ITECH 2100

Programming 2
School of Science, Information Technology and Engineering

Major Assignment

Semester 2, 2017

Learning Objectives:

The learning objectives for this major assignment are:

K1. Explain the principles of inheritance, composition and their consequences

K2. Discuss basic object oriented concepts

K3. Explain the principles of event-driven programming

S1. develop object-oriented programs involving several interacting classes

S2. incorporate pre-written classes, including those from the SDK, into software solutions

S3. develop object oriented programs which involve both object oriented and event driven aspects

A1. design, develop, test and debug programs from supplied program specifications

Purpose

By completing this assignment, you will develop your understanding of and apply object-oriented programming techniques, in particular those relating to inheritance and polymorphism. You will develop skills and experience in the use of the Java Collections API classes and I/O classes. You will demonstrate your ability to successfully design and develop a working event-driven system.

You will develop a multi-class program to meet functional specifications applying design principles, and making use of the ArrayList and Map abstract data types.

You are encouraged to partner with another student and engage in paired programming to complete this assignment.

This major assignment will be submitted in 3 stages. Each stage will build on the previous stage. You will also be asked to demonstrate or describe your code to highlight particular concepts you have implemented. This demonstration/reflection will be individually assessed.

Assignment Overview:

This assignment requires you to complete various tasks involving classes you write with your partner. You may also use classes that you find in the Java SDK. The system that you will create is a Zoo Animal Management System (ZAMS). The ZAMS is a tool used by a zoo that manages multiple animal enclosures. The enclosures may be used potentially at different sites, for different purposes (e.g. free and open range site or reptile enclosure (indoors), or small enclosure, large enclosure, tropical habitat, elephant habitat, etc.) Each enclosure may also have different opening times. Each enclosure has a name and is associated with a particular venue/site and session time. For each enclosure, there are a number of available sub-enclosures available for guests to view animals. This IT system, ZAMS, will enable the head zoo keeper manager to manage information about the enclosures as well as the individual animals housed in each enclosure. It will be possible to inspect details for each animal separately. The details will include specifics of the animal as well as specific details regarding the individualised diet for each animal. There are generic classes created for each animal group housed in an area (e.g. African animals are housed in the African free range enclosure, so there is a class of animals: African animals). There will also be particular animal sub types (e.g. Lion, Elephant, Chimpanzee) .   The animal categorisation will determine the way that information is provided for viewing the animals as different enclosure may have different rules regarding feeding, opening hours and public access. Each animal type (e.g. elephant) will have information available in the system used to calculate the amount of food required for that animal and when medications are required for that animal type. Each individual animal (e.g. Lucy the elephant) has individual data stored such as ID, name, weight, vaccination status). You will be required to develop your application using Java in the Eclipse environment. You will create a GUI using the java.swing library package.

You will need to zip your project up when you are finished each stage and submit it as a single zip file to Moodle by the due date. You will also need to submit a report. The report will require some written and diagrammatic information describing your system. Please write clearly and IN YOUR OWN WORDS (plagiarism is NOT acceptable – refer to Course Description**).**

It is expected that each student pair will creatively design and author a unique system based on individual design choices that fit with the requirements listed in each task. Before you begin, read over the entire assignment and understand all the tasks. Acknowledge authors for all code that you submit using in code comments.

All code must conform to stylistic standards including proper commenting, appropriate choice of identifier names (including case), proper indenting and other readability issues. Otherwise, marks will be deducted. Consult http://www.oracle.com/technetwork/java/codeconventions-150003.pdf

Stage One is due in week 7. Stage Two is due in week 9. Stage Three is due in week 11.

In weeks 10 and 12, you will be interviewed by your tutor in your lab class. You will be asked questions about your code.

Stage 1a Designing and Documenting your system Due date: Week 7

Create a document named: surnameStudentIDAssign1.doc . In your document, you must include the following section headings:

- *Authors,*
- *System Overview,*
- *Class Diagram,*
- *Testing.*

System Overview

In the System Overview section you are to write 100-200 words describing your system and how it could be used in an imaginary scenario. You will need to review this after completing all tasks and ensure that your overview explains your entire system. State any assumptions you are making in your system. It is expected that you will create a system that can accommodate managing a number of enclosures. For each enclosure, there should be an associated site name, address, zoo keeper and a number of animals each associated with a unique item number in the enclosure on which it appears. Each enclosure object has related information such as opening hours and the enclosure item name. You may add further attributes such as description if you wish.

