Assignment 2a

COMP 2526 Object-Oriented Programming with Java

Milestone 1 due in lab during the week of February 8th -12th
Complete assignment due on Friday February 19th at 11:59 PM

1 Purpose

Use object oriented programming techniques to design and implement a solution that will be flexible and easy to modify in future iterations. You will apply what you have learned to a larger and more difficult program that we will develop over several assignments. After each submission, you will be provided with a sample solution that is flexible enough to continue into the next iteration.

2 Description

You are going to create a simulation of a simple world. A simulation like this is often referred to as the "Game of Life" (unrelated to the Parker Brothers board game).

The world simulation begins by placing Plants and Herbivores (plant eaters) on a two-dimensional grid of Cells. The grid displays the Plants (green) and Herbivores (yellow) by filling in the Cells where they are found. Blank Cells represent empty areas. Herbivores "graze" (feed) by moving around the grid eating Plants they find. Herbivores must find a Plant to eat before 5 "turns" have passed, or they die. A "turn" is a step in time which occurs when the user clicks anywhere on the window displaying the world. Herbivores move by looking for an empty neighbouring Cell (the Cell may have a Plant or it may not, but it may not have another Herbivore) and randomly picking one to enter. They move 1 Cell per turn. Plants do not move.

It is important that you consider future changes before building your simulation. Future changes might include any or all of or none of: movement patterns, reproduction, rules governing behaviour, physical representations, and more. By carefully thinking about what could be added or changed and designing with that flexibility in mind, you will find it easier to implement future iterations.

Some classes to consider:

1. Plant
   (a) Displayed as a green Cell
   (b) Cannot move
   (c) Initial placement is random
2. Herbivore
   (a) Displayed as a yellow Cell
   (b) Moves into "empty" Cells randomly (an empty cell is a cell that does not contain another Herbivore)
   (c) Eats Plants in its Cell, if any
   (d) Must find a plant to eat before 5 "turns" have passed or it dies
   (e) Must find a plant to eat within 4 "turns" after being born or it dies

    (f) Initial placement is random

3. Cell
    (a) Can hold a Plant or an Herbivore or nothing
    (b) Represented as a square in the world
4. World
    (a) Holds Cells
    (b) No wrap around on the world (it?s flat, but nothing falls off the edge either)
    (c) World has a 10% chance to create a Herbivore and a 30% chance to create a Plant each turn
5. Main
    (a) Drives the program
    (b) When user clicks on the window, a turn has passed
    (c) **This class is provided, and you must use it**
6. RandomGenerator
    (a) Produces a random value
    (b) **This class is provided, and you must use it**
7. GameFrame
    (a) Frame of program
    (b) **This class is provided, and you must use it**

3 Requirements

This assignment should demonstrate how expanding and maintaining a software system can be easier with a well planned design. You must adhere to these rules:

1. Import the zipped framework into Eclipse (File >Import >General >Existing projects into workspace >Next and choose the archive)
2. You may NOT modify the Main.java, RandomGenerator.java, or GameFrame.java files.
3. Cell must contain these methods:
    (a) init() sets up the layout
    (b) getLocation() returns the location of the Cell on the World
    (c) getAdjacentCells() returns the adjacent Cells – corners only return 3 Cells, sides only return 5, and all others return 8
4. World must contain these methods:
    (a) init() puts the Cells on the world and adds the appropriate number of Herbivores and Plants
    (b) getCellAt() retrieves the requested Cell from the specified location in the World
    (c) takeTurn() moves Herbivores one Cell (and they eat, if possible)
5. Plant must contain these methods:
    (a) init() sets the background to be green
    (b) setCell() puts the Plant on the specified Cell
6. Herbivore must contain these methods:
    (a) init() sets the background to be yellow
    (b) setCell() puts the Herbivore on the specified Cell
    (c) move() moves the Herbivore one cell (where it eats a Plant if the Cell contains a Plant)
7. Things to consider
    (a) a TurnListener class can call the takeTurn method in GameFrame and extends MouseAdapter (overrides mouseClicked)

   (b) Cell needs methods to set/get/remove Plants/Herbivores
   (c) Herbivore needs methods to figure out where Plants and empty cells are

8. **Milestone (10% of mark)**: In lab during the week of February 8th, your lab instructor will ask you to demonstrate your partially-implemented solution. You must demonstrate:
   (a) **The basic classes provided**
   (b) **Plants and Herbivores displayed in your GUI**
   (c) You are not required to demonstrate movement, eating, dying, or reproduction

It is likely you will want to add more classes and methods. Good luck, and have fun!

4 Marking Guidelines

60%	Functionality (does it work like the version provided)
20%	Good object oriented design
10%	Comments and style (follow your lab instructor's guidelines)
10%	Milestone week of February 8th (second lab)