Assignment 1 – Creating a Live Eco-System

1. Overview of the Assignment

1.1 Learning objectives

By doing this assignment you will learn how to:

- Identify problems that need solving
- Design a project with classes and execution flow
- Design classes with strict Encapsulation
- Implement classes and execution flow per the design
- Draw graphic primitives (rectangle, oval, triangle, quad, arc etc.)
- Compose complex shapes with primitives
- Animate the complex shapes with translation and rotation

Big picture for assignment 1-3: In these series of assignments you will create a simulation of an ecosystem existing in nature, with animals that move, eat and die and predators that hunt them. We will build on concepts you already know from IAT167 about building event driven interactive object oriented graphical applications. In the process you will expand on your knowledge of OOP concepts, learn Java and expand your practical understanding of concepts of object oriented programming. Our goal is for you to be comfortable to write and/or modify code to solve problems you may encounter in interactive applications on the web front and back end, mobile apps, VR, etc. The sequence of assignments has requirements that represent a set of small problems which give you an opportunity to apply concepts we learn in the lectures. However, the assignments are not the goal. Ultimately, you will be able to apply what you learn to new problems is what we are after.

There will be three assignments with fairly strict requirements to give you structure in which you should practice new skills. This should equip you with tools for the fourth assignment, in which we let you choose what you will build with the skills you have acquired.

The goal of **the first assignment** is to get you going. The problems you are asked to solve are what you have already learned in IAT167 and involve with animation, collision detection, and working with objects. You will learn how this is done in Java, which is a full version of the language that Processing is derived from.

1.2. Overview of Assignment 1

In this assignment, you will create a simulation of a vibrant garden or an area of a forest, which is filled with appropriate live creatures and food hunting activities (please note, **we don't accept pond or sea filled with fish**, because they will get too close to some of the lab tutorials down the road)*. In the process you will learn code design and programming using object oriented and event-driven concept model and Java as the primary library, plus incorporating some external libraries when they become necessary.

We want to get you into the habit of identifying problems that need solving, and then figuring out an algorithmic solution before you start coding. This involves coming up with a concept and relevant problems, identifying classes (and objects) your solution will have, and what these objects will be responsible for, i.e. what their methods will do. Hence, this assignment will have three separate submissions, at three different times:

Phase 1: A design document
Phase 2: An application implementing your design
Concept assessment: A reflection on concepts involved in your coding (like a follow-up quiz)

Specifically, **Phase 1:** concept design that includes identification of problems to solve, initial design of classes and execution flow, and (when necessary) Pseudocoding Programming Process (**PPP**) for some crucial methods. **Phase 2:** implement the classes and execution flow per the design**, draw and move the creatures using transformations, adding food objects, and interactions between the creatures, environment, and foods.

Please read the following detailed specifications carefully to fulfil your assignment, and the submission requirements to submit (pay attention to the penalty for violation). Make sure you check the grading rubric to learn what we are looking for.

* **Please note violation of this requirement will lead to a penalty of -10pts (i.e. -2pts for Phase 1 and -8pts for phase 2).**

** Like any pre-design, you can't predict and design everything perfectly, therefore it's normal for you to dynamically adjust and modify the design when it turns out to be necessary in the process of its implementation

2. Phase 1 requirements (Due: Sept 26 Mon 11:59pm)

1) The report should consist of 4 sections: **Project description**, **Proposed structure, Main execution flow,** and **PPP for important methods** *
2) Project description needs to outline the project regarding what it is about, environment features, types of creature involved, their typical behaviors, constraints for their behaviors, etc.
3) A list of classes, each of which includes the rationale that you thought of them for existence in your project, potential fields, and potential methods. For fields and

methods, provide a description for each about their responsibility (what they are used for)

4) For this assignment, the list of classes must include at least a class for the environment, a class for the main creature, and a class for the food. More specifically, **a)** environment must specify its dimensional sizes, background color etc., and a method to render itself. **b)** The creature class should incorporate relevant **fields** (at least size, position, velocity, scale factor, among others) and **methods** to draw the creature using graphic primitives, to move itself, to detect edges and change moving direction, to eat food etc., plus a **constructor** that initializes each of the fields with some parameter. **c)** The food class should have similar structure to the creature class, except that it doesn't need methods for moving or edge detection, but a respawn method that generates a new food object at a random location within the garden
5) Finally specify the execution flow that will happen in the main module (typically the panel class that is a JPanel's subclass), which indicates things like what objects or collection of objects need to be included and why, and what of their methods need to be called in what panel class' methods, providing a description for each about what the calling will do on the screen.
6) All the classes must be designed with **strict Encapsulation (except for constants) – to indicate, please use "-" sign before any field or method that is "private" and "+" sign before any constant or method that is "public". Also remember to maintain minimal public interface by providing only essential public methods.** Also the names of the class, fields and methods must follow the naming convention with appropriate upper- or lower case for the **initial** and ***CamelCase*** thereafter
7) **PPP**: for the creature class, write pseudocode *in English* to outline the procedures to implementing its methods to draw, move, do edge detection/resolution, and eat food.