The entry price and opening hours or viewing hours that appears on the enclosure for a particular enclosure may be different from the default opening hours entered into the system. The viewing hours displayed on the enclosure depends on the type of animal. There should be a number of possible types of enclosure animals incorporated into your system. *Example* enclosure item types are as follows:

1. A standard enclosure will have the opening hours calculated based exactly on the default hours entered into the system. Entry price will be a standard price for these enclosures.
2. A premium enclosure item can be created with a mechanism to indicate that it is to have additional opening hours at various times of the year. There may be a surcharge will increase the price of the visitor entry based on a constant percentage mark up value specified by the zoo management;
3. It is possible to create a low entry cost special evening enclosure that is available after dark for visitors to pay and view a limited number of nocturnal animals at night ;

The ZAMS system will have at least 3 different enclosure item types (e.g. standard enclosure item, premium seasonal enclosure item or evening enclosure item). You may choose a name for each enclosure type and what the distinguishing features of each type of enclosure are. It is suggested (at a minimum) that each enclosure type will have a different process implemented for calculating the entry price. All enclosures will have a process for displaying the actual opening hours for each particular enclosure.

Each enclosure created is expected to have at least 5 associated enclosure animals. The total number of enclosure animals is limited to a maximum of 30 and your system should allow for multiple pages of enclosure animals. Your system will be able to add enclosure animals to the enclosure after it has been created. The enclosure details will include a name and description as well as details naming a data file that will be used to preload enclosure animals. There will be a number of general animal classifications. Each generalised class representing a generic enclosure type (e.g. Reptile, or African open range, or Australian). Within each enclosure created, there will be a number of different animals that have been classified to be housed in that enclosure. For example, if you have a Reptile enclosure, you will have a ReptileEnclosure class and then in that class, there will be scope to add a number of reptiles (e.g. Snake, Lizard). Each actual animal created will be instantiated as one of a specialised type of animal (e.g. reptile) that inherits from a very generic animal class. (Lizard is a Reptile, Reptile is a Animal)

Class Diagram

In this section, you are to provide a class diagram for **every** user authored class you anticipate in *your* first version of the system. The first version will include a **text based enclosure driven system** with classes that will be used to create objects to represent 3 different enclosures, as well as different enclosure animals of various types that belong to each enclosure. You are expected to use inheritance so that you have an abstract enclosure animal class and specialisation classes for different types of animals.

Testing

As you develop your classes and methods, you will write testing code to verify that your code is working correctly. In this section, you will describe your initial chosen test data and explain how that data will help you test your code. Make sure that you choose a variety of test cases to ensure you can be satisfied that your code is working. Describe your test cases and include screen shots of your testing results (so far) in this section.

Stage 1b - Task 2 Creating classes for enclosures and enclosure animals using associations and inheritance. Due date: Week 7

Create a new project in Eclipse called Assignment 1.

Within this project create a package called EnclosureManagementSystem.

1. Author six classes within your newly created package: General classes to create first are Enclosure, Animal and your Driver class. In addition to these, you are to create at least THREE classes representing possible types of enclosure animals that are specializations on the general abstract class: Animal. All Animals have (at least) the following attributes: animalName (String), animalWeight(double). You will also create an abstract class Enclosure and a number of specialisation classes for different enclosure types. All Enclosures will have (at least) the following attributes: enclosureName(String), enclosureManager(Person), enclosureOpeningTime(String), enclosureClosingTime(String). When designing your specialisation classes, you may choose further additional attributes particular to each specific enclosure type. Each enclosure type will have different getViewingPrice() and getOpeningHours() methods that will override the getViewingPrice method in the abstract class Enclosure and calculate the price and opening hours to display on the enclosure based on the rules for each enclosure.

