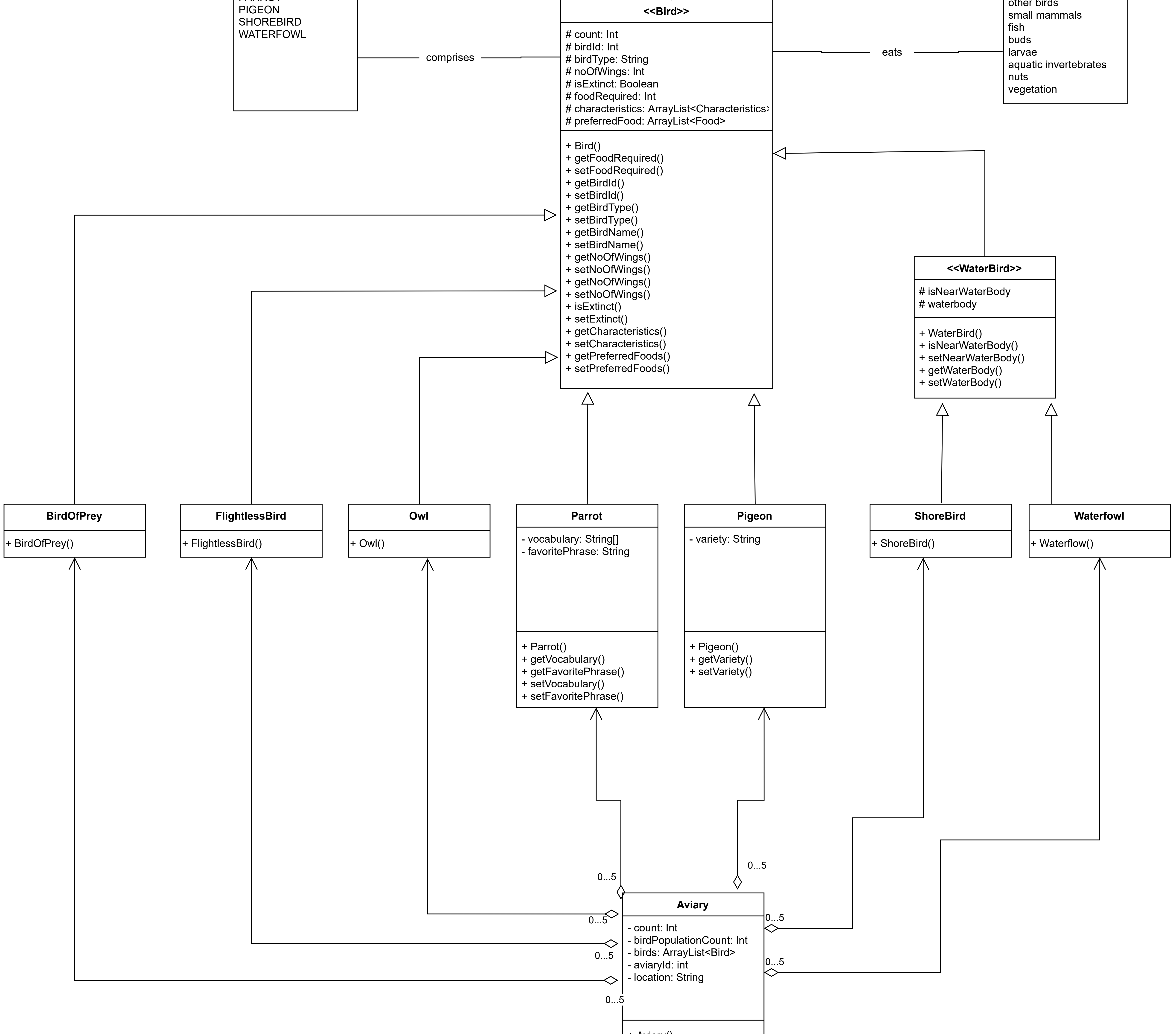
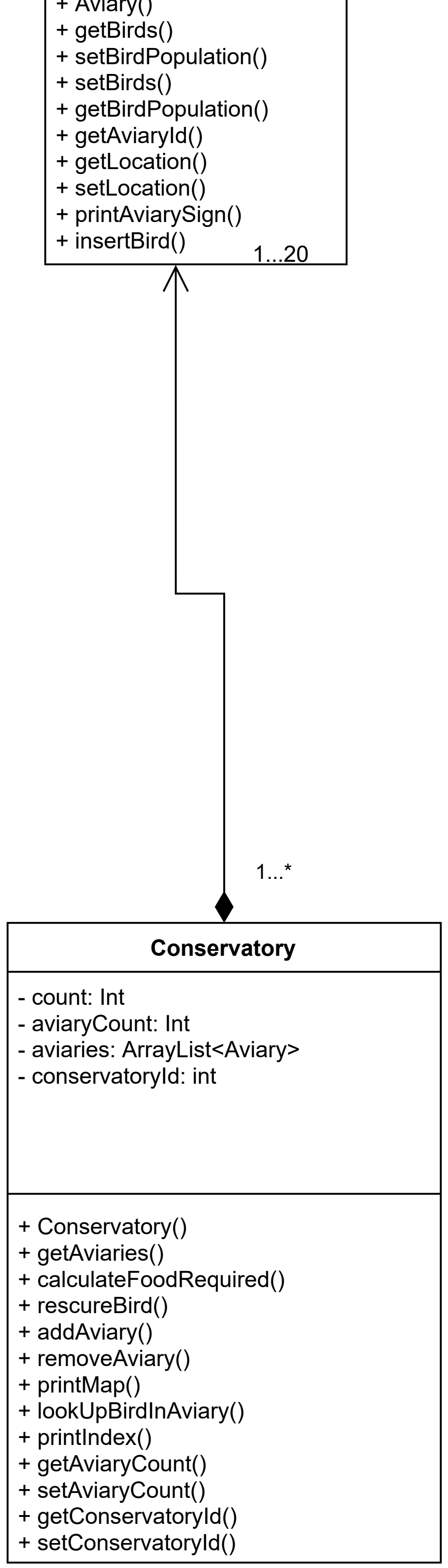


