Testing for Project 1

Sanctuary s; Sanctuary otherS; Primate p; Housing h;

testReceivePrimate - s.receivePrimate(p)
expected: adding Primate p to isolation cage.
expected: OutOfBoundException() if isolation cage is full.

testSpeciesList - s.getSpeciesList()
expected: list of species in isolation and in enclosures.

testPrimateList - s.getPrimatesList()
expected: a hash-map of the format: (species_name: housing_type)

testPrimatesInEnclosure - getPrimatesInEnclosure(e)
expected: a sign (hash-map) of the format (name: [sex, favFood])

testShoppingList - s.getsShoppingList()
Expected: a hash-map of format (name: fav_food)

testFindHousingOf - findHousingOf("species_name")
Expected: name of housing
Expected: "Species is currently not in Sanctuary" if species does not exist.

testExchangePrimate -s.ExchangePrimate(id, otherS)
expected: removing p from s and adding p to otherS isolation.

testIncreaseCageCapacity - increaseCageCapacity(c)
expected: Increase the Isolation capacity.

testIncreaseCageCapacity - increaseTroopCapacity(t)
expected: Increase the Enclosure capacity.

testMoveToIsolation - s.moveToIsolation(p)
expected: primate moves to Isolation
Expected: IllegalOperation if primate is not in isolation

testMoveToEnclosure - s.moveToEnclosure(p)
expected: primate moves to enclosure
Expected: IllegalOperation if primate is not in enclosure
testGetters
expected: correct details of primate.

<<interface>>
Sanctuary

+ provideHome(name: String, age: int, sex: char, species: String, size: int, weight: int, favFood: String)
+ getFreeCages()
+ getNumOfCages()
+ getNumOfTroops()
+ increaseCagesBy(n: int)
+ increaseTroopsBy(m: int, area: int)
+ increaseEnclosureAreaBy(String id, int area)
+ moveToIsolation(String name)
+ moveToEnclosure(String name)
+ getSpeciesList(): TreeMap<String, ArrayList<String>>
+ getEnclosureSign(id: String): ArrayList<String[]>
+ getHousing(species: String): ArrayList<String>
+ getShoppingList(): TreeMap<String, Integer>
+ getPrimateList(): TreeMap<String, String>
+ moveToAnotherSanctuary(name: String, sanctuary: Sanctuary)

PrimateSanctuary

- n: int
- m: int
- freeCages: int
- isolation: ArrayList<Cage>
- enclosures: ArrayList<Enclosure>
- animalNames: ArrayList<Enclosure>

- PrimateSanctuary(n: int, m: int, area: int)
- gatherPrimateDetails(): ArrayList<Information>
+newPrimateSanctuary(n: int, m: int, area: int)

Cage

-primate: Primate
-id: String
-empty: boolean

~removePrimate()
~ getId(): String
~ isEmpty(): boolean
~ containsPrimate(name: String): boolean
~ putPrimate(Primate primate)
~ getPrimateDetails(): Information
~ increaseAreaBy(area: int)
~ popPrimate(): Primate

Enclosure

-id: String
-totalArea: float
-occupiedArea: String
-species: String
-troop: ArrayList<Primate>

~Enclosure(area: int)
-searchTroopFor(name: String): Primate
~ getId(): String
~ getTroopSpecies(): String
~ getFreeArea(): int
~ containsPrimate(name: String): boolean
~ putPrimate(Primate primate)
~ getPrimateDetails(): ArrayList<Information>
~ increaseAreaBy(area: int)
~ popPrimate(name: String): Primate

Primate

-name: String
-sex: char
-size: int
-species: String
-weight: int
-age: int
-favFood: Food

~getSpaceRequirement(size: int): int
~getFoodRequirement(size: int): int
~getName(): String
~getSex(): char
~getSize(): int
~getSpecies(): String
~getWeight(): int
~getAge(): int
~getFavFood(): String

Information

Information (housingId: String, name: String, age: int, sex: char, String species, int size, int weight, String favFood)
+ getHousingId(): String
+ getName(): String
+ getFavFood(): String
+ getAge(): int
+ getWeight(): int
+ getSex(): char
+ getSize(): int
+ getSpecies(): String