Department of Computer Science, Mathematics & Physics

COMP2232 – Object-oriented Programming Concepts

Semester 2, 2021-22: Assignment #2

Zoo Keeper v.2

Read this entire document before starting your assignment!

This assignment is to be completed in teams of TWO. All work is to have been created by the team submitting the work. Any form of copying, cheating or plagiarism will result in the assessment being disqualified and awarded a grade of 0 (zero).

The purpose of this assignment is to provide further exposure to object-oriented programming concepts, the Java programming language and the implementation of Graphical User Interfaces. The program must be implemented using a Graphical User Interface (GUI).

Overview

The Cave Hill Zoo holds many animals from around the world. The cages are clustered in four zones of the zoo: (A) African Savanna, (B) Amazonian Jungle, (C) Eurasian Wilds & (D) Frozen Tundra.

Within these areas (A, B, C & D) animals are separated and assigned to their own cage. Each cage is labelled using its area letter and a number (e.g. A-1, A-45, B-6).

New animals are registered and assigned zones & cages. Each is given a name. However, if another animal of the same species also has the same name, another name must be given to that new animal. (e.g. We can have Fred the monkey and Fred the lion but there cannot be two monkeys called Fred or two lions called Fred). Animals are also categorized as Herbivore, Omnivore or Carnivore.

The primary focus of a keeper is to ensure the health and safety of the animals. This involves three types of activities: feeding the animals to correct type and amount of food and administering the correct medication to sick animals. A program is required to assist in setting up & organizing the keeper's work.

How the Program Works

The program will have multiple screens (as follows):

1 - Welcome

The keeper is welcomed and asked for their name. The main screen is then loaded and a file containing a list of all of the animals & their details is loaded into the program.

2 – (Main) Feed & Heal Animals

For each animal in the zoo, the program will display its details.

Feeding Animals

This involves preparing the feeding list for all of the animals. The keeper must **select** (from a list) the type of food for the animal and **enter** the quantity of food. ***Note:*** *Herbivores may only eat hay, fruit or grain; carnivores may only eat fish or meat; omnivores may eat hay, fruit or fish (no meat or grain). The interface will prevent the user from selecting inappropriate food types for a particular animal.*

Once this has been done for each animal, the feeding list can be printed to a text file for the keeper.

The text file will contain the title "Feeding List" along with the date. The list will be grouped by zoo zone (A, B, C, D) giving the name of the zone, name & species of the animal, quantity & type of food they are to be given.

A summary will be included giving the total amount of each food type per Zone. This summary allows the keeper to determine how much food they much get from the supply warehouse and load onto each zone's delivery cart. (*See sample Feeding List file, at right*)

Once the keeper has entered all of the feeding information and created the summary, the keeper can click on a "Feed" button. This will use the feeding information to run a **simulation** of the feeding process. At the end of this, a summary will be output to a file. Any animals which were overfed will be indicated as having died. These animals will not be in the main list but in a separate list at the bottom of the file. (*Report title, date, number of animals successfully fed, number overfed, list of those animals who were overfed: cageID, name, species, original hunger status and the amount and type of food*)

```
Feeding List - 1 March 2022

(A) African Savanna
Gary Giraffe       2 Hay
Eli Elephant       1 Hay
Lenny Lion         3 Meat

Food Summary
3 Hay
3 Meat

(B) Amazonian Jungle
Manny Monkey       2 Fruit
Polly Parrot       1 Fruit

Food Summary
3 Fruit

(C) Eurasian Wild
Benny Bear         2 Meat
Yanni Yak          3 Hay
Randolph Reindeer  4 Hay

Food Summary
7 Hay
2 Meat

(D) Frozen Tundra
Penny Penguin      2 Fish
Pete Polar Bear    2 Fish

Food Summary
4 Fish
```

1- Sample of FeedingList.txt

Healing the Animals

Similar to feeding, this involves preparing the healing list for all of the animals. Each animal in the zoo is checked and for those who are sick (low health level), the keeper must indicate the quantity of medicine for that animal. ***Note:*** *healthy animals will* ***not*** *be permitted medicine. Herbivores may be given Herbicine; Carnivores – Carnicine; Omnivores – Herbicine or Omnicine. The interface will prevent the user from selecting inappropriate medicine types for a particular animal.* Once this has been done for each of the sick animals, the healing list can be printed to a text file for the keeper.

