Predator-Prey Simulation (100pts)
Due Thursday Mar 31, 2016, 11:59pm

Important Notes

- Submission location: /rd/cs246s2016/${USER}/project5
- **This assignment is to be done on your own.** If you need help, see the instructor or TA.
- Please start the assignment as soon as possible and get your questions answered early.
- Read through this specification completely before you start.
- Some aspects of this specification are subject to change, in response to issues detected by students or the course staff.

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.

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

Both row and column indices start from 0. In the example, the (1, 2)th square `R` (Rabbit) has a 3×3 neighborhood centered at the square:

E F G
F R E
E F G

The (0, 0)th square `R` (Rabbit) has only a 2×2 neighborhood:

R E
E F

Meanwhile, the (2, 0)th square `B` (Badger) has a 3×2 neighborhood:

E F
B E
F 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 some survival rules. Badgers, Foxes, and Rabbits start at age 0, and grow one year older when the next cycle starts. For every life form, a set of rule applies.

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

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

For example, in the neighborhood

G R F
E B F
G G F

there are three Foxes but only one Badger. The central element (square) will become `E` (Empty) in the next cycle according to the rule 2.

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

1. Empty if the Fox is currently at age 5;
2. otherwise, Empty, if there are at least as many Badgers as Foxes in the neighborhood;
3. otherwise, Empty, if Badgers and Foxes together outnumber Rabbits in the neighborhood;
4. otherwise, Fox (the Fox will live on).

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. Also, Rabbits are not outnumbered. Suppose the Fox on the central square is at age 3. It will stay on in the next cycle.

2.3 Rabbit

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

1. Empty if the Rabbit's current age is 3;
2. otherwise, Empty if there are at least as many Foxes and Badgers together as Rabbits in the neighborhood;
3. otherwise, Empty if there is no Grass in the neighborhood (the Rabbit needs food);
4. otherwise, Rabbit (the Rabbit will live on).

For example, if the neighborhood is

R G G
F R R
F B R

the central element (square) will be `R` in the next cycle. (Here we assume that this Rabbit is under age 3.)

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

1. Rabbit if more than one Rabbit and more than one Grass in the neighborhood;
2. otherwise, Grass if more Grasses than Rabbits in the neighborhood;
3. otherwise, Empty.

For example, if the neighborhood is

G 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 replaced by other life forms. More specifically, the life form on an Empty square in the next cycle will be

1. Badger, if more than one Badger and no more Foxes than Badgers in the neighborhood;
2. otherwise, Fox, if more than one Fox and more Foxes than Badgers in the neighborhood;
3. otherwise, Rabbit, if the number of Rabbits exceeds one in the neighborhood;
4. otherwise, Grass, if at least one neighboring Grass;
5. otherwise, Empty.

3 Task

Your program must do the following:

- Randomly generate a predator-prey simulation system based on the given width of the grid.
- Update the system each cycle based on the survival rules.
- Use conditional compilation (e.g., *#if* and *#endif* directives) for predefined test cases. Your code should print out the same text messages as shown in the sample runs given in Section 4.
- Provide at least the following functions, but probably more:
  - A function that creates a random predator-prey system.
  - A function that finds the neighborhood of a given cell.
  - A function that updates a given cell.
  - A function that counts the number of life forms of a specific type in the neighborhood of a given cell.
  - A function that increases the ages of animals on the grid once each cycle.
  - A function that accepts input for creating a random predator-prey system.
  - A function that prints out the predator-prey system.

  Pay special attention to designing your functions and your program organization. Note that your project will be graded on its organization and modularity as well as functionality.
- Provide a prototype for every function in your program in a header file, which you then include in your source file using the *#include* directive. The file should be named the same as your program but with extension .h. Submit it together with your source code.

In addition, you should have the following in your submission:

- a Makefile called *Makefile* that has targets for building the program as well as each test case.
- a README describing how to make a run the program.
- All of your files should be committed to the mercurial repository in your course directory.

4 Sample Runs and Test Cases

Below you will find four sample runs. Use preprocessor conditional compilation for running at least three of these test cases, and provide a single Makefile with a target for each version of the test. Your Makefile should be called "Makefile". Your code should print out the same text messages as shown.

- A sample run with 3×3 grid and 1 cycle:

```
The Predator-Prey Simulator

Grid width: 3; cycle(s): 1
Initial World
R R F
G F R
G R B

Final World:
R R F
R F E
G E B
```

- A sample run with 6×6 grid and 10 cycles:

```
The Predator-Prey Simulator

Grid width: 6; cycle(s): 10
Initial World
R E F G G R
E F R E G R
B E F G B F
F G G F F E
R B B B E F
G F E E R R

Final World:
E E G G G E
F F F G F F
G E G G F E
G G G G F E
G G B B E E
G G E E F F
```

- A sample run with 10×10 grid and 7 cycles:

```
The Predator-Prey Simulator

Grid width: 10; cycle(s): 7
Initial World
B E E E E E G G G G
E E E F F F G G E G
E E E E E E G B G G
E G B E E E E E B E
E E F G E G E B B B
E R E R E F E R E G
R R R E E G F E R E
R R R R B E E F R E
B R F R R F F E E E
E E R R R E F F E E

Final World:
R R F F F F G G G G
R R E E E E G G G G
E E F E F F G E G G
R R E F F E E B E B
E E R E F F B E E B
R R E F F F E E E E
R E E E E F E F F F
E R F E E E F E E E
F E F F E E E F F F
F E F E F F E E E E
```

- A sample that accepts width (4) and cycle (6) from the user input, randomly generates the system and runs:

```
The Predator-Prey Simulator

Enter the width of the grid: 4
Enter the number of cycles: 6
B R E G
R E E F
G E F G
E F F G

Final World:
F F G G
E E E G
G F F G
G G G G
```