Com S 228 Spring 2015

Project 1: Predator-Prey Simulation (100 pts)

Due at **11:59pm**

Wednesday, Feb 4

1. Problem Description

This project simulates interactions among different forms of life in a world. The world is represented by an $N \times N$ grid that changes over a number of cycles. Within a cycle, each square is occupied by one of the following five life forms:

Badger (B), Fox (F), Rabbit (R), Grass (G), and Empty (E)

An Empty square means that it is not occupied by any life form.

Below is a world example as a 6×6 grid.

```
G R R G B F
G R F F R F
E F F R G G
G G F B R R
R R E R E R
E F F G R F
```

Both row and column indices start from 0. In the example, the (1, 1)th square R (Rabbit) has a 3×3 neighborhood centered at the square:

```
G R R
G R F
E F F
```

The (0, 0)th square G (Grass) has only a 2×2 neighborhood:

```
G R
G R
```

Meanwhile, the (2, 0)th square E (Empty) has a 3×2 neighborhood:

```
G R
E F
G G
```

Generally, the "3×3" neighborhood of a square includes only those squares that lie within the intersection of the world with a 3×3 window centered at the square. When a square is on the border, the dimension of its neighborhood reduces to 2×3, 3×2, or 2×2.

2. Survival Rules

The world evolves from one cycle to the next one. In the next cycle, the life form residing on every square is decided from that in the *current cycle* as well as those in the 3×3 neighborhood centered at the square, according to a set of survival rules. Badgers, foxes, and rabbits start at age 0, and grow one year older when the next cycle starts.

2.1 Badger

A badger dies of old age or hunger, or from a group attack by foxes when it is alone. The life form on a Badger square in the next cycle will be

a) Empty if the Badger is currently at age 4;
b) otherwise, Fox, if there is only one Badger but there are more than one Fox in the neighborhood;
c) otherwise, Empty, if Badgers and Foxes together outnumber Rabbits in the neighborhood;
d) otherwise, Badger (the badger will live on).

The new life form taking over the square, if a Fox, will have age 0 when the next cycle starts. For example, in the following neighborhood of a Badger at age 1:

```
G R F
E B F
G G F
```

there are three Foxes. The central element (square) will become `F` (Fox) in the next cycle according to the rule b).

2.2 Fox

A fox dies of old age or hunger, or an attack by one or more badgers. The life form on a Fox square in the next cycle will be

a) Empty if the Fox is currently at age 5;
b) otherwise, Badger, if there are at least as many Badgers as Foxes in the neighborhood;
c) otherwise, Empty, if Badgers and Foxes together outnumber Rabbits in the neighborhood;
d) otherwise, Fox (the fox will live on).

The new life form, if a Badger, will have age 0 when the next cycle begins. For example, in the neighborhood

```
G R F
E F B
G R R
```

there are two Foxes, one Badger, and three Rabbits. There are more Foxes than Badgers. Meanwhile, Rabbits are not outnumbered. Suppose the Fox on the central square has age 3. It will be aged 4 in the next cycle.

2.3 Rabbit

A rabbit dies of old age or hunger. It may also be eaten by a badger or a fox. More specifically, the life form on a Rabbit square in the next cycle will be

a) Empty if the Rabbit's current age is 3;
b) otherwise, Empty if there is no Grass in the neighborhood (the rabbit needs food);
c) otherwise, Empty if there are at least as many Foxes and Badgers combined as Rabbits in the neighborhood;
d) otherwise, Rabbit (the rabbit will live on).

For example, if the neighborhood of a rabbit at age 3 is

```
R G G
F R R
F B R
```

the central element (square) will be `E` in the next cycle according to rule a).

2.4 Grass

Grass may be eaten out by rabbits. Rabbits may also multiply to take over the Grass square. In the next cycle, the life form on a Grass square will be

a) Empty if at least *twice as many* Rabbits as Grasses in the neighborhood;
b) otherwise, Rabbit if there are more Rabbits than Grasses in the neighborhood;
c) otherwise, Grass.

If the new life form is a Rabbit, it will be aged 0 when the next cycle starts. For example, if the neighborhood is

```
R G B
B G R
F E R
```

the central element (square) will be `R` in the next cycle.