The text file will contain the title "Healing List" along with the date. The list will be grouped by zoo area (A, B, C, D) giving the name of the zone, name & species of the animal, current health level and quantity of medicine they are to be given.

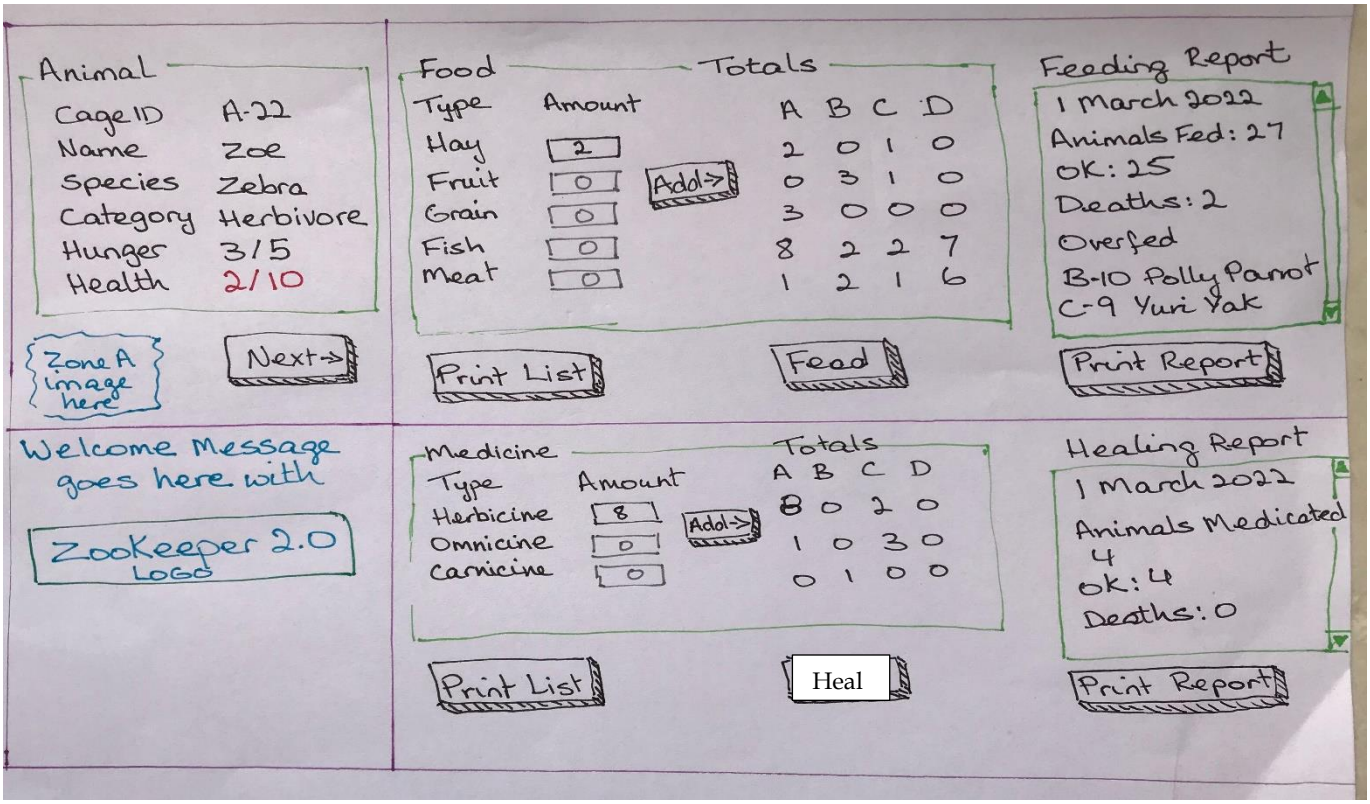
At the **end of the file**, will be a summary giving the total amount of medicine to be administered for the day. This summary allows the keeper to determine how much medicine they much get from the zoo's pharmacy. (*See sample Healing List file, below*).

```
Healing List - 1 March 2022

(A) African Savanna
Zoe Zebra (health: 2) needs 6 units

(B) Amazonian Jungle
Manny Monkey (health: 1) needs 4
units

(C) Eurasian Wilds
[No medical attention required]

(D) Frozen Tundra
Sally Seal (health: 2) needs 4
units

Summary of Medicine: 14 units
```