2. You must provide at least 2 constructors for each particular specialised enclosure type class:
    a. A default constructor which assigns each instance variable a default value. The String variables (e.g. enclosureName) should be initialised to "unknown", the price should be initialised to a minimum default price and the opening hours set to appropriate default opening hours for that type of enclosure. Any object fields should be initialised to null.
    b. A constructor with parameters which assign values to each instance variable. Note that the values to initialise the values and objects should be passed in using arguments when the constructor is called.
3. Create a seventh class, EnclosureAnimalDatabase that you will use to manage a group of enclosure animals. This class will be associated with the enclosure class. The enclosure will have an attribute that is referencing an object based on this class. The EnclosureAnimalDatabase class will contain an ArrayList of EnclosureAnimals as well as an appropriate method named *insert* that could be used to add a particular EnclosureAnimal into to the list of EnclosureAnimals. The insert method should take 2 parameters: an integer value representing the enclosure animal number (each animal in the enclosure will have a unique animal number for that enclosure) and an animal object. Also author a method named *retrieve* that will get the enclosure animal from a given index position from the Arraylist. You can assume initially that the array index position corresponds to the animal number in the enclosure.
4. Author get and set methods for your classes for instance variables where appropriate.
5. Write a toString() method in each class that will return a String containing all the relevant data for each of your objects.
6. In each of your specialised enclosure item classes, create the getViewingPrice method to perform an appropriate price calculation based on the data in the object.
7. Create a TestDriver class. In this class, provide a main method containing code to implement your test plan to test all constructors and methods including your toString () method, for each class you have written. Run your tests and make sure your class is behaving correctly.
8. Write further code in your test driver so that you can create an enclosure for a particular zoo site and add animals to that enclosure. This should be menu driven so that you can manually input some data. You may also like to hardcode some test cases.
9. In your documentation, outline your test plan (For example, outline what objects you created in order to demonstrate that your classes written so far are working correctly).
10. Explain how you have tested your code and the results of your testing. **Include screen shots** in your documentation demonstrating the output when you tested your code.
11. **Document all methods in your code using comments**. *You may be creative with the individual class attributes you associate with your objects and methods you create. If you have any difficulty with ideas, discuss this with your tutor.*

After submission of stage 1, your tutor will give you feedback to help you progress to stage 2 and stage 3.

Stage 2 Convert your Animal to become an abstract class, Populating your Enclosure from an input file, producing output to a report text file, creating an enclosure occupant list, using abstract data types (Map). Due Week 9

1. Create an input file MyZoo.txt containing at least 10 different EnclosureAnimals and at least 3 different EnclosureTypes (ensure you have at least one of each different enclosure type). The data for each will include the Name of the enclosure and then a number of animals. The category for each animal should be identified.

   Your file may look something like this: (it doesn't have to look exactly like this – the format is up to you). In this example, for the venue: "Kathleen's Zoo", there are 3 enclosures: African Safari, Standard Australian, Nocturnal.

```
Kathleen's Zoo

African Safari Enclosure
Elephant
Ellie
0123110
400kg
Up-to-date
3 years
Open range
Elephant
Edward
01231102
350kg
Due
0.5 years
Open range
Zebra
Zoolah
0120021
150kg
Up-to-date
Open range
Giraffe
Gerry
0249901
100kg
Up-to-date
End-enclosure
```

```

Standard Australian Enclosure
Wombat
William
0112232
70kg
Up-to-date
Echidna
Ellie
0112245
3kg
n/a
End-enclosure

Nocturnal Enclosure
Snake
Sally
0023114
2kg
n/a
Frilled Neck Lizard
Freddy
0024153
.8kg
n/a
End-enclosure

END DATA
```

2. In your submission report describe the actual format of data in your text file. Name the fields of data that will be in the file and the order in which they will be found. (e.g. Venue or Enclosure name will be first and will be a string on one line, then a number of enclosure animals will follow ….etc.)