2.5 Empty

Empty squares are competed by other life forms. More specifically, the life form on an Empty square in the next cycle will be

a) Rabbit, if more than one neighboring Rabbit;
b) otherwise, Fox, if more than one neighboring Fox;
c) otherwise, Badger, if more than one neighboring Badger;
d) otherwise, Grass, if at least one neighboring Grass;
e) otherwise, Empty.

If the new life form is a Badger, Fox, or Rabbit, it will have age 0 when the next cycle begins. For example, if an Empty square has neighborhood

```
R B
E F
G R
```

the central element (square) will be `R` in the next cycle.

3. Task

You will implement an abstract class `Living` to represent a generic life form. It has five subclasses `Badger`, `Empty`, `Fox`, `Grass`, and `Rabbit`. You also need to implement a `World` class which has a public member `Living[][]` to represent a grid world.

The class `PredatorPrey` repeatedly simulates evolutions of input worlds, either randomly generated or read from files. In each iteration, it interacts with the user who chooses how the world will be generated, and enters the number of cycles to simulate. Each iteration prints out the initial world and the final world.

Your random world generator `world` may follow the uniform probability distribution so that `Badger`, `Empty`, `Fox`, `Grass`, and `Rabbit` have equal likelihoods to occupy every square. Or you may use a different distribution, as long as no life form has zero chance to appear on a square.

World creation from an input file will weigh more than random creation in our grading. When zero or a negative number of cycles is entered by the user, your code does nothing but waits for a positive input.

A new living form, if a Badger, Fox, or Rabbit, has age 0 at its creation, whether initially by the class `world` or later on under a survival rule. After surviving a cycle, its age increases by one.

Templates are provided for all classes. Be sure to use the package name `edu.iastate.cs228.hw1` for the project.

Below is a sample simulation scenario over three initial worlds. In the first iteration, the user entered 1 for a randomly generated world, 3 to specify the grid to be 3 by 3, and 1 to simulate just one cycle. The simulator printed out the initial and final worlds. The second iteration simulated a randomly generated 6 by 6 grid over 8 cycles. In the third iteration, the user typed 2 for a file input, entered the file name "grid1.txt", and specified 10 cycles. After the iteration, the user typed 3 to end the simulation. (Any number other than 1 and 2 could have ended the simulation.)

```
The Predator-Prey Simulator
keys: 1 (random world)  2 (file input)  3 (exit)

Trial 1: 1
Random world
Enter grid width: 3
Enter the number of cycles: 1

Initial world:

E B G
B R F
R G F

Final world:

B E G
B E E
R E E

Trial 2: 1
Random world
Enter grid width: 6
Enter the number of cycles: 8

Initial world:

F E E F R R
R F F R R G
G G B E B R
G F G R E R
E G R R B G
B F E F R B

Final world:

G G R R E R
G F R R E R
G G R R R E
G E R E E R
G G R R E R
G G G R B E

Trial 3: 2
World input from a file
File name: grid1.txt
Enter the number of cycles: 10

Initial world:

F E R G E B E B R R
F R R F G R F G F F
F B B E E F F G E G
G G G G G R F G B F
R B R G R E B B B E
E R E E G B R G G B
E G F B B R F F B G
G G G R F R G F R F
R B E B G E F B R G
G F E F F G F G G B
```

```
Final world:

R R R G G G G E F E
G R R G G G G G F E
G G G G G G G G G G
G G G G G R G G B E
G E R R R E G G B E
G R E E R R G G G G
G G R R R G G G G G
G G R R R G F E G G
G G R R G G F E F G
G G F G G G G G G G

Trial 4: 3
```

Your code should print out the same text messages to engage user interactions.

4. Input/Output Format

The format for worlds is shown in the sample runs above. Note one space between adjacent characters ‘B’, 'E', 'F', 'G', or 'R'. No blank lines.

You may assume all input files to be correctly formatted, containing ‘B’, 'E', 'F', 'G', and 'R' as the only non-blank characters.

5. Junit Classes

JUnit classes include BadgerTest, EmptyTest, FoxTest, GrassTest, LivingTest, PredatorPreyTest, RabbitTest, and WorldTest. *You need to implement all these classes*.

6. Submission

Write your classes and JUnit test classes in the edu.iastate.cs228.hw1 package. Also submit the text files with your JUnit tests. Turn in the zip file not your class files. Please follow the guideline posted under Documents & Links on Blackboard Learn.

Include the Javadoc tag @author in each class source file. Your zip file should be named Firstname_Lastname_HW1.zip.