Fundamentals of Programming

COMP1005 Assignment
Swamp Of Life

Semester 2, 2022 v1.0

Discipline of Computing
Curtin University

1 Preamble

In practicals you have implemented and learned about simulations, object-orientation and (soon) how to automate the running of multiple simulations. In this assignment, you will be making use of this knowledge to extend a given simulation to provide more functionality, complexity and allow automation. You will then report on the results generated by the simulation.

2 The Problem

We will be simulating the movement of creatures/beings in an ecosystem – the Swamp Of Life. The animals should include: **ducks, newts and shrimp**. Each creature will have independent behaviour and interaction rules that are impacted by the state of the creature and changes in "mode" for the simulation. There should also be non-moving objects, such as food sources.

You will be given some sample code, showing a range of approaches to similar problems. For the assignment, you will develop code to model the beings using objects, and to add features to the simulation (e.g. more functionality, statistics, graphics). Your task is to extend the code and then showcase your simulation, varying the input parameters, to show how they impact the overall simulation. **Note:** you can also write your own code from scratch

The required extensions are:

1. **Movement:** Movement should change based on the age of the creature and use a combination of reasoning and randomness for each move. You should provide the option of Moore or Von Neumann neighbourhoods.
   Prompts: *How do they move towards or away from targets/threats? Are they sometimes faster or slower? Are they always active, or does time have an impact?*
2. **Boundaries and Terrain:** You should be able to read in the layout of the area from a file, including boundaries to stop the beings going beyond the grid, and limiting movement within the grid (walls). This should include some representation of land and water, and the height/depth of the terrain.
   Prompts: *How can beings move between cells taking barriers into account? How do you stop them moving to invalid spaces? How might the beings negotiate height?*
3. **Interaction:** Your code should recognise when beings are in the same/neighbouring cells and respond accordingly. It could be an interaction when one block away, or

require beings to be in the same cell – depending on the scenarios.
Prompts: *How do you recognise a target/threat is near? Add rules as to what happens when different combinations of beings meet.*

4. **Life-cycles:** Your creatures should go through life-cycles, common to all creatures in that class. This will include reproduction and death
   Prompts: *How do you trigger/time a change of state for the phases of development? Does death take the creature out of the simulation, or do they impact the environment? How realistic are your rules for reproduction?*
5. **Non-moving objects:** You should model special cells/zones and their effect on the beings. Beyond terrain, you will need a way to indicate that a cell has a "landmark" for them to interact with e.g. food.
   Prompts: *How do you recognise the location of the non-moving objects? Can your beings "see" or "smell" and move accordingly?*
6. **Visualisation/Results:** Enhance the display for the simulation to vary the representation of beings and the area they move in. You should display a log of events and/or statistics for the simulation and also be able to save the simulation state as a plot image or a csv file.
   Prompts: How will you differentiate the beings? Are they the same throughout the simulation? You may choose an "image" representation, to improve on using coloured dots.

Note: Your program should allow command line arguments to control the parameters of the experiment/simulation.

Your code should include comments to explain what each section does and how. It is useful to keep track of your changes in the comments at the top of the program. Feel free to re-use the code and approaches from the lectures and practicals. **However, remember to cite and self-cite your sources.** If you submit work that you have already submitted for a previous assessment (in this unit or any other) you have to specifically state this.

Beyond the working program, you will submit a documents: the Simulation Project Report. There will be **bonus marks** for additional functionality and the use of more advanced programming techniques (e.g. interactivity, high quality visualisation, 3D space, parameter sweep etc.) but only if they're sensible and done well. Make sure to discuss the additional work in your Report, this will be easy if you make notes and keep old (incremental) versions of your code.

Remember : Think before you code!

3 Submission

Submit electronically via Blackboard. You can submit multiple times – we will only mark the last attempt. This can save you from disasters! Take care not to submit your last version late though. Read the submission instructions very carefully.

You should submit a single file, which should be zipped (.zip). Check that you can decompress it on lab machines. These are also the computers that your work will be tested on, so make sure that your work runs there. The file must be named FOP_Assignment_<id> where the <id> is replaced by your student id. There should be no spaces in the file name; use underscores as shown.

