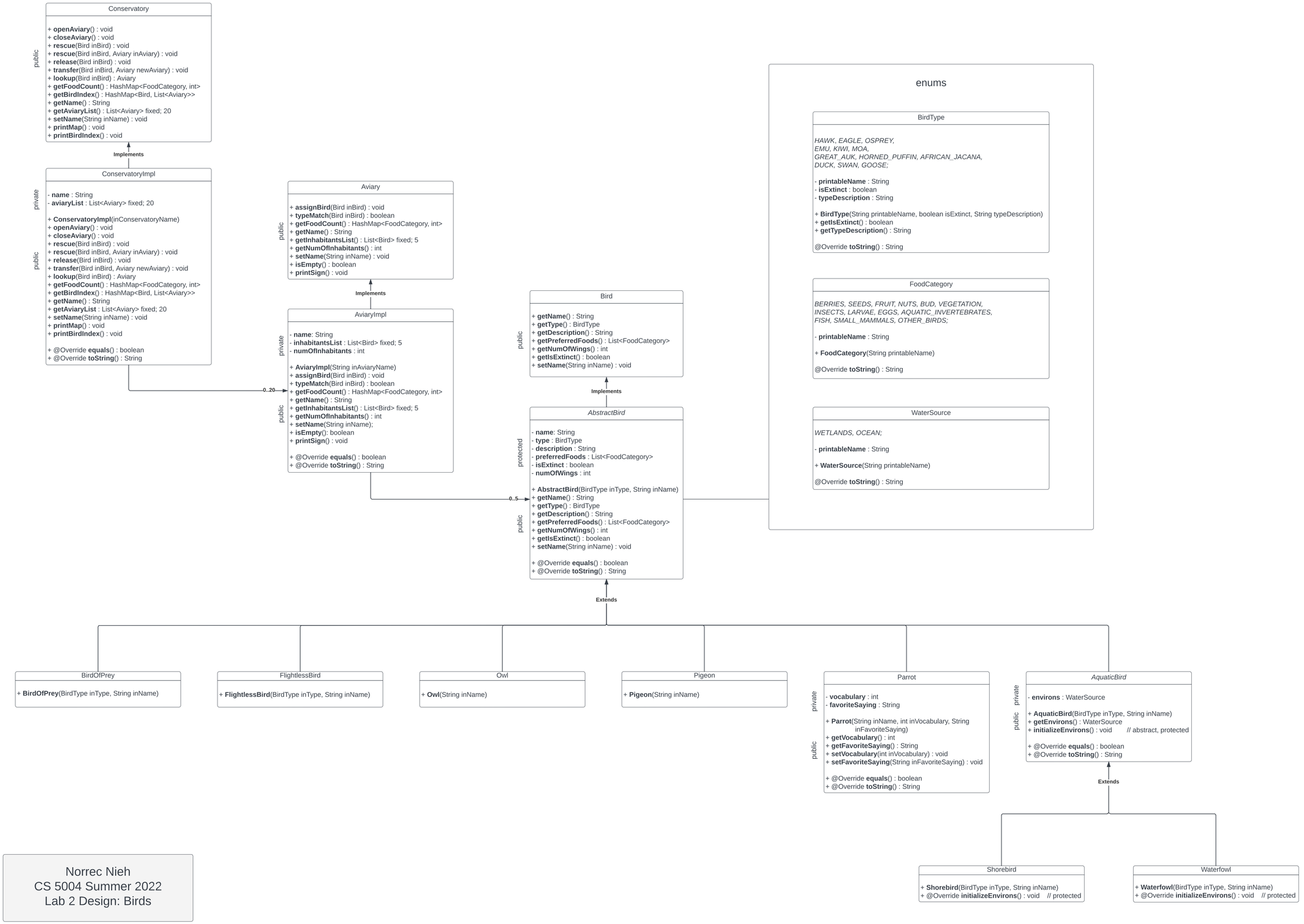

Norrec Nieh
CS 5004 Summer 2022
Lab 2 Design: Birds

Norrec Nieh
CS 5004 Summer 2022
Lab 2: Birds – Testing Plan

AbstractBirdTest

1) Test constructors for each subcategory of Bird:
    a. testBirdConstructor
        i. BirdOfPrey
            1. Creating a HAWK
            2. Creating an EAGLE
            3. Creating an OSPREY
        ii. FlightlessBird
            1. Creating an EMU
            2. Creating a KIWI
            3. Creating a MOA
        iii. Owl
        iv. Pigeon
        v. Parrot
        vi. Shorebird
            1. Creating a GREAT_AUK
            2. Creating a HORNED_PUFFIN
            3. Creating an AFRICAN_JACANA
        vii. Waterfowl
            1. Creating a DUCK
            2. Creating a SWAN
            3. Creating a GOOSE
2) Test constructor exceptions:
    a. testBirdOfPreyConstructorException
        i. Throw exception if not HAWK, EAGLE, or OSPREY
    b. testFlightlessBirdConstructorException
        i. Throw exception if not EMU, KIWI, or MOA
    c. testParrotConstructiorException
        i. Throw exception if vocabulary is a negative number
    d. testShorebirdConstructorException
        i. Throw exception if not GREAT_AUK, HORNED_PUFFIN, AFRICAN_JACANA
    e. testWaterfowlConstructorException
        i. Throw exception if not DUCK, SWAN, or GOOSE
3) Test Parrot-specific methods on a Parrot obj
    a. testGetVocabulary
    b. testGetFavoriteSaying
    c. testSetVocabulary
    d. testSetFavoriteSaying
4) Test AquaticBird methods
    a. testGetEnvirons
        i. 1. Call initializeEnvirons and getEnvirons on a Shorebird obj
        ii. 2. Call initializeEnvirons and getEnvirons on a Waterfowl obj

(cont'd)

(cont'd)

5) Test AbstractBird getters and setter for one general Bird case
    a. testGetName
    b. testGetType
    c. testGetDescription
    d. testGetPreferredFoods
    e. testGetNumOfWings
    f. testGetIsExtinct
        i. Assert true if extinct
        ii. Assert false if not extinct
    g. testGetSetName
6) Test equals() method for a few general Bird cases
    a. testEquals
        i. Assert true for each pair of like birds
        ii. Assert false for pair of unlike birds
        iii. Assert false for bird and aviary obj
7) Test toString() method for each type of Bird
    a. testToString
        i. Print sign description for each type of Bird; each type should differ

AviaryImplTest

1) testConstructor
2) testAssignBird
    a. Check that a general case Bird can be placed in an Aviary
    b. Check that a BirdOfPrey cannot be placed in an Aviary with another Bird category
    c. Check that a BirdOfPrey *can* be placed in an Aviary with other BirdOfPreys
    d. Check that a FlightlessBird cannot be placed in an Aviary with another Bird category
    e. Check that a FlightlessBird *can* be placed in an Aviary with other FlightlessBirds
    f. Check that a Waterfowl cannot be placed in an Aviary with another Bird category
