CLI-Based Game of Life Simulation

Nikola Oljaca

October 12, 2024

Overview

This project implements a Command Line Interface (CLI) based simulation of Conway's Game of Life. The program allows users to create, load, save, and run simulations of a cellular automaton within a customizable world grid. It supports various operations such as setting individual cell states, adding predefined figures like Gliders or Methuselahs, and running the simulation for a specified number of generations.

Files

The repository contains the following key files:

- `CLI.cpp` - Implements the CLI interface allowing users to interact with the Game of Life simulation.
- `World.cpp` - Contains the logic for the world grid and the rules for evolving the simulation.
- `World.h` - Header file defining the World class and its associated methods.

Features

The CLI provides the following options for interacting with the simulation:

1. **Create World:** Initializes a new world grid with user-specified dimensions.
2. **Load World:** Loads a world state from a file.
3. **Save World:** Saves the current world state to a file.
4. **Toggle Printing:** Enables or disables printing of the world state after each generation.
5. **Set Delay:** Sets the delay between generations (in milliseconds).

6. **Run Simulation:** Runs the simulation for a specified number of generations.
7. **Set Cell State:** Allows manual setting of a specific cell's state.
8. **Get Cell State:** Retrieves the current state of a specific cell.
9. **Add Figure:** Adds predefined figures like Glider, Toad, Beacon, or Methuselah at specified coordinates.

Compiling and Running

To compile the project, use the following command:

```
g++ CLI.cpp World.cpp -o gameoflife
```

To run the program, execute the compiled binary:

```
./gameoflife
```

Usage Example

When running the program, you will be prompted with a menu to select various options:

```
1. Create world
2. Load world
3. Save world
4. Toggle printing
5. Set delay
6. Run simulation
7. Set cell state
8. Get cell state
9. Add figure
0. Exit
Enter choice:
```

After choosing an option, follow the prompts to interact with the simulation. For example, creating a world requires specifying the dimensions, while adding a figure requires entering the type of figure and its position.

World Class

The `World` class is responsible for managing the state of the grid and running the simulation. The primary functions include:

- `run(generations, printEnabled, delayMs)`: Runs the simulation for a given number of generations with optional printing and delays.
- `setCellState(x, y, state)`: Sets the state of a specific cell.
- `getCellState(x, y)`: Gets the state of a specific cell.
- `addGlider(x, y)`, `addToad(x, y)`, `addBeacon(x, y)`, `addMethuselah(x, y)`: Adds predefined patterns to the world grid.

License

This project is open source and distributed under the MIT License. Feel free to modify and distribute as per the terms of the license.