3. Modify your Animal class so that it is an abstract class. Change your getViewingPrice and SetPrice methods to be abstract methods in EnclosureAnimal so that they must be overridden and provided in each inherited special enclosure item class.
4. Write a method on your enclosure class to read the enclosure and enclosure animals information from the file, and create a new restaurant enclosure. This includes populating the enclosure with enclosure animals based on data from the file. You will need to create the correct enclosure item type based on information in the file.
5. Write a toString() method that will generate an output string detailing the enclosure. Use polymorphism where possible.
6. Extend your system so that it will allow a zoo keeper to display all enclosures and generate a report providing details based on a particular selected enclosure and some or all animals in that enclosure. The detailed report can report on 1 or more enclosure animals that have been selected and provide details about the selected animals. The report should also display information such as entry price and opening hours for the enclosure.
7. Add further code to your driver class and write a test method which tests your system using a text based menu system. Test using a scenario involving displaying an enclosure and asking the user to select animals to report upon from that enclosure. In your report document, describe the testing scenario and the expected behaviour of your system. Provide screen shots of your system testing in your final report. Write a method to appropriately populate your enclosure with details read from a file. **Ensure you have comments to describe each of your new classes in your code.**
8. Add a method to output all the details about an enclosure and all its animals saved to a file. This output report must for each enclosure save details particular to the enclosure, including the enclosure animals and their data.
9. Test your classes to check they are working correctly. Document your testing process – describe your test data and the outcome of your testing. Include screen shots in your report document illustrating your testing process and the outcome of your testing.
10. Using a Map data type to organise a collection of enclosures, create an EnclosureDatabase class that manages a collection of enclosures. In this class, you will have a class attribute named myEnclosures that refers to a Map data type in which you can store a number of enclosures. Each enclosure will have a unique enclosure code (String of 5 characters). This enclosure code will be used as a key to look up the enclosure and retrieve it from the Map. The map data type enables you to look up and easily retrieve details for a enclosure item given its code off the printed enclosure. Write some methods in this class to get a particular enclosure given its key and to add a new enclosure.
11. Further test your system to ensure it is working, document your testing process.
12. Update your UML class diagram so that it is consistent with your code. (You may use Enterprise architect to import your code and generate the class diagram).

After you have submitted stage 2, feedback will be provided by your tutor to enable you to progress to stage 3. In week 10, you will be asked to demonstrate and discuss with your tutor: your UML, system testing and explain the code for step 10.

Stage 3 creating a patron GUI Interface Due Week 11

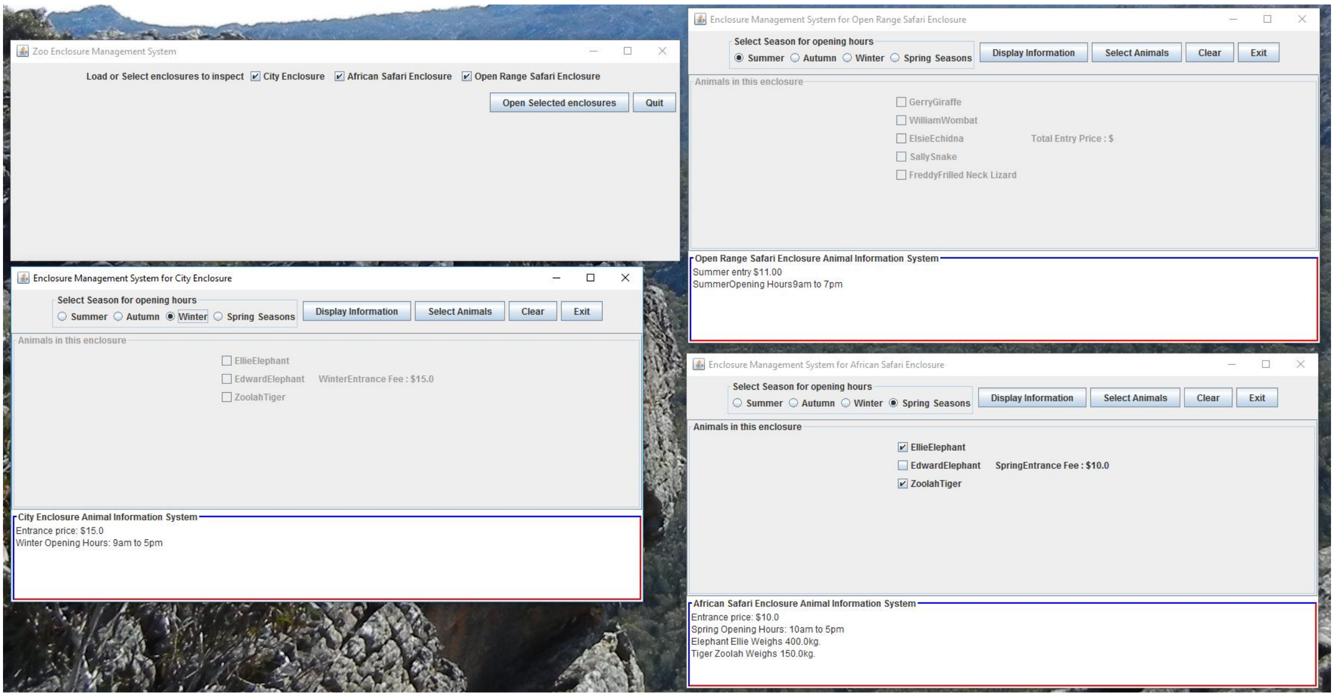
In this stage, you will create a GUI interface for your system. This will replace your text based enclosure in the TestDriver created in stage 2. This may be used on a touch screen for visiting patrons at the front of each enclosure. There is example code on moodle that demonstrates how to create a GUI. You are strongly encouraged to refer to those examples.

