CPSC 428/528 – Artificial Intelligence
East Stroudsburg University of Pennsylvania
Spring 2025

Programming Assignment 2

Release date: Feb 27, 2025
Due Date: April 3, 2025
Max. Points: 100
Bonus Points: 35

Weightage: This assignment is worth 15% of your course work (programming assignments)

Assignment Description

Title: Studying Artificial Life using the Conway's Game of Life Implementation in C# Unity

Objective: Implement Conway's Game of Life using C# and Unity, exploring the properties of a cellular automaton on an finite, regular grid of cells with a finite number of states, where time is discrete. The next state of each cell depends on its current state and the state of its 8 neighboring cells. This classic problem has applications in mathematics, computability theory, and theoretical biology. This assignment not only tests the students' programming skills but also encourages creativity and exploration of emergent behaviors in Conway's Game of Life. It provides an opportunity for students to hone their programming skills in object-oriented programming within the Unity framework that we developed during our class while also gaining a deeper understanding of cellular automata. **Please strictly follow the framework we developed and modify it suitably for this assignment.** This assignment requires students to extensively research the topic on their own, apart from the instructional materials covered during the class.

You must use the pathfinding framework that we developed during the class hour for this assignment. You are most welcome to customize or refactor this framework for optimization, etc.

ChatGPT Warning:

Students are welcome to use ChatGPT for logical thinking, coding help; however, the instructor has the right to question individual students during office hours to explain different sections of the code to determine whether students have understood and written the code themselves or directly copied and pasted it from ChatGPT without understanding it. If a student is unable to explain the code, their programming assignment will be graded as zero, even if the assignment produces the intended output.

Instructions:

1. **Game Logic Implementation (50 points)**
    - Implement the rules of Conway's Game of Life accurately as described in the problem statement:
        - Cells are born if they have exactly 3 neighbors.
        - Living cells die if they have fewer than 2 neighbors (loneliness).
        - Living cells die if they have more than 3 neighbors (overcrowding).

2. **Pattern Visualization (20 points)**
    - Implement a visually appealing way to display the grid and cells.
    - Use different colors or sprites to distinguish between alive and dead cells.

3. **Pattern Variety and custom patterns (20 points)**
    - Allow users to select from a predefined list of interesting initial patterns (e.g., gliders, oscillators).
    - Any other interesting patterns that you find.

4. **Documentation and Comments (10 points)**
    - Write clear and concise comments in your code to explain complex logic and algorithms.
    - Include screenshots of your simulation showing nice patterns.
    - Create a README file that explains how to run the simulation and interact with it.
    - Cite all reference materials!

5. **Bonus Points (5 points each, max 35 points)**
    - Implement the ability to start and stop the simulation.
    - Add user controls for clearing the grid and generating random initial patterns.
    - Create a grid of cells that can be toggled between alive and dead states by clicking or dragging the mouse.
    - Allow users to control the grid size (increasing or decreasing the number of rows and columns).
    - Implement a feature that allows users to analyze the evolution of patterns over time (generations).
    - Provide controls for stepping through generations one by one.
    - Allow users to speed up or slow down the simulation.

Grading rubric

Accurate implementation of rules (0-50 points)
0 points for not implementing the rules at all

15 points for partial implementation with some inaccuracies
30 points for mostly accurate implementation with minor issues
50 points for a fully accurate implementation (including code for infinite grid)

Pattern Visualization (20 points)
Visual appeal and distinction between alive and dead cells (0-20 points)
0 points for no visualization or unclear distinction
5 points for basic visualization with limited distinction
10 points for clear visualization with some aesthetic appeal
15 points for visually appealing and distinct representation
20 points for exceptional visualization

Predefined patterns and custom pattern creation (20 points)
0 points for no pattern options or custom pattern creation
5 points for limited predefined patterns or limited custom pattern creation
10 points for a variety of predefined patterns and basic custom pattern creation

Clarity of code comments and README (0-10 points)
0 points for no comments or unclear README
2 points for minimal and unclear comments or documentation
10 points for clear comments and a basic README, well-documented code, (including reference materials).

Extra features and user controls (0-35 points)
Up to 35 points for implementing any combination of bonus features:

Grades:

- 90-100 points: Excellent
- 80-89 points: Very Good
- 70-79 points: Good
- 60-69 points: Satisfactory
- Below 60 points: Needs Improvement

Additional Notes for Grading:

- Code quality, readability, and organization will be considered.
- The implementation should be bug-free and handle edge cases gracefully.
- The user interface should be intuitive and responsive.
- Creativity in pattern visualization and user interactions will be appreciated.

Submission Guidelines:

- Students should submit their Unity project files as a package along with any necessary assets as a zip in D2L folder.
- Include a README file with instructions and on how to run and interact with the simulation as a text file.
- Include the documentation along with screenshots of the simulation output as a pdf file.