# 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

### Implementation

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

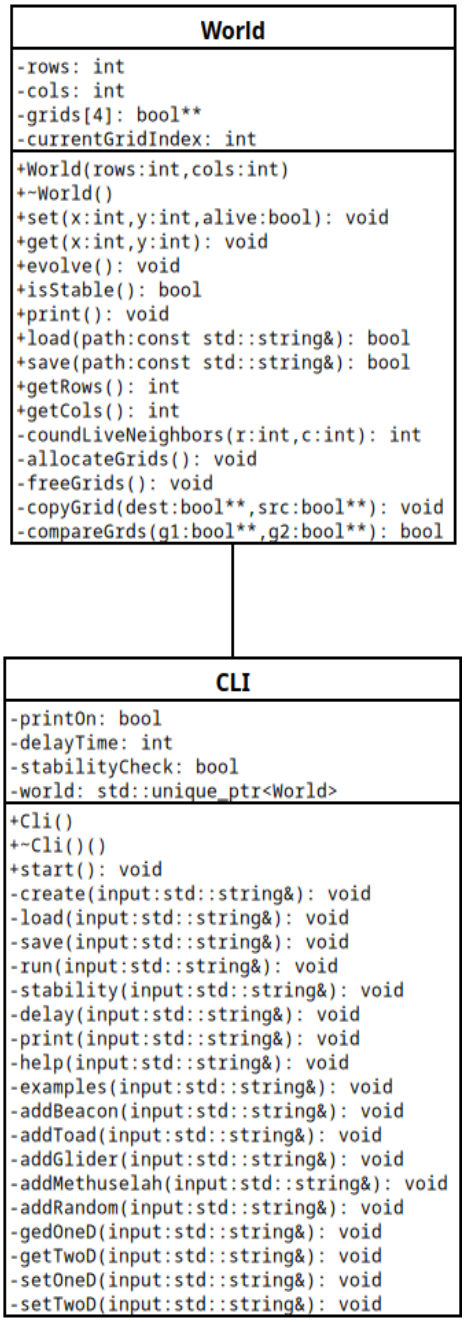


**Figure 1: UML Class Diagram of Conway's Game of Life Implementation**

## 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.

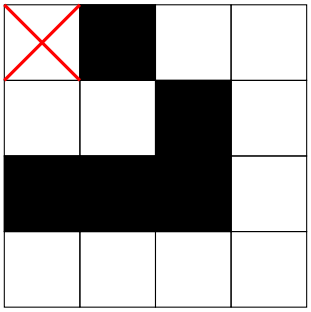

**Figure 2: Glider**

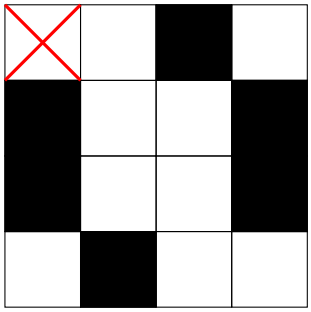

**Figure 3: Toad**

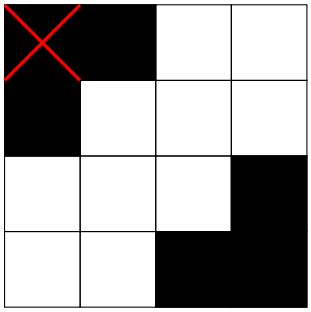

**Figure 4: Beacon**

## Methuselah

We chose one of the smallest methuselahs, the **R-pentomino**, as our pattern. A small configuration (only 5 cells) that takes over 1,100 generations to stabilize. The anchor point (red X) marks the top-left position $(x, y)$ where the pattern is placed.

- **Pattern**: R-pentomino
- **Cells**: 5 live cells in 3×3 bounding box
- **Evolution**: Complex long-term behavior (1,103 generations)
- **Anchor**: Top-left corner at command coordinates

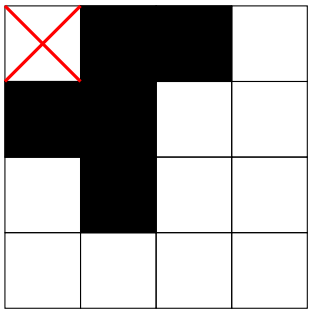

**Figure 5: Methuselah**

**Exercise 1.4: Optimization Level and Compilation Settings**

**Exercise 1.5: Simulation Time per Grid Size**