GUI Design

Draw a schematic sketch of your GUI design. On your diagram, highlight where you intend each panel to be and the contents of each panel (e.g. buttons, lists, radio buttons, labels). Where you have content to be displayed based on an object in your code, note this on your diagram. Also, indicate the events you expect your GUI to respond to and what happens following each event. For example, “when the button < X> is pressed, this event will trigger a call to method <Y> which will perform a calculation using the values in the selected list and display the result in the text field in the center panel. “

Design and coding of the GUI using the swing library

Create a new class that will manage your GUI System. This class will take one parameter when constructed, a reference to an Enclosure object. So, if you invoke your GUISystem class twice (with two different enclosures), you will create two separate windows – one for each enclosure. Figure 1 gives an example GUI where 3 different enclosures were loaded in a demonstration system (your system does not need to look exactly like this one).


Zoo Enclosure Management System
Open Selected enclosures
Quit
Enclosure Management System for City Enclosure
Enclosure Management System for Open Range Safari Enclosure
Enclosure Management System for African Safari Enclosure
Select Season for opening hours
Display Information
Select Animals
Clear
Exit
Animals in this enclosure

Load or Select enclosures to inspect ☑ City Enclosure ☑ African Safari Enclosure ☑ Open Range Safari Enclosure

○ Summer ○ Autumn ● Winter ○ Spring Seasons

□ EllieElephant
□ EdwardElephant WinterEntrance Fee : $15.0
□ ZoolahTiger

City Enclosure Animal Information System
Entrance price: $15.0
Winter Opening Hours: 9am to 5pm

● Summer ○ Autumn ○ Winter ○ Spring Seasons

□ GerryGiraffe
□ WilliamWombat
□ ElsieEchidna Total Entry Price : $
□ SallySnake
□ FreddyFrilled Neck Lizard

Open Range Safari Enclosure Animal Information System
Summer entry $11.00
SummerOpening Hours9am to 7pm