    g. Check that a Waterfowl *can* be placed in an Aviary with other Waterfowl
3) testAssignBirdException
    a. Check that assigning a Bird to a full Aviary raises an exception
4) testTypeMatch
    a. Check that any Bird matches type with an empty Aviary
    b. Check that a BirdOfPrey does not match type with an Aviary with another Bird category
    c. Check that a BirdOfPrey matches type with an Aviary with only BirdOfPreys
    d. Check that a FlightlessBird does not match type with an Aviary with another Bird category
    e. Check that a FlightlessBird matches type with an Aviary with only FlightlessBirds
    f. Check that a Waterfowl does not match type with an Aviary with another Bird category
    g. Check that a Waterfowl matches type with an Aviary with only Waterfowl
5) testGetFoodCount
    a. Check that function correctly returns a HashMap of FoodCategory keys and int values
6) testGetName
7) testGetInhabitantsList
8) testGetInhabitantsListException
    a. Throw exception if numOfInhabitants == 0
9) testGetNumOfInhabitants
10) testIsEmpty
    a. Assert true if empty
    b. Assert false if not empty
11) testSetName
12) testPrintSign
    a. Print an Aviary with a single Bird
    b. Print an Aviary with multiple Birds
    c. Print an Aviary with multiple Birds that are the same type
13) testPrintSignException
    a. Check that attempting to print an empty Aviary raises an exception
14) testEquals
    a. Check that the two Aviaries have the same contents
15) testToString
    a. Print cursory details of the Aviary

ConservatoryImplTest

1) testConstructor
2) testOpenAviary
3) testOpenAviaryException
    a. Throw exception if there are already 20 aviaries
4) testCloseAviary
5) testCloseAviaryException
    a. Throw exception if there are no open aviaries
6) testRescue
    a. Check that a Bird can be rescued and assigned to an Aviary in the general case
    b. Check that a Bird can be rescued and assigned to a specific Aviary
7) testRescueException
    a. Throw exception if there are no available Aviaries for an incoming Bird
    b. Throw exception if attempting to rescue an extinct Bird
    c. Throw exception if assigning Bird to a specific full Aviary
8) testRelease
9) testTransfer
10) testTransferException
    a. Throw exception if typeMatch == false;
    b. Throw exception if Aviary is full
11) testLookup
    a. Check that correct Aviary is returned
12) testGetFoodCount
    a. Check that a correct HashMap is returned for a populated Conservatory
    b. Check that a correct HashMap is returned for an empty Conservatory
13) testGetBirdIndex
    a. Create a Conservatory and check that a correct HashMap is returned
14) testGetName
15) testGetAviaryList
16) testGetAviaryListException
    a. Throw exception if there are no Aviaries open
17) testSetName
18) testPrintMap
19) testPrintBirdIndex
20) testEquals
21) testToString
    a. Print cursory details of Conservatory