Assignment 2a

COMP 2526 Object-Oriented Programming with Java

Milestone 1 due in lab during week ending 13 Oct 2017
Milestone 2 due in lab during week ending 20 Oct 2017
Completed assignment must be submitted to D2L by 23:59 PM on Sunday 5 Nov 2017

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 in COMP 1510 and COMP 2526 to a larger and more difficult program that we will develop over several assignments.

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 with the appropriate colour. 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 10 "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.

*** * * If there are Plants in any neighbouring Cells, the Herbivore will randomly choose one of the Cells that contain Plants to move to, otherwise if there are no neighbouring Plants the Herbivore will choose an empty Cell randomly picked. * * ***

Plants do not move. Plant however will "seed." If a Plant is surrounded by at least 3 empty neighbouring Cells and at least two neighbouring Plants to cross-pollinate, it will "seed", i.e., create a new Plant. The new Plant is "seeded" into a random neighbouring empty cell.

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, inhabitants, geographic disturbances, 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 you may want to consider when designing your solution:

1. Plant

   (a) Displayed as a green Cell
   (b) Cannot move
   (c) Will "seed" a neighbouring empty Cell with a new Plant if it has at least 2 neighbouring Plants to cross-pollinate and at least 3 empty neighbouring Cells from which to choose the new Plant's home Cell.
   (d) Initial placement is random
2. Herbivore
   (a) Displayed as a yellow Cell
   (b) Moves into "empty" adjacent Cells randomly (an empty cell is a cell that does not contain another Herbivore)
   (c) Eats the Plant in its Cell, if any
   (d) Must find a plant to eat before 10 "turns" have passed or it dies
   (e) 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) When creating a new World, each new Cell has a 20% chance of containing a Herbivore and a 30% chance of containing a Plant. Use the provided RandomGenerator class to generate a number in the range [0, 99]. If the value is greater than or equal to 80, generate an Herbivore, else if the value is greater than or equal to 50 generate a Plant.
5. TurnListener
   (a) extends MouseAdapter
   (b) implements the mouseClicked method (when the mouse is clicked, the GameFrame must take a turn)
6. Main
   (a) Drives the program
   (b) When user clicks on the game window, a turn passes
   (c) **This class is provided, and you must use it**
7. RandomGenerator
   (a) Generates random numbers from an array of int
   (b) **This class is provided, and you must use it**
8. 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. Create a new Eclipse Java project called COMP 2526 Game Of Life Part A
2. Create a single package: ca.bcit.comp2526.a2a
3. Copy the three furnished classes to the new package
4. Use Checkstyle

5. You may NOT modify the logic in Main.java, RandomGenerator.java, or GameFrame.java files, but you must comment the files and modify the code to satisfy Checkstyle.
6. We think you must have a Cell class. Cell must contain:
   (a) constructor that accepts three parameters: public Cell(World world, int row, int column)
   (b) public void init() sets up the layout
   (c) public Point getLocation() returns the location of the Cell on the World
   (d) public Cell[] getAdjacentCells() returns the adjacent Cells – corners only return 3 Cells, sides only return 5, and all others return 8
7. We think you must have a World class. World must contain these methods:
   (a) constructor that accepts two parameters: public World(int rows, int columns)
   (b) public void init() puts the Cells on the world and adds the appropriate number of Herbivores and Plants
   (c) public Cell getCellAt(int row, int column) retrieves the requested Cell from the specified location in the World
   (d) public void takeTurn() removes dead herbivores, checks each plant to see if it "seeds," and then moves remaining living Herbivores one Cell (and they eat, if possible)
8. You probably want a Plant class, too. Plant must contain these methods:
   (a) constructor that accepts one parameter: public Plant(Cell location)
   (b) public void init() sets the background to be green
   (c) public void setCell(Cell location) puts the Plant on the specified Cell
9. Of course we need an Herbivore. Herbivore must contain these methods:
   (a) constructor that accepts one parameter: public Herbivore(Cell location)
   (b) public void init() sets the background to be yellow
   (c) public void setCell(Cell location) puts the Herbivore on the specified Cell
   (d) public void move() moves the Herbivore one cell (where it eats a Plant if the Cell contains a Plant)
10. Things to consider
    (a) The TurnListener class extends MouseAdapter (overrides mouseClicked) and will call the takeTurn method in GameFrame and
    (b) Cell needs methods to set/get/remove Plants/Herbivores
    (c) Herbivore needs methods to figure out where Plants and empty cells are.

4 Milestone 1 (10% of mark): UML Design due during the week ending Friday October 13th

You must create and present your UML Design due in lab during the week ending Friday October 13th:

1. Work in pairs.
2. With your partner, derive your classes, their relationships with each other, and establish a basic interface (collection of important public methods) to your classes. A simple approach is to use Abbott's Heuristic for Natural Language Analysis:
   (a) Write a full description of the problem as you perceive it in as much detail as you can.
   (b) Generally we can map proper nouns to objects and common nouns to classes.
   (c) We can map doing verbs to methods, and being verbs (is-a) to classifications.
   (d) Having verbs suggest composition (has-a).
   (e) Adjectives tend to map to attributes (instance variables).
3. Each pair of students must present their design to the rest of the set who will help suggest improvements to the design.

4. The design must be presented as a UML diagram.
5. Pairs will have an opportunity to modify their design for marking. Note that each student will develop their own code for their joint design. Only the design stage is a team-project.

5 Milestone 2 (10% of mark): First code iteration due in lab during the week ending Friday October 20th

Your first code iteration is due in lab during the week ending Friday October 20th:

1. Work individually.
2. Begin to code your design. Your code must compile, and display Plants and Herbivores in the correct proportions in your GUI. No other requirements need to be met, i.e., no movements, eating, dying, "seeding," etc.
3. Submit a final copy of your UML design (as a 1-page PDF) to D2L before 11:59 PM on Friday October 20th.

6 Marking Guidelines

50%	Functionality (does it work)
20%	Good object oriented design (10% of mark based on UML diagram)
10%	Milestone 2 due week ending October 20th at 11:59:59 PM
20%	Comments and style (use Checkstyle)