# Inheritance Exercise

# Muhammad Naufaldi – 2301942614

## Question set 1

## State the relationship between the Genus and Species objects.

Inheritance. Species is a subclass of a genus

## State the relationship between the Species and Specimen objects.

Aggregation

## Construct the unified modelling language (UML) diagram for the Species object.

-speciesName: String

+Species(species:String, genus:String)

+setSpeciesName(species:String):void

+getSpeciesName():String

+toString():String

+equals(species:Species)bool

Species

....

....

Genus

## Outline two ways in which the programming team can benefit from the way the relationships between the three objects, Specimen, Species and Genus, have been represented in the code.

* Classes can be accessed through other class
* Make 2 classes at oncce

## The Genus class implements a toString() method that produces an output string that is different from the one produced by the toString() method in the Species class. Consider the following code fragment:

## Species human = new Species ("homo", "sapiens"); System.out.println(human.toString() );

### Outline why calling the toString() method in this code does not cause an error.

Because the super class is already called in the constructor and all the data can be accessed.

### Identify the term for this property

Overriding

## Question set 2

1. Define the term encapsulation.

Bundling data and methods in one unit.

1. Outline two benefits provided by encapsulation.

* Protect the data from an access outside
* Makes it easier to understand

1. Identify an accessor method in the Specimen class.

* getName()
* getCage()
* getTOA()

1. Identify an instance variable in the Specimen class.

private Species toa;

1. Construct code for the Genus object including a constructor, accessor methods and a toString() method.
2. public class Genus extends Family {
3. private String genusName;
4. public Genus(String genusName, String Family){
5. super(Family);
6. setGenusName(genusName);
7. }
8. public String getGenusName() {
9. return genusName;
10. }
11. public void setGenusName(String genusName) {
12. this.genusName = genusName;
13. }
14. public String toString(){
15. return "Genus: " + getFamilyName() + " " + speciesName;
16. }
17. }
18. The Specimen object could have been designed as a sub-class of the Species object.

Outline one advantage and one disadvantage of having the Specimen object as a sub-class of the Species object.

Advantage : You can have less variable ( remove toa)

Disadvantage: You needed to create genus too.

## Question set 3

1. Outline the changes that would be needed in order to add a description of each animal’s individual markings to the program.

public class Specimen {

private String name;

private int cageNumber;

private Species toa; // "Type Of Animal"

**private String description;**

...}

1. An array is used to store the Specimen objects corresponding to the animals in the zoo. Construct a method countSpecimens( Specimen[] animals, Species s ) that will output the number of specimens of the given species in the zoo.

 public int countSpecimens( LinkedList<Specimen> animals, Species s ){

        ListIterator<Specimen> iterator = animals.listIterator();

        int count = 0;

        while( iterator.hasNext()) {

            Specimen temp = iterator.next();

            if(temp.getTOA() == s){

                count = count + 1;

            }

        }

        return count;

1. Construct an algorithm in pseudocode for listSpecies( Specimen[] animals ), which will generate a list of the different species in the zoo.

listSpecies( LinkedList<Specimen> animals ){

ListIterator<Specimen> iterator = animals.iterator();

LinkedList<Species> Species = new LinkedList<Species>;

While(iterator.hasNext()){

Species temp = iterator.next();

If(Linked list species contains temp.getTOA()){

Continue;}

Else{

Add temp.getTOA() to linked list species;}

}

Return linked list species

\*Its pseudocode as requested

## Question set 4

1. Identify the features of an abstract data type (ADT).

It has operations and functions that are necessary for the data type that is being used.

Doesnt have an implementation

1. Construct a method, makeList(), that builds a linked list containing the Specimen objects from an input array. LinkedList makeList( Specimen[] animals ) { }

LinkedList makeList( Specimen[] animals ) {

        LinkedList temp = new LinkedList<Specimen>

        for(int i = 0; i < animals.length; i++){

            temp.add(animals[i])

        }

        return temp;

    }

1. Construct a method, makeSpeciesList( LinkedList animals ), that will return a linked list of Species objects, one for each animal specimen present in the zoo. Note that the list returned will contain duplicate Species objects if the zoo has more than one specimen of a given species.

LinkedList makeSpeciesList( LinkedList<Specimen> animals ){

        LinkedList tempL = new LinkedList<Specimen>

        ListIterator<Object> iterator = animals.listIterator();

        while( iterator.hasNext()){

            Specimen temp = iterator.next();

            tempL.add(temp.getTOA());

        }

        return temp;

    }

1. Construct a method, makeSpeciesListUnique( LinkedList allSpecies ), that will take a linked list of Species objects as described in (c) and produce a new linked list that contains only one Species object for each species in the zoo.
2. LinkedList makeSpeciesListUnique( LinkedList<Specimen> animals ){
3. LinkedList tempL = new LinkedList<Specimen>
5. ListIterator<Object> iterator = animals.listIterator();
6. while( iterator.hasNext()){
7. Specimen temp = iterator.next();
8. if(tempL.contains(temp)){
9. continue;
10. }
11. else{
12. tempL.add(temp.getTOA());
13. }
15. }
16. return temp;
17. }