* **Please refer to the sample report provided (in a separate file) for what the report might look like. Please note the report is incomplete, which just serves as an approximate template for your reference.**

3. Phase 2 requirements (Due: Oct 3 Mon 11:59pm)

First of all, the implementation **must** be done using **Java 8 System Library – please try to avoid versions later than Java 11 to avoid compatibility issues to prevent it from running)**. You will get **0** if you use **Processing library** (except for using its **PVector** class for motion) or any other 3rd party libraries.

1) Create the class for the environment per your design in Phase 1, which should have relevant fields for its specifications (in constants) and a draw method that draws a general background and at least two decorated features to represent the environment. E.g. some plants with flowers, trees, clouds, sun, etc. Please note the environment size, to make it obviously stand out of the display window, must be smaller than the window's panel by at least 20 pixels on each of the sides.

2) Implement the class for the type of creature that you have chosen, which should incorporate relevant **fields** (at least positions, velocity, color among others) and a **method** to draw the creature using graphic primitives, plus a **constructor** that initializes each of the fields with some parameter
3) The creature must consist of a head, eyes, a body, and at least two different visual features on the appearance of the creature, which will be determined randomly. For example, for a tiger, randomly determine the number of stripes, slimness of the body, and proportion of tail size etc.
4) Your creature class must use PVector (from Processing library) for position and velocity (if you haven't done so in your initial design of Phase 1, please do it now in the code).
5) Draw your creature using transformations (translation for positioning and movement plus scale for appearance).
6) Your creature should move. Start at the center of the environment and move in a randomly determined direction with constant speed.
7) The visual appearance of your animal should look "natural" when moving in different directions, including facing its moving direction, micro-animations for moving e.g. alternating legs when walking, flapping wings when flying (**Hint:** you can refer to the relevant techniques that you have learned in IAT-167 for so doing)
8) When the creature hits any of the edges of the environment, make it change its movement by turning in some natural way by way of appropriate transformation and PVector function calls.
9) When the creature turns its direction, its body or any of its part should not swing out of the edges of the environment
10) Create the class for food with relevant fields (again use PVector for positioning), a constructor that initializes each of the fields with some parameter. Render the food as something meaningful for the creature, e.g. worm for a bird, rather than simple dot.
11) Before food appears, the creature will simply move around hunting for food. When food appears, your creature will move toward it until it reaches the food location (or is close enough to reach it – justify your algorithm) and eat it (i.e. make it disappear).
12) When the food is reached, it is consumed and new one appears at a random location **with a delay of 5 seconds** (Hint: you can achieve this by creating a custom timer that counts the number of frames elapsed since certain time as you have learned in IAT-167, but please note that the framerate is under your control rather than 60 fps as in Processing)
13) All the classes must be implemented with **strict Encapsulation (except for constants) – with minimal public interface methods** (The rule of thumb to check: no redundant public methods that are not called upon in your project)**.** Also the names of the class, fields and methods must all follow the **naming convention** with appropriate upper- or lower case for the **initial** and ***CamelCase*** thereafter.

3. Bonus

(**Please note: if you have bonus features implemented, you must notify your TA by providing a comment to specify at the top of the panel class, or you may lose the bonus credit due to without notice by the TA**)

You can win up to **2 pts bonus marks** by adding some micro-animated features to your creatures or environment. For instance, for the creature, it could be, when appropriate, one or more of the following micro-animation features: eye rolling, whisker waving, tail wagging, wiggling body while crawling (up to 0.5pt each), some interesting and sensible mini-animation while food got eaten (e.g. splash of crumbs from mouth, up to 1 pt), or make food a moving creature as well, so that it hangs around somewhere and will try to escape when the creature gets close enough to it, and you'll have to speed up to catch it before it moves out of garden (up to 2 pts). Please note the bonus caps at 2pts even if the add-up of your bonus features goes above it.

4. Submission and Grading Instructions

Phase 1 (4 pts)

- Submit a design report in PDF that must be named with the following format: ***LabNumber_FirstName_LastName_AssignmentNumber_StudentNumber.pdf***, e.g. D**104_Jim_Silvester_Assignment1_1234567.pdf**
  (Please note failure to meet any of these submission requirements above would result in a penalty of -0.25 pt)
- **No late submission will be accepted**. If you do not complete the assignment by the deadline, you will receive **0**. For a legitimate reason a late submission may be considered pending discussion with your TA **before** the deadline. You may be required to provide supporting documents.

Phase 2(16 pts)

- The project must be named **in Eclipse** with the following format ***LabNumber_FirstName_LastName_AssignmentNumber _StudentNumber***, e.g. D**104_Jim_Silvester_Assignment1_1234567**
- Please note the **entire project** MUST be submitted (not just the source files)
- **To submit, export the project** (including **all the libraries used: i.e. Java System Library and Processing's core.jar file**) **into a zip file** (Archive File) and name it exactly **the same as the project name**
  (Please note failure to meet any of these submission requirements above would result in a penalty of -0.25 pt each)
- **No late submission will be accepted**. If you do not complete the assignment by the deadline, you will receive **0**. For a legitimate reason a late submission may be considered pending discussion with your TA **before** the deadline. You may be required to provide supporting documents.
- Make sure your code is syntax error free so that it runs properly on the lab machine. You would receive **0** for the coding part **if your code failed to run** due to syntax or runtime errors. In case there is discrepancy of running between grader's machine and your machine, the lab machine will be used as the standard one for examination.

You are graded on completeness and correctness.