The file must contain the following:

- Your **code**. This means all files needed to run your program. That includes input files used as part of the assignment if that is required to run your program.
- **README** file including short descriptions of all files and dependencies, and information on how to run the program.
- **Report** for your code, as described in Section 3.1.
- A signed and dated **cover sheet**. These are available on Blackboard. You can sign a hard copy and scan it in or you can fill in a soft copy and digitally sign it.
- **You will also need to submit the Report to TurnItIn.**

Make sure that your zip file contains what is required. Anything not included in your submission will not be marked, even if you attempt to provide it later. It is your responsibility to make sure that your submission is complete and correct.

3.1 Simulation Project Report

You need to submit your Report in Word doc or pdf format. You will need to describe how you approached the implementation of the simulation, and explain to users how to run the program. You will then showcase the application(s) you have developed, and use them to explore the simulation outputs. This exploration would include changing parameters, simulation time and perhaps comparing outcomes if you switch various features on/off.

THE REPORT MUST BE SUBMITTED THROUGH TURNITIN AND IN THE ZIP FILE

Your **Documentation** will be around 6-10 pages and should include the following:

- **Overview** of your program's purpose and features. A discussion of your code, explaining how it works, any additional features and how you implemented them.
- **User Guide** on how to use your simulation (and parameter sweep code, if applicable)
- **Traceability Matrix** of features, implementation and testing of your code
- **Showcase** of your code output, including:
  - **Introduction:** Explain the features, modifications and parameters you are showcasing
  - **Results:** Present the results of at least three different simulation scenarios.
    - **Methodology:** Describe how you have chosen to set up and compare the simulations for the showcase. Include commands, input files – anything needed to reproduce your results.
    - **Discussion:** Show and discuss the outputs/results of each scenario.
  - **Conclusion and Future Work:** Give conclusions and what further investigations and/or model extensions could follow.

3.2 Marking

Marks will be awarded to your submission as follows:

- **[30 marks] Code Features.** Based on your implementation and documentation
- **[30 marks] Demonstration.** Students will demonstrate their code and respond to questions from the markers. Each feature should be demonstrated, and we will also assess the usability and flexibility of the code.
- **[40 marks] Report.** As described in section 3.1.

Marks will be deducted for not following specifications outlined in this document, which includes incorrect submission format and content.

3.3 Requirements for passing the unit

Please note: As specified in the unit outline, it is necessary to have attempted the assignment in order to pass the unit. As a guide, your assignment must score at least 15% (before penalties) to be considered to have attempted this assignment. We have given you the exact mark breakdown in Section 3.2. Note that the marks indicated in this section represent maximums, achieved only if you completely satisfy the requirements of the relevant section.

Plagiarism is a serious offence. This assignment has many correct solutions so plagiarism will be easy for us to detect (and we will). For information about plagiarism, please refer to http://academicintegrity.curtin.edu.au.

In the case of doubt, you may be asked to explain your code and the reason for choices that you have made as part of coding to the unit coordinator. A failure to adequately display knowledge required to have produced the code will most likely result in being formally accused of cheating.

Finally, be sure to secure your code. If someone else gets access to your code for any reason (including because you left it on a lab machine, lost a USB drive containing the code or put it on a public repository) you will be held partially responsible for any plagiarism that results.

3.4 Late Submission

As specified in the unit outline, you must submit the assignment on the due date. If there are reasons you cannot submit on time, you should apply formally for an Assessment Extension.

Students with a Curtin Access Plan should include a submission note to indicate the extra time they have taken, ensuring they have submitted the CAP to Blackboard for us to check.

In the event that you submit your assignment late, you will be penalised based on the number of days it is late. Note that if you are granted an extension you will be able to submit your work up to the extended time without penalty – this is different from submitting late.

3.5 Clarifications and Amendments

This assignment specification may be clarified and/or amended at any time. Such clarifications and amendments will be announced in the lecture and on the unit's Blackboard page. These clarifications and amendments form part of the assignment specification and may include things that affect mark allocations or specific tasks. It is your responsibility to be aware of these, either by attending the lectures, watching the iLecture and/or monitoring the Blackboard page.