4CCS1PPA Programming Practice and Applications

Coursework 3: Predator/Prey Simulation

Michael Kölling, Josh Murphy, Jeffery Raphael
Questions? programming@kcl.ac.uk

Your task is to extend a predator/prey simulation. You should use the foxes-and-rabbits-handout project (available for download from KEATS) as a basis for your own project, and modify and extend it to make it more interesting. Note that this is slightly different from the book project, so please use the version from KEATS, not from the book projects.

You **must** replace the Fox and Rabbit classes with different kinds of predator and prey to simulate a different habitat (for example, under water, in a jungle, or in a fantasy world). You may add additional classes to those included in the existing project.

This project is a pair programming task. You must work in pairs. Information about pair programming is provided separately. We will not accept submissions by individuals.

---

1 Core tasks

You should aim at completing all core tasks. They are as follows.

- Your simulation should have at least five different kinds of acting species. At least two of these should be predators (they eat another species), and at least two of them should not be predators (they may eat plants). Plants can either be assumed to always be available (as in the original project), or they can be simulated (see below).
- At least two predators should compete for the same food source.
- Some or all of the species should distinguish male and female individuals. For these, the creatures can only propagate when a male and female individual meet. ("Meet" means they need to be within a specified distance to each other, for example in a neighbouring cell.) You will need to experiment with the parameters for breeding probability to create a stable population.
- You should keep track of the time of day. At least some creatures should exhibit different behaviour at some time of the day (for example: they may sleep at night and not move during that time).

You should implement the core tasks first before you move on to the extension tasks.

2 Challenge tasks

Once you have finished the core tasks, implement one or more extension tasks. You can either choose from the following suggestions, or invent your own. You will be graded on a maximum of four extensions.

- Add plants. Plants grow at a given rate, but they do not move. Some creatures eat plants. They will die if they do not find their food plant.
- Add weather. Weather can change, and it influences the behaviour of some simulated aspects. For example, grass may not grow without rain, or predators cannot see well in fog.
- Add disease. Some animals are occasionally infected. Infection can spread to other animals when they meet.

If you invent your own extension tasks, check with your class supervisor before implementing them. You can get their comments to ensure they your idea is not too simple or too difficult.

To get full marks on this assignment you must demonstrate exceptional technical aptitude with the challenge tasks you implement.

3 Submission and Deadline

The submission consists of two parts: your code and a report documenting it. The code and report must be submitted before the deadline, by **both members of your pair**. Both the source code and the report should clearly state the names of both authors.

The Code

You have to submit a Jar of your project to the "Assignment 3: Code Submission" link in the assignment 3 section on the PPA KEATS page, before the due date. **The Jar file must contain your source code, i.e., the *.java files.** Your code will be assessed for:

1. Program Correctness —The application meets all of the program specifications, i.e., the student has completed all of the base tasks including following submission instructions (e.g., the student submitted a Jar file of their BlueJ project).
2. Code Elegance —The application is written in such a way that the code is reusable and efficient (i.e., memory usage and complexity). The application appropriately uses loops and functions to reduce code complexity and/or repeated code. The application does not have hard-coded solutions or poorly designed solutions. A poorly designed solution is overly complicated, utilises excessive amounts of memory or utilises a slower approach to a problem.
3. Documentation —The application is sufficiently documented. Good documentation/comments should explain what the code does and how it does it. Comments can also be used to highlight nuances in your solution, e.g., a segment of code that only works under certain conditions.
4. Readability —The application is easy to understand and uses good programming practices.

The report

The report must be submitted to the "Assignment 3: Report Submission" link in the assignment 3 section on the PPA KEATS page, before the deadline. The report should be no more than four pages long. Marking the assignment will involve an interview in your lab class. More detail about the interview will be provided separately. The report must clearly state the names and student numbers of all students who worked on the submission. The report should also contain the following :

- A description of your simulation, including the types of species that you are simulating, their behaviour and interactions.
- A list and description of all extension tasks you have implemented.
- Code limitations/problems, e.g., code segment that uses a hard coded solution.

The report will be marked out of 100. Additional details on how marks are given can be found in the Marking Rubric. Once your submission is marked and written feedback has been returned to you, you will have the opportunity to get additional verbal feedback on your submission from your lab TAs.

4 Deadline

After submitting your work on KEATS, check that (1) your Jar file and report (pdf) has been successfully uploaded and (2) that the Jar file contains all of your source files. This assignment (code and report) is due **Sunday 23rd February, 23:59**.