HPCA-PC Exercise Sheet 2 — Group 1

Mariia Isaeva
7910764
s6047802@stud.uni-frankfurt.de

Luiz Augusto da Silva Feitosa
7890756
s1506025@stud.uni-frankfurt.de

Joshua Spingler
8243375
s7457265@stud.uni-frankfurt.de

Tim Wolf
7416419
s9677570@stud.uni-frankfurt.de

Conway's Game of Life

Exercise 1.1: Design of the Cellular Automaton

Implementation

Our implementation follows an object-oriented approach with two main classes: *World Class* (`World.h, World.cpp`) and *CLI Class* (`Cli.h, Cli.cpp`). The *World Class* manages the cellular automaton's state and evolution logic and the *CLI Class* handles user interaction and command parsing.

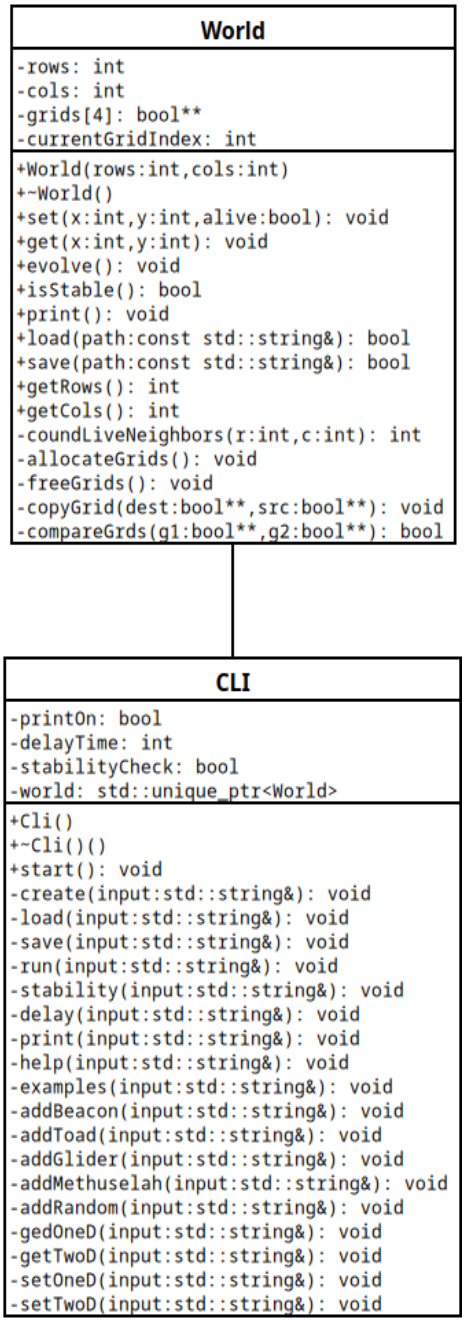


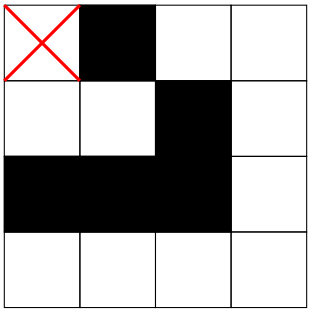
Figure 1: UML Class Diagram of Conway's Game of Life Implementation

Exercise 1.2: Implement Conway's Game of Life

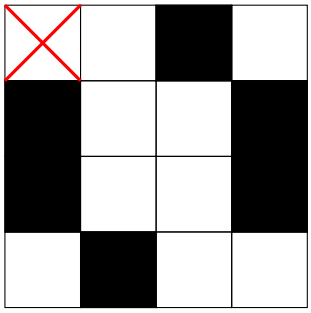
Exercise 1.3: Implement a Command Line Interface

Starting cell of multi-cell figures at the grid position (x,y)

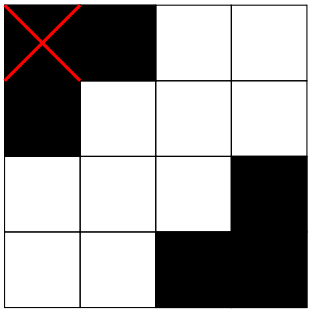
All multi-cell patterns use the top-left corner as their anchor point (x, y), marked with a red X in the diagrams below. When placing patterns using commands like `glider 5 5`, the pattern is positioned such that its top-left cell aligns with grid coordinates $(5, 5)$. The toroidal grid behavior ensures that patterns wrap around the world boundaries. For example, placing a *glider* at $(width - 1, height - 1)$ will cause parts of the pattern to appear at the opposite edges of the grid.



Glider



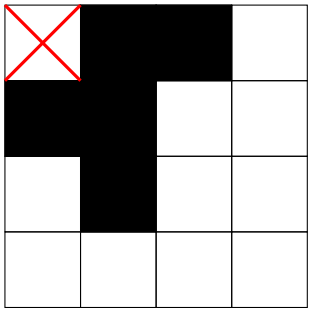
Toad



Beacon

Methuselah

add explanation — work in progress (Luiz)



Methuselah

Exercise 1.4: Optimization Level and Compilation Settings

Exercise 1.5: Simulation Time per Grid Size