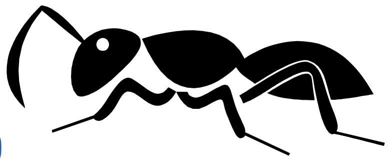
# C490 Homework #3

| | |
|---|---|
| Points: | : 40 points |
| Due Date: | : July 18th (8:30am) |
| Submissions: | : Canvas and hardcopy |



## PART I: THE WORLD OF ANTS (10 POINTS)

The goal for this programming project is to create a simple 2D predator–prey simulation. In this simulation, the prey is ants, and the predators are anteaters. These critters live in a world composed of a 20 X 20 grid of cells. Only one critter may occupy a cell at a time. The grid is enclosed, so a critter is not allowed to move off the edges of the grid. Time is simulated in time steps. Each critter performs some action every time step.

The ants behave according to the following model:

- Move. Every time step, randomly try to move up, down, left, or right. If the cell in the selected direction is occupied or would move the ant off the grid, then the ant stays in the current cell.
- Breed. If an ant survives for three time steps, then at the end of the third time step (i. e., after moving), the ant will breed. This is simulated by creating a new ant in an adjacent (up, down, left, or right) cell that is empty. If there is no empty cell available, no breeding occurs. Once an offspring is produced, the ant cannot produce an offspring until three more time steps have elapsed.

The anteaters behave according to the following model:

- Move. Every time step, if there is an adjacent cell (up, down, left, or right) occupied by an ant, then the anteater will move to that cell and eat the ant. Otherwise, the anteater moves according to the same rules as the ant. Note that an anteater cannot eat other anteaters.
- Breed. If an anteater survives for **eight** time steps, then at the end of the time step, it will spawn off a new anteater in the same manner as the ant.
- Starve. If an anteater has not eaten an ant within the last **three** time steps, then at the end of the third time step, it will starve and die. The anteater should then be removed from the grid of cells.

During one turn, all the anteaters should move before the ants.

Our goal is to develop a program to implement this simulation and draw the world using ASCII characters of “o” for an ant and “X” for an anteater. Now **an incomplete version of the program is given** (WorldOfAntsSim.java, in Canvas), you just need to complete it. The incomplete version contains five classes. The *Organism* class encapsulates basic data common to both ants and anteaters. This class has several abstract methods that are to be defined in the derived classes of *Ant* and *Anteater*, which simulate ants and anteaters respectively. The *World* class stores data about the world by creating a WORLDSIZE by WORLDSIZE array of type Organism. And the *WorldOfAntsSim* class utilizes the above three classes to create a simulation with a world initialized with 5 anteaters and 100 ants. After each time step, the simulation will prompt the user to press Enter to move to the next time step. You should see a cyclical pattern between the population of predators and prey, although random perturbations may lead to the elimination of one or both species.

The code for classes *Organism, Ant, World*, and *WorldOfAntSim* are given. First, you need to read the code to understand it. Then you need to implement the ***Anteater*** class, which should contain at least the following:

- Constants for ticks to breed and starve
- Two constructors
- Override the fours abstract methods in class Organism, namely, breed(), move(), starve(), and getPrintableChar(). Note that the behave model for anteaters is different from that for ants!

## PART II (10 POINTS)

Write a program that calculates the average of N integers. The program should prompt the user to enter the value for N and then afterward must enter all N numbers. If the user enters a non-positive value for N, then an exception should be thrown (and caught) with the message “N must be greater than 0.” You should have a loop in your program so that the user will enter a value for N again until a positive value is entered. If there is any exception as the user is entering the N numbers (**hint:** for this case, you do not have to **throw** an exception explicitly. Question: who might throw an exception then?), an error message should be displayed and the user prompted to enter the number again. Your program should behave like the binary code (***CalculateAverage.class***) posted in Canvas.

**Requirements:**

(1) Use try-throw-catch mechanism
(2) Wisely use loops, so that anytime when an exception is caught, the user has a chance to enter the number again.

## PART III (10 POINTS)

Use the PanelDemo.java in Canvas as the starting point, modify the program so that it behaves like the binary code (***ModifiedPanelDemo.class***) posted in Canvas.

## PART IV (10 POINTS)

Write a program that need to know how to convert integers from base ten (ordinary decimal) notation to base two notation. Use Swing to perform input and output via a window interface. The user enters a base ten numeral in one text field and clicks a button with "Convert" written on it; the equivalent base two numeral then appears in another text field. Be sure to label the two text fields. Include a "Clear" button that clears both text fields when clicked. Your program should behave like the binary code (***NumberConverter.class***) posted in Canvas.

**Hints:** include a private method that converts the string for a base ten numerals to the string for the equivalent base two numeral, an example of such method is given as below:

```
private String baseTwo(String inputNumber)
{
      int input = Integer.parseInt(inputNumber);
      if ( input == 0 )
      {
            return "" + 0;
      } // end of if ()

      if ( input < 0 )
      {
            return "error";
      } // end of if ()

      String result = "";
      while ( input != 0 )
      {
            result = "" + input % 2 + result;
            input /= 2;
      } // end of while ()
      return result;
}
```

## WHAT TO SUBMIT:

- Submit your code to Canvas (using the "Assignments" function).
- Submit a hard copy of your code and test-run output (or screenshot).
- Make sure that you follow the "Assignment_style-guideline_C490" or you might lose credits.