Kiruthiga Muralidharan

Abstract

UML Class diagram and testing plan for Part - 1

Lab 1 - Birds

CS5010 – Programming Design Paradigm

**UML Class Diagram:**

**A diagram of a type of text

Description automatically generated with medium confidence**

**Test Plan:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Method** | **Description** | **Input Assignment** | **Expected Output** |
| testRescueBird () | Tests if a bird is rescued and added to the library | Parrot("Parrot", "Has a cute voice", false, 2, List.of(PreferredFood.SEEDS, PreferredFood.FRUITS), 20, "Hey beautiful!"); | Parrot |
| testAddBirdSuccess () | Tests if the bird is added. (Aviary is within the capacity and there is no problem of coexistence) | Parrot("Parrot", "Has a cute voice", false, 2, List.of(PreferredFood.SEEDS, PreferredFood.FRUITS), 20, "Hey beautiful!"); Owl("Owl", "Has big eyes", false, 2, List.of(PreferredFood.SEEDS, PreferredFood.BUDS), "Round"); | 2 (No. of birds added) |
| testAviaryFull () | Tests if the Aviary adds 6th bird | Parrot1, Parrot2, Parrot3, Parrot4, Parrot5, Parrot6  Aviary1 | IllegalStateException |
| testCalculateFoodRequirements () | Tests the food requirements for the conservatory | Parrot("Parrot", "Has a cute voice", false, 2, List.*of*(PreferredFood.*SEEDS*, PreferredFood.*FRUITS*), 20, "Hey beautiful!");  Owl("Owl", "Has big eyes", false, 2, List.*of*(PreferredFood.*FISH*, PreferredFood.*OTHER\_BIRDS*), "Round"); | Key: SEEDS, Value:1  Key: FRUITS, Value:1  Key: FISH, Value:1  Key: OTHER\_BIRDS, Value: 1 |
| testPreferredFoodListSize() | Tests if the food list size is between 2 and 5 | Parrot("Parrot", "Has a cute voice", false, 2, List.*of*(PreferredFood.*SEEDS*), 20, "Hey beautiful!"));  false, 2, [“Berries”, “Seeds”], [“Freshwater”]); | IllegalArgumentException |
| testPrintMap() | Tests the map containing aviaries and the birds present in it. | Parrot("Parrot", "Has a cute voice", false, 2, List.*of*(PreferredFood.*SEEDS*, PreferredFood.*FRUITS*), 20, "Hey beautiful!");  Owl("Owl", "Has big eyes", false, 2, List.*of*(PreferredFood.*FISH*, PreferredFood.*OTHER\_BIRDS*), "Round"); | Aviary 1 – Parrot, Owl |
| testBirdIndexPrint() | Tests the birds conservatory index | Parrot("Parrot", "Has a cute voice", false, 2, List.*of*(PreferredFood.*SEEDS*, PreferredFood.*FRUITS*), 20, "Hey beautiful!");  Owl("Owl", "Has big eyes", false, 2, List.*of*(PreferredFood.*FISH*, PreferredFood.*OTHER\_BIRDS*), "Round"); | Parrot located in Aviary1  Owl located in Aviary 1 |
| testAviaryLimit() | Tests the limit of aviaries (20) | List.of(aviary1…aviary21) | IllegalArgumentException |
| testPrintSign() | Tests the aviary sign | Parrot("Parrot", "Has a cute voice", false, 2, List.*of*(PreferredFood.*SEEDS*, PreferredFood.*FRUITS*), 20, "Hey beautiful!"); | Aviary 1 Aviary of Birds!!!  Parrot with its unique characteristic Has a cute voice!! |
| testCoexistence() | Tests if the input birds can coexist together | BirdOfPrey("Prey1", "Predator", false, 2, List.*of*(PreferredFood.*SEEDS*, PreferredFood.*FRUITS*));  FlightlessBird("Flightless1", "Flightless", false, 2, List.*of*(PreferredFood.*SEEDS*, PreferredFood.*FRUITS*));  Only 1 aviary | IllegalArgumentException |