[[1]](#footnote-1)

**Aim of the Project:**

The project simulates an ecosystem consisting of lions and antelopes. Every animal takes up a space on a grid and what happens after every simulation is determined by a range of rules.

This is then represented graphically

**Basic possibilities of a cell:**

A cell can either be an empty space, a baby lion or antelope or an adult lion or antelope

**Breeding Ages and other constants:**

By default these are the values stipulated in the question but the user can change it.

\*Note: When we talk about surrounding a cell, we are talking about the 8 cells that are adjacent or 1 diagonal of the cell.

**Walkthrough of programme:**

1. The programme starts by allowing you to continue with the programme as it was in Part2 but now the new option of playing a previously recorded file is there
2. Trying the new option (which can only happen if there is a previously recorded file- which we can do later in the programme), shows us a grid similar to the main simulation except there is no difference between adults and children in terms of display but the choice of colour change is there
3. New to this display as opposed to the one in the main simulation is the ability to use the slider to choose any generation that was recorded in the file.
4. We can then go to the start screen and choose to go through the old options.
5. The grid is then randomly filled based on the probability ratio of empty spaces to lions to antelopes, the user chooses this ratio.
6. The user can then change the default constants for adulthood for lions and antelopes (also known as the breeding age), the maximum age for lions and antelope and the probability of a natural death for a lion.
7. If a block is filled with an animal, a random age in the range of 1 to the maximum age is generated for that animal. If this is below the adult or breeding age, it is classified as a baby, otherwise it’s an adult.
8. Baby antelopes are represented by an ‘a’, adults by an ‘A’, baby lions by a ‘l’, adults by a ‘L’ and an empty space by an ‘e’.
9. Each cell contains information about those around it, in terms of adults and total lions and antelopes surrounding it.
10. The grid is then outputted using GUI made possible by Tkinter
11. The user can then choose from the following options:
    1. Change colour assignments (Each of the 5 states of a cell)
    2. Simulate one generation
    3. Simulate continuously
    4. Pause this simulation
    5. Reset simulation
    6. Change constants again
    7. Click on a Cell and find out details about it

(A simulation is done using rules discussed below in ‘[What happens in one Simulation?](#W)’**)**

1. The user stays on the screen from which all features in step 8 are available until the user presses the close button.
2. New Feature: In the screen preceding the main simulation the user can choose to record the simulation to a file of their choice as well as stipulate the number of generations to record for.
3. If the user goes to the main simulation after recording something, the grid on display is the same one as in the start of the file.

**What happens in one Simulation?**

First it is checked what is contained in a cell

\*Note the following list is represented in order of preference, meaning if more than one condition is fulfilled; the one higher in the list is the one that happens.

If a cell is empty:

* If there are at least 4 neighbouring lions and at least 3 of these are adults and there are at most 3 neighbouring antelopes, a baby lion of age 1 is born into that cell.
* If there are at least 4 neighbouring antelopes and at least 3 of these are adults and there are at most 3 neighbouring lions, a baby antelope of age 1 is born into that cell.

If the cell has a lion:

* If it is of the maximum age for a lion, it dies
* If it is surrounded by at least 6 lions and no antelopes, its dies of starvation
* Unless turned of by the user, a random number in the natural death probability range is generated and it determines if lion dies of natural causes.

If the cell contains an antelope:

* If it of the maximum age for an antelope, it dies
* If it surrounded by 5 or more lions, it is eaten.
* If it is surrounded by 8 antelope, it dies of starvation

All cells that still contain an animal (Except new-borns) get one year older. If the animal is now of the adult age for its species; it becomes an adult.

**What makes this programme special?**

* The programme provides a user-friendly GUI and help is available in each screen
* The programme allows user to change all constants, even during simulations
* The programme allows colour changes for the grid
* After each simulation statistics are updated. Statistics are both helpful and relevant
* The programme treats adults and children separately
* The programme is efficient in terms of speed and memory used and it at good speeds
* Programme can be restarted with different constants
* Programme does not freeze, lag or crash
* Very User-friendly and easy to understand
* Programme is done entirely in GUI, making use of many features including font size and colour, sliders, buttons and canvases
* Appropriate error pop-ups for bad inputs
* Natural deaths can be turned off
* Probability ratios can be changed
* You can click on the grid and that cell will turn white and information about that cell will appear on a pop-up
* Input ranges are also restricted to values that are viable to simulate
* Based on grid size inputs, the grid makes use of square sizes that are of maximum size to fill as much as possible (eg 100 x 100 grid contains very small squares compared to a 10 x 10 grid but they occupy the same total space).
* See [What is new to Part3 of the Project?](#W)

**Technical aspects of the programme:**

* Input ranges are restricted to values that are viable to simulate
* The programme makes use of lots of exception handling and input validation to ensure that inputs make sense to the programme (eg. In the probability screen all values can’t be 0)
* The programme makes use of python’s built in random, math and copy functions
* Code that is used multiple times are put in methods, the best example of this is the fill method which is used throughout the programmes because it updates surrounding cells whenever something happens
* Comments are used to easily understand what each piece of code does
* Many if statements because the programme has the feature of representing adults and children differently
* Formatting of output is used to make output neat and readable
* The programme makes use of two classes, Project3Info.py stores information about the animals in the grid and Project3GUI.py
* The compression was checked manually on smaller cases to ensure that information is not corrupted during the compression and reading process.

**What is new to Part3 of the Project?**

* The user can record simulations to a file to be played back later. The information is stored in the file with the header stating the grid size and number of simulations recorded. To save space and time, the file makes use of simple yet powerful run-length compression of each frame. After the header, each line specifies the corresponding generation starting from 1 (So 5th line stores info about 4th frame). A letter followed by a number specifies the number of that block in a row eg. E6 A4 means that there are 6 empty spaces followed by 4 antelopes. The grid is treated as if it was one long row and this is converted back to the grid when it is being displayed.
* The user can play this recording. The best feature here is the slider which lets the user choose any generation and it is immediately generated; no long wait for everything to be simulated.
* There are a few more screens to accommodate the new features so there are more help buttons to explain them.
* No features of Part2 were recorded to accommodate these new features.

**Known Errors:**

The programme has been extensively tested especially with corner cases and no errors have yet been found.

1. Yes it’s there again. It’s what makes my project unique [↑](#footnote-ref-1)