2- Sample of HealingList.txt

Once the keeper has entered all of the medicine information and created the summary, the keeper can click on a "Heal" button. This will use the healing information to run a **simulation** for administering medication. At the end of this, a summary will be output to a file. Any animals which were overmedicated will be indicated as having died. These animals will not be in the main list but in a separate list at the bottom of the file. (*Report title, date, number of animals successfully healed, number overdosed, list of those animals who were overdosed: cageID, name, species, original health status and the amount of medication they were given*)

Main Screen & Functionality



3-Screen Layout

Animal panel

- All fields are read-only.
- Provides details on the animal which the user is currently handling: Cage ID, Name, Species, Category, Hunger level, Health level.
- Health status of sick animals (health status less than 8) is indicated in a red font.
- Displays the image for the zone to which the current animal belongs.
- Only animals with health and/or hunger status less than maximum will be displayed.
- The "Next" button is disabled until the animal has been allocated food (i.e. food has been added via the food's "Add" button.
- Clicking the "Next" will load the next animal whose health and/or hunger status less than maximum.

Food panel

- Displays the food types and allows allocation of food to the animal.
- All fields initialized to 0.
- Only data entry fields for food types valid for the current animal's category will be editable. All other food type fields will be non-editable. (e.g. Herbivore: hay, fruit, grain are editable; other fields disabled.)
- Once a field contains a value > 0 then all other fields become disabled (non-editable - this prevents the user from feeding more than one type of food to a single animal).

- Clicking the "Add" button will save the food information by appropriately creating and storing a Meal object to the *feedingList* (data structure). The food will also be used to update the food **Totals** section appropriately. Values of the data entry fields are reset to 0, ready for the next animal. The "Add" (food) button is disabled.

(Food) Totals section

- This section is read-only and displays a running total of each type of food by zone.
- Once "Add" (food) button is clicked on the **Food** panel, the value is used to update the relevant total in this section. (Zone & food type will be used in updating the correct value.)

Feeding Report panel

- This panel is read-only, scrollable and will be populated once the "Feed" button is clicked.
- It displays results of the simulated feeding: date, total fed successfully and total overfed
- It also displays list of details of those Overfed (cage id, name, species); if there were no deaths (overfed) then no list is necessary.

Print (Food) List button

- Initially disabled; enabled when all animals have been addressed (i.e. entire list of animals has been successfully traversed).
- Clicking this button will print the Feeding list to a text file called "FeedingList.txt".

Feed button

- Initially disabled; enabled when all animals have been addressed (i.e. entire list of animals has been successfully traversed).
- Clicking this button will feed all of the animals (simulation) based on the meals in the *feedingList* (data structure).

Print (Feeding) Report button

- Initially disabled; enabled when all animals have been fed via the feeding simulation process.
- Clicking this button will print the feeding report to a text file called "FeedingReport.txt". (Report title, date, number of animals successfully fed, number overfed, list of those animals who were overfed: cageID, name, species, original hunger status and the amount and type of food)

Medicine panel

- Displays the medicine types and allows allocation of medicine to the animal.
- All fields initialized to 0.
- Only data entry fields for medicine types valid for the current animal's category will be editable. All other medicine type fields will be non-editable. (e.g. Herbivore: herbicine; other fields disabled.)
- Once a field contains a value > 0 then all other fields become disabled (non-editable - this prevents the user from allocating more than one type of medicine to a single animal).
- Clicking the "Add" (medicine) button will save the medicine information by appropriately creating and storing a Prescription object to the *healingList* (data structure). The medicine will also be used to update the medicine **Totals** section appropriately. Values of the data entry fields are reset to 0, ready for the next animal. The "Add" (medicine) button is disabled.

(Medicine) Totals section

- This section is read-only and displays a running total of each type of medicine by zone.
- Once "Add" (medicine) button is clicked on the **Medicine** panel, the value is used to update the relevant total in this section. (Zone & medicine type will be used in updating the correct value.)

Feeding Report panel

- This panel is read-only, scrollable and will be populated once the "Heal" button is clicked.
- It displays results of the simulated healing: date, total medicated successfully and total overdosed
- It also displays list of details of those Overdosed (cage id, name, species); if there were no deaths (overdose) then no list is necessary.

