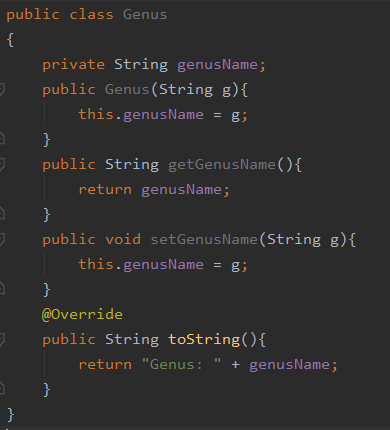
Question set 1

1. Inheritance / “is a” relationship
2. Aggregation / “has a” relationhip

|  |
| --- |
| Species |
| -speciesName:String |
| +Species(s:String, g:String)  +setSpeciesName(s:String):void  +getSpeciesName():String  +toString():String  +equals(s:Species):boolean |

1. It makes testing codes easier and helps avoid duplicate codes since all the functions from the parent class are inherited hence no need to repeat and retest codes in the new class.
   1. Since species inherits the toString from genus class, the toString that was initialized in the species class overrides the one in the genus class hence there are no errors.
   2. Polymorphism

Question set 2

1. Encapsulation is the idea of hiding specific data in class but making it accessible with accessor methods.
2. Data security, it prevents unwanted access from clients and prevents invalid data to be assigned to a variable. It allows people to access certain variable without revealing details of the variable
3. public String getName(){ return name; }
4. private String name;
5. 
6. Advantage: It will be easier to store due to the data being compiled into one group.

Disadvantage: This adds extra memory overload since it has to keep track of the data of both the parent and child class.

Question set 3

1. Add all of this into the class:

private String markings = "None";

public void setMarkings(String markings) {

this.markings = markings;

}

public String getMarkings() {

return markings;

}

Add the markings to the toString method as well.

public void countSpecimens(ArrayList<Specimen> animals, Species s ){

int sum = 0;

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

if (s == animals.get(i).getTOA()) {

sum++;

}

}

System.*out*.println(sum);

pseudocode for listSpecies(Specimen[] animals)

define a list for specimens and its length list = Specimen[10]

FOR each element in animals

FOR each element in list

if animals element by index i equals list element by index j then get to next index i

else animals element by index i not in index j append

print out list

Question set 4

1. An ADT has no implementation details that are known and has standard methods.
2. public LinkedList makeList( ArrayList<Specimen> animals ) {

LinkedList list = new LinkedList();

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

list.addFirst(animals.get(i));

}

return list;

1. public ArrayList<Species> makeSpeciesList(ArrayList<Specimen> animals) {

ArrayList<Species> species = new ArrayList<Species>();

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

species.add(animals.get(i).getTOA());

}

return species;

}

public ArrayList<Species> makeSpeciesListUnique(ArrayList<Specimen> animals) {

ArrayList<Species> species = new ArrayList<Species>();

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

int flag = 0;

if (species.size() == 0) {

species.add(animals.get(i).getTOA());

}

for (int j = 0; j < species.size(); j++) {

if (species.get(j) != animals.get(i).getTOA()) {

flag = 1;

}else {

flag = 0;

break;

}

}if (flag == 1){

species.add(animals.get(i).getTOA());

}

}

return species;

}