○ Summer ○ Autumn ○ Winter ● Spring Seasons

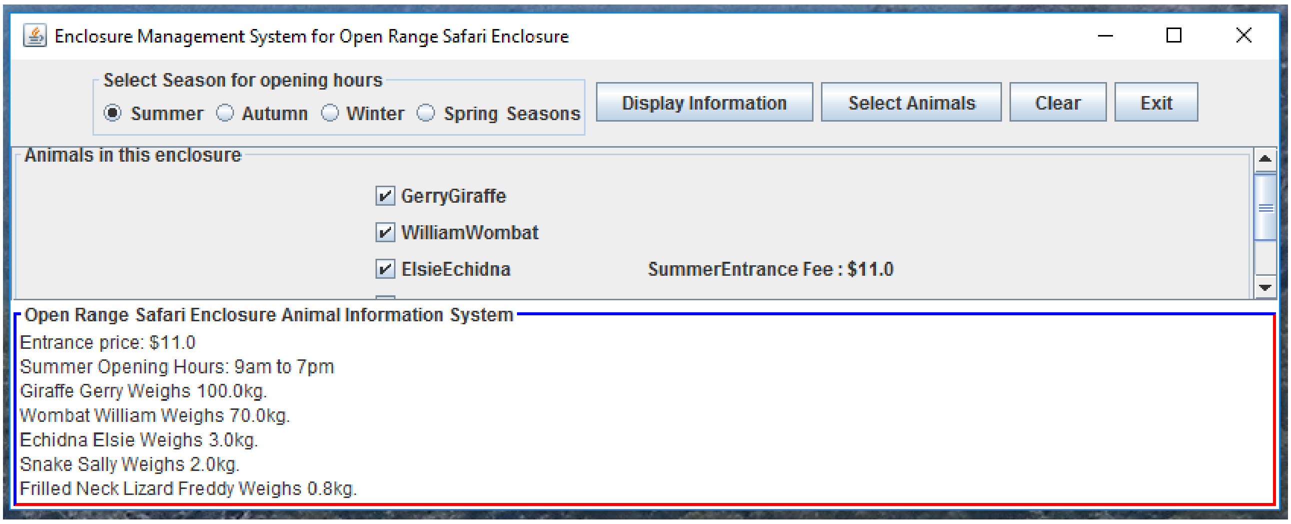
☑ EllieElephant
□ EdwardElephant SpringEntrance Fee : $10.0
☑ ZoolahTiger

African Safari Enclosure Animal Information System
Entrance price: $10.0
Spring Opening Hours: 10am to 5pm
Elephant Ellie Weighs 400.0kg.
Tiger Zoolah Weighs 150.0kg.

Figure 1. Sample GUI interface when 3 enclosures have been selected to inspect

The example shown in Figure 1 is indicative of some of the requirements for your GUI window for each Enclosure. You will also have the initial startup GUI window where you provide a list of enclosures available to view. You must be able to select an enclosure to display and then when you click on the button to open that enclosure(s), each enclosure will appear in a new separate window. It is expected that no submissions will be identical, so ensure that your GUI has some unique elements to it. Be creative! It is expected that each student pair will creatively design and author a unique system based on individual choices that fit with the requirements.

The Sample GUI in figure 1 shows that in the case of the African Safari Enclosure, the user has selected some animals following selection of the button “Select Animals” which then enabled the Animals in the enclosure checklist to be enabled. When the user selects individual animals, then details of these particular animals are shown in the bottom information display. If there are a lot of animals in the enclosure, or if the window is resized to be smaller then the middle area in which the checkboxes of animals are shown should have a scroll bar allowing the user to scroll through checkboxes for multiple animals. This is shown in Figure 2 below in the GUI for the Open Range Safari Enclosure.



Enclosure Management System for Open Range Safari Enclosure

Select Season for opening hours

● Summer ○ Autumn ○ Winter ○ Spring Seasons

Animals in this enclosure

☑ GerryGiraffe
☑ WilliamWombat
☑ ElsieEchidna

SummerEntrance Fee : $11.0

Open Range Safari Enclosure Animal Information System

```
Entrance price: $11.0
Summer Opening Hours: 9am to 7pm
Giraffe Gerry Weighs 100.0kg.
Wombat William Weighs 70.0kg.
Echidna Elsie Weighs 3.0kg.
Snake Sally Weighs 2.0kg.
Frilled Neck Lizard Freddy Weighs 0.8kg.
```

Figure 2. Enclosure GUI showing scrollbar on animal display and select area

Note: your GUI does not need to create /edit or save your enclosures. Those administrative functions from stage 2 could be done using the text based enclosure by the zoo keeper. The GUI is primarily used for patron touch screens at the front of each enclosure to provide information for visitors.

You must satisfy the following requirements for full marks on coding for stage 3:

- Display in a startup GUI window the list of Enclosures available to inspect, select one or more (using a checklist) to then view to edit. For each enclosure selected, create a new GUI window for that enclosure.
- Create and use at least 2 buttons on the start up GUI, to open an enclosure and to exit the system
- Include a button that 'does' something based on the enclosure (e.g. calculate and display the food requirements for animals in the enclosure, or display next feeding time, or display when vaccinations are next due for each animal etc.) **NOTE THIS IS NOT SHOWN IN THE SCREEN SHOTS FROM THE DEMO SYSTEM ABOVE**
- Use check boxes to allow a user to select particular enclosure animals to view details about. These should be populated by a vector list and should be consistent with the enclosure animals that are stored for that enclosure.
- Display in a scrollable output text box the animal details based on user selection.
- In addition, your code must be well designed into methods applying the principles of coupling and cohesion. You must deal with exceptions appropriately.