Print (Medicine) List button

- Initially disabled; enabled when all animals have been addressed (i.e. entire list of animals has been successfully traversed).
- Clicking this button will print the Healing list to a text file called "HealingList.txt".

Heal button

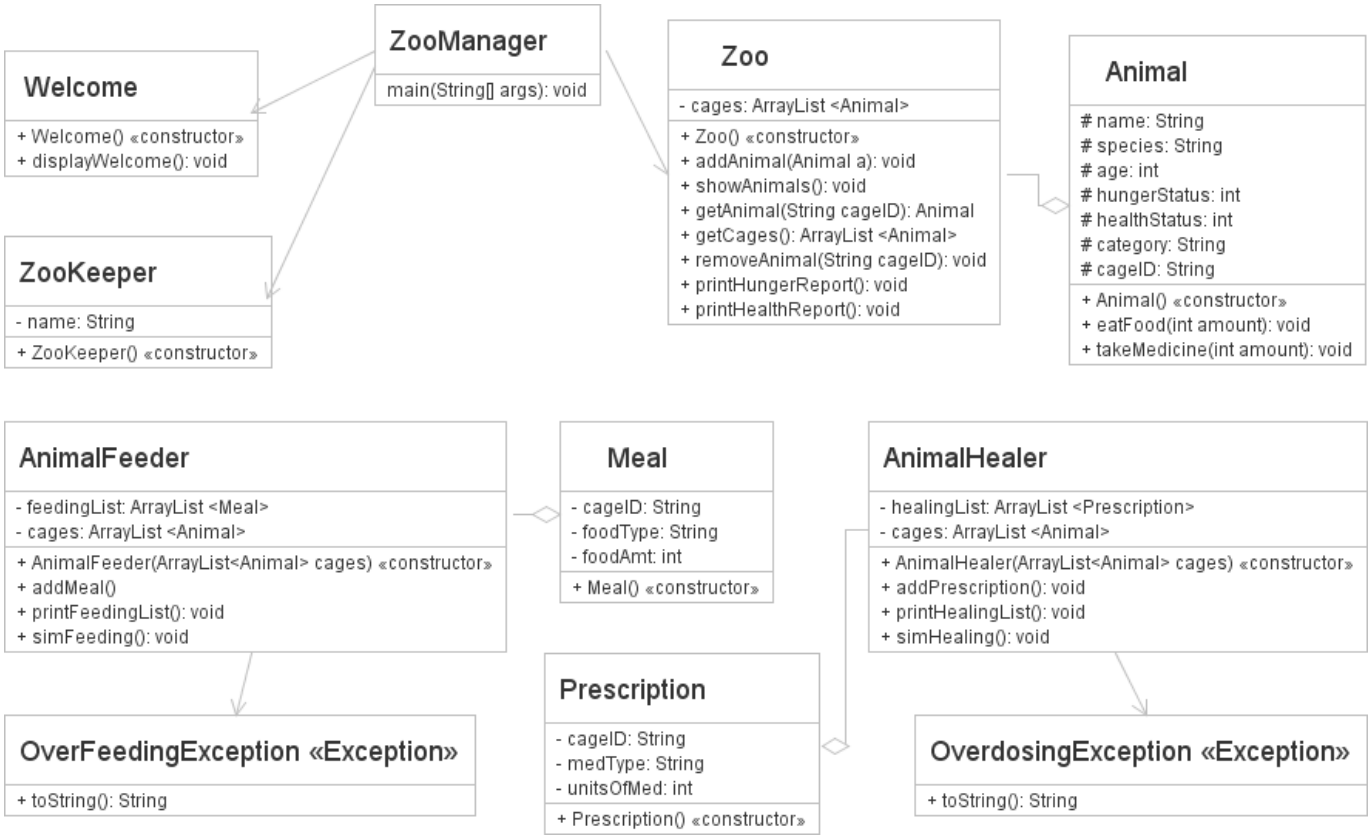
- Initially disabled; enabled when all animals have been addressed (i.e. entire list of animals has been successfully traversed).
- Clicking this button will heal all of the animals (simulation) based on the prescriptions in the *healingList* (data structure).