<<Enumeration>>
Characteristics
sharp, hooked beaks
visible nostrils
live on ground
no wings
underdeveloped wings
facial disks
short, curved beak
intelligence
mimicry
mammalian feeding
live near water
has
<<Enumeration>>
Birds
BIRD_OF_PREY
FLIGHTLESS_BIRD.
OWL
PARROT
<<Enumeration>>
Food
berries
seeds
fruit
insect


PARROT
PIGEON
SHOREBIRD
WATERFOWL
comprises
<<Bird>>
# count: Int
# birdId: Int
# birdType: String
# noOfWings: Int
# isExtinct: Boolean
# foodRequired: Int
# characteristics: ArrayList<Characteristics>
# preferredFood: ArrayList<Food>
+ Bird()
+ getFoodRequired()
+ setFoodRequired()
+ getBirdId()
+ setBirdId()
+ getBirdType()
+ setBirdType()
+ getBirdName()
+ setBirdName()
+ getNoOfWings()
+ setNoOfWings()
+ getNoOfWings()
+ setNoOfWings()
+ isExtinct()
+ setExtinct()
+ getCharacteristics()
+ setCharacteristics()
+ getPreferredFoods()
+ setPreferredFoods()
eats
other birds
small mammals
fish
buds
larvae
aquatic invertebrates
nuts
vegetation
<<WaterBird>>
# isNearWaterBody
# waterbody
+ WaterBird()
+ isNearWaterBody()
+ setNearWaterBody()
+ getWaterBody()
+ setWaterBody()
BirdOfPrey
+ BirdOfPrey()
FlightlessBird
+ FlightlessBird()
Owl
+ Owl()
Parrot
- vocabulary: String[]
- favoritePhrase: String
+ Parrot()
+ getVocabulary()
+ getFavoritePhrase()
+ setVocabulary()
+ setFavoritePhrase()
Pigeon
- variety: String
+ Pigeon()
+ getVariety()
+ setVariety()
ShoreBird
+ ShoreBird()
Waterfowl
+ Waterflow()
0...5
0...5
0...5
0...5
0...5
0...5
0...5
Aviary
- count: Int
- birdPopulationCount: Int
- birds: ArrayList<Bird>
- aviaryId: int
- location: String
+ Aviary()


+ Aviary()
+ getBirds()
+ setBirdPopulation()
+ setBirds()
+ getBirdPopulation()
+ getAviaryId()
+ getLocation()
+ setLocation()
+ printAviarySign()
+ insertBird()
1...20
1...*
Conservatory
- count: Int
- aviaryCount: Int
- aviaries: ArrayList<Aviary>
- conservatoryId: int
+ Conservatory()
+ getAviaries()
+ calculateFoodRequired()
+ rescureBird()
+ addAviary()
+ removeAviary()
+ printMap()
+ lookUpBirdInAviary()
+ printIndex()
+ getAviaryCount()
+ setAviaryCount()
+ getConservatoryId()
+ setConservatoryId()

Testing Plan

1. Create Instance of each classification class and check if they belong to the same parent class, i.e., the Bird class.

2. Assign random words to a Parrot object's vocabulary array and check if the contents match with the initial array.

3. Assign a favoriteWord to Parrot object and check if favoriteWord and supplied word match.

4. Assign favoriteWord and vocabulary array and check if favoriteWord exists in vocabulary.

5. Assign variety to Pigeon  and assert variety.

6. For Shorebird and Waterfowl, check if isNearWaterbody evaluates to true.

7. Assert nearby water bodies for Shorebird and Waterfowl objects

8. If a Bird instance has isExtinct == true,  rescueBird() and insertBird() should throw errors.

9. preferredFoods array content of each Bird object should be part of the Foods Enum.

10. Check if more than 5 birds are being added to an aviary.

11. Check if more than 20 aviaries are being added to a conservatory.

12. Check if certain types of birds (Flightless birds, birds of prey, and waterfowl are being added together in an aviary.

13. Check if the aviary sign printed is valid.

14. Check if the map printed is valid.
15. Check if the index printed is valid.
16. Check if bird lookup in the aviary returns the correct aviary.
17. Check if the food calculated is what is needed by the birds in a conservatory.
18. Check if all aviaries in the conservatory are full.