Allocated Marks: See Course Description

You will be marked on the design and quality of your code, your documentation and also your ability to individually answer questions based on this code in an interview.

Due Date: See Course Description

Please refer to the Course Description for information relating to late assignments and special consideration.

Plagiarism:

Please refer to the Course Description for information regarding Plagiarism.

Assignment Submission:

Assignments must be submitted by the due date and your assignment should be completed according to the General Guidelines for Presentation of Academic Work (http://www.ballarat.edu.au/aasp/student/learning_support/generalguide/) and programming standards (http://www.oracle.com/technetwork/java/codeconventions-150003.pdf).

The following criteria will be used when marking of your assignment:

- successful compilation
- successful completion of the required tasks
- ability to answer questions about your code and concepts demonstrated in your code
- adherence to the guidelines provided
- quality of code that adheres to the programming standards for the Course including:
  - comments and documentation
  - code layout
  - meaningful variable names

You are required to provide the following documentation:

- a statement of what has been completed and acknowledgement of the names of **all** people (including other students and people outside of the university) who have assisted you and details on what parts of the assignment that they have assisted you with
- a table of contents and page numbers
- tasks requiring written responses
- class diagrams
- list of references (including websites, the text book and any other resources) used (APA style); please clearly specify if none have been used.

For each stage submission, submit the code project source code and your documentation in one zip file to Moodle.

Student ID: ____________________ **Student Name:** __________________________________

Stage 1a - documentation	**2**
System Overview Description	**0.5**
Class Diagrams	**0.75**
Test plan	**0.75**
Stage 1b - coding	**4**
Demonstrated appropriate use of abstract classes and specialization classes with appropriate use of overriding of methods for getViewingPrice()	**1**
Appropriate constructors, private variables where appropriate	**0.5**
Appropriate use of parameters in method design	**0.5**
Appropriate use of an ArrayList collection class	**1**
Evidence of testing	**1**
Comments	
Total Marks	**/6**

Student ID: ____________________ **Student Name:** ________________________________

Stage 2 (to be assessed based on demonstration in week10)	**9**
Demonstrated reading from and writing to text file successfully	**2**
Documentation of file format in report, including example data from test data.txt file	**0.5**
Use of abstract class and abstract methods with overridden methods in inherited classes.	**0.5**
Additional functionality to display the entry viewing price and opening hours for an enclosure.	**1**
Appropriate use of map abstract data type for storing and retrieving a number of enclosures, using a String code or name for key.	**1**
Demonstrated use of polymorphism where appropriate.	**1**
Text based menu driven system driver	**1.0**
Testing – thoughtful, thorough, demonstrates above requirements work	**1.5**
Updated UML diagram	**0.5**
Comments	
Total Marks	**/9**

Student ID: ____________________ **Student Name:** _________________________________

Stage 3	**9 *(+2)***
GUI design sketch and description	**0.5**
GUI works with correct functionality – with start up GUI and new windows for each enclosure	**1**
Buttons implemented using appropriate listeners on all windows	**1**
Text fields used for output of enclosure and animal details	**1.5**
Check boxes populated based on a vector created from the enclosure animals on the enclosure	**1**
Enclosure Name displayed in window	**0.5**
Scrollable panel displays all enclosure animals	**1.5**
Well designed code using methods, parameters and variables appropriately	**2**
Bonus marks awarded for extra functionality	***2***
Comments:	
Total Marks	**/9**

Student ID: ____________________ **Student Name:** _________________________________

Week 12 Interview (individual mark)	**6**
Your tutor will ask you to demonstrate and explain features in your code as well as provide examples showing the following concepts : use of information hiding, abstract data types, data management façade class, try and catch clauses, inheritance and polymorphism.	
Comments:	
Total Marks	**/6**