Print (Healing) Report button

- Initially disabled; enabled when all animals have been healed/medicated via the healing simulation process.
- Clicking this button will print the healing report to a text file called "HealingReport.txt". (Report title, date, number of animals successfully healed, number overdosed, list of those animals who were overdosed: cageID, name, species, original health status and the amount of medication they were given)

Coding

1. Your program must be implemented using a Graphical User Interface (GUI) with Java & its Swing components.
2. You will be provided with a sample data file for loading animals into the zoo.
3. You will use the classes depicted in the UML diagram below. (You may reuse and **update** any necessary files from Assignment #1.) You must create and use the classes defined. However, you may create additional classes and methods where necessary for the functionality of your program. Appropriate **mutators & accessors** are expected within the classes where there are data members present.



Animal (Animal.java)

- eatFood: accepts an integer amount of food and feeds the animal, updating its hunger status appropriately.
- takeMedicine: accepts an integer amount of medicine and heals the animal, updating its health status appropriately.
- Mutators and accessors for each data member

Meal (Meal.java)

- Used to represent a meal to be fed to a specific animal.

Prescription (Prescription.java)

- Used to represent medication prescribed to heal a specific animal.

Zoo (Zoo.java)

- Constructor: used to instantiate the animal list.
- addAnimal: accepts an Animal object and appends it to the list of animals
- showAnimals: displays details about each of the Animal objects within the animal list. If no there are no animals in the list, it should display an appropriate message indicating this.
- getAnimal: accepts the cageID as a parameter value, using it to locate a specific animal (object) in from the cages list; the object is returned
- getCages: returns the cages arrayList
- removeAnimal: accepts the cageID as a parameter value, using it to locate a specific animal (object) in from the cages list and removing it from that list
- printHungerReport: prints list of animals (name, species, hunger status)
- printHealthReport: prints list of animals (name, species, health status)

OverFeedingException (OverFeedingException.java)

- class to be thrown when an animal is overfed and dies
- toString: to be overridden with "Animal overfed and died."

OverdosingException (OverdosingException.java)

- class to be thrown when an animal is overmedicated and dies
- toString: to be overridden with "Animal overdosed and died."

Welcome (Welcome.java)

- class used to display welcome message

ZooKeeper (ZooKeeper.java)

- ZooKeeper() <<constructor>>

AnimalFeed (AnimalFeeder.java)

- AnimalFeeder(ArrayList<Animal> cages): receives cages (reference) and refers its own cage to it.
- addMeal: gets the necessary meal information from the user, creates the Meal object and adds in to the *feedinglist*. Error checking is expected.
- printFeedingList: prints the feeding list using details store in the *feedingList* data structure.
- simFeeding: used to feed each animal in the zoo, using the details in the *feedingList*.

AnimalHealer (AnimalHealer.java)

- AnimalHealer(ArrayList<Animal> cages): receives cages (reference) and refers its own cage to it.
- addPrescription: gets the necessary prescription information from the user, creates the Prescription object and adds in to the *healinglist*. Error checking is expected.
- printHealingList: prints the feeding list using details store in the *healingList* data structure.
- simHealing: used to heal each sick animal in the zoo, using the details in the *healingList*.

ZooManager (ZooManager.java)

- This is the driver class containing your **main** function.
- It will be used to instantiation and coordinate communication between Zoo, ZooKeeper, AnimalFeeder & AnimalHealer.

Marks

This assignment is worth **22%** of your final course mark. In addition to marks for code, overall marks will also be allocated for: Documentation/code formatting; User-friendly interface; Successful compilation; Correct execution.

There must be appropriate use of parameter passing and returning values from methods.

Marks will be deducted for:

- Violations of the Code Conventions. *Please ask if not sure.*
- Missing/inappropriate use of access specifiers
- Violations of **Assignment Rules**.

Deliverables

1. Assignment 2 is due for submission on **Sunday 3rd April 2022 by 11:55pm** via the Moodle/eLearning submission tool.
2. Only **ZIP** files will be accepted. No other compression types should be used. Failure to comply with this instruction will incur a penalty.
3. Ensure that you follow the **Assignment Rules** (see course page).
4. You must submit a **Group Plagiarism Declaration Form** with this assignment, if you do not accept the online one. Your submission will not be marked until your declaration is received.

PLEASE NOTE: The specifications for this assignment are subject to change. You will be notified if any such changes were to occur.