Notre Dame University – Louaize
Faculty of Natural and Applied Sciences
Department of Computer Science

Forest Simulation

CSC 323 – Object Oriented Design
Project

Submission Guidelines

The outcome of your project should be the following:

- 1 StarUML file
- Visual Studio solution and project with all the code files included

NB: Compress all the files into one Zip file and submit it through Blackboard by the assigned deadline. The name of the Zip file should be: YourName(s)_Project.
Assignments submitted through other means (email, hard copy, CD, etc.) will not be considered.

1. General Requirements

You are required to produce a **UML class diagram**, which represents a forest simulation software. You are also required to implement this software application using **C#**.

The forest is represented by a flat (2D) rectangle, as shown in Figure 1. The size (height and width) of the rectangle should be specified by the user. The forest shall have a light green background color. The space inside the forest is illustrated as small squares (e.g., 10 × 10). The color of these squares shall be a slightly darker shade of green. These squares represent empty space, and are separated from one another by a small margin.

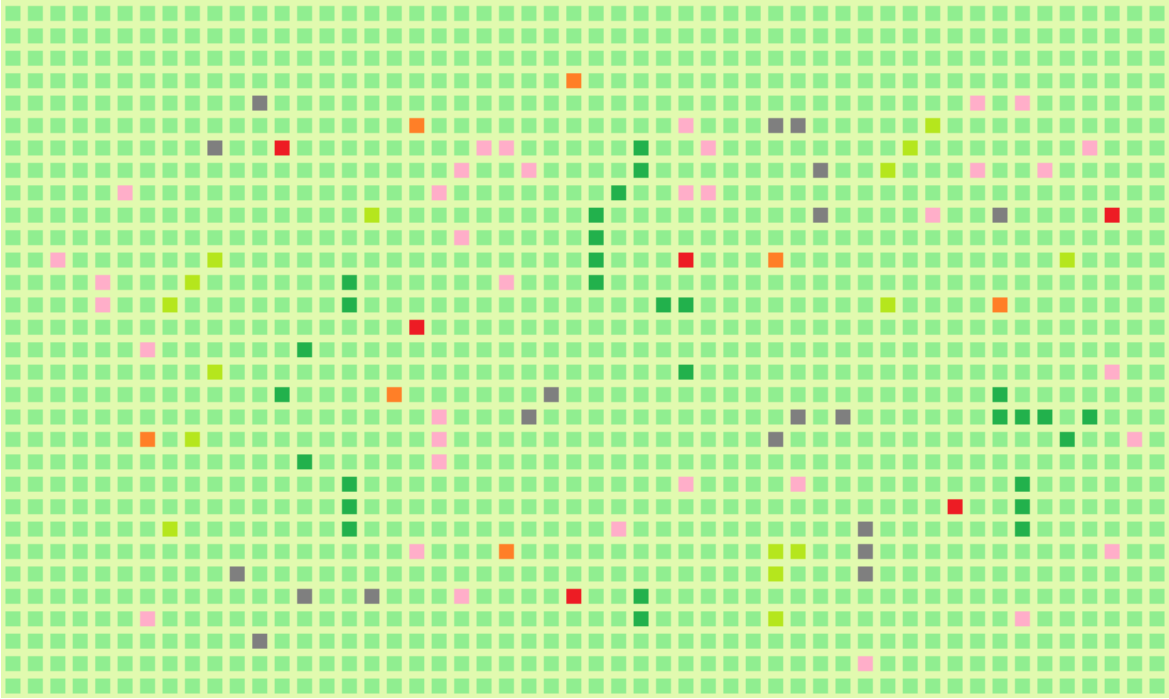


Figure 1. Rendered Forest Simulation

2. Forest Elements

Different elements can be allocated to the empty spaces inside the forest. Each of these elements is represented by a small colored square.

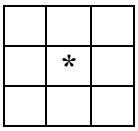
The moving animals inside the forest are **elephants**, **lions**, and **deer**. There are also stagnant (non-moving) things including: **rocks**, **trees**, and **plants**.

Upon initializing the forest, you are required to randomly allocate to the empty spaces a number of elephants, lions, deer, rocks, trees, and plants. These numbers are specified by the user.

3. Iterations

The simulation passes through several iterations. Each iteration runs after a certain period of time (e.g., a few seconds or milliseconds). This period of time is specified by the user.

In each iteration, elephants, lions, and deer randomly move to any of the eight neighboring cells (spaces).



The rules of the movement within the forest are presented in Table 1. When it is not possible to move to a certain location you have to randomly choose one of the other spaces. If it is not possible to move to any of the eight space, then the element stays in its place.

Table 1. Movement Matrix

	Space	**Rock**	**Tree**	**Plant**	**Elephant**	**Lion**	**Deer**
Elephant	Occupy	X	Eats	Destroys	X	Kills	Kills
Lion	Occupy	X	X	Destroys	X	X	Eats
Deer	Occupy	X	X	Eats	X	X	X

* *"X" means that it is not possible to move to this location*
* *Eats, Destroys, and Kills means that the new element will take the place of the old one.*

4. Gestation, Starvation, and Lifetime Periods

Animals (elephants, lions, and deer) have **lifetimes** after which they would die, and hence would no longer exist in the forest. The lifetime period indicates the number of iterations that the animal can pass through before dying of natural causes.

Female animals also have different **gestation periods** (number of iterations). When the end of a gestation period is reached, each female animal will give birth to a single newborn animal that is randomly set to male or female. The newborn animals shall also be randomly placed in one of the eight neighboring cells. If none of these cells are empty, the newborn would die. Newborn females can start reproducing after five iterations.

All animals have a **starvation period**, which indicates the number of iterations that the animal can pass through without eating before it dies.

In case an animal dies, it shall be replaced by an empty space.

Lifetimes, gestation periods, and starvation periods shall be specified by the user. Different values are required for elephants, lions, and deer. However, values are common within the species (e.g., all elephants could have a lifetime of 50 and all lions could have a lifetime of 30).

5. Controlling the Simulation

The user should have the ability to pause, stop, and restart a simulation. Additionally, you should provide a settings form, which allows the user to specify all the user-defined parameters mentioned in the previous sections.

6. Some Quality Guidelines

- Mind the quality of your code (e.g., readability, efficiency, etc.)
- Separate your business logic from the presentation in a way that makes non-UI related class unaware about the UI
- Follow the design principles, patterns, and best practices that are covered in this course