Game of Life

Team number:

103

Team members:

Yanjuan Li 001497203

Jingyi Cui 001493484

1. Problem Description

- Genetic Algorithm is a method to search the optimal solution by simulating the natural evolution process.
- Game of Life is a cellular automation which imposes fixed rules on the cells and “played” on an infinite two-dimensional discrete grid. Its successive generations are dependent only on the previous generation
- In our project, we use Genetic Algorithm to find the best pattern. Then using this pattern as an input to start our game of life(20*20). After successive generations, we try to find a stable pattern so that the game will last forever.

2.Rules of Game of Life

- Each “live” cell will die if it has fewer than two or more than three neighbors.
- Each “dead” cell will come alive if it has exactly three neighbors.

3.Implementation Details

- Initialization

Initial (“seed”) population: 1,000

Proportion of organisms that survive and breed: 0.5

Maximum number of generations: 10,000

Note: It may take you at least eight minutes to get the result.

3.Implementation Details

- Genotype and Phenotype

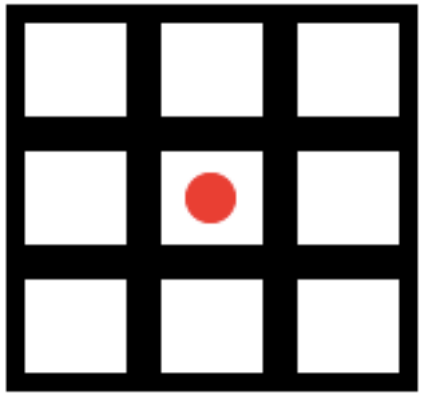
In our project, we use one bit(0 or 1) to represent each gene. If the genotype of the gene is 0, the phenotype of that is dead. If the genotype of the gene is 1, the phenotype of that is live.

About genotype generation, we assign a random integer to each gene, if the random number of that gene less than “cellChanceToLive” (a parameter, set up the chance for cell survival), we will assign 1 to that gene and that gene will live. if the random number of that gene larger than “cellChanceToLive”, we will assign 0 to that gene and that gene will die.

3.Implementation Details

- Next Generation

We consider nine cells as a group, and the next generation of each cell will depend on the status of the other eight cells around it.



3.Implementation Details

- Mutation

We assign a random integer to each gene, if this number is less than "mutationChance" (a parameter, set up the chance for cell mutation), then the status of next generation will be generated randomly (true or false).

- Fitness

It is max generation that our pattern lasts for.

3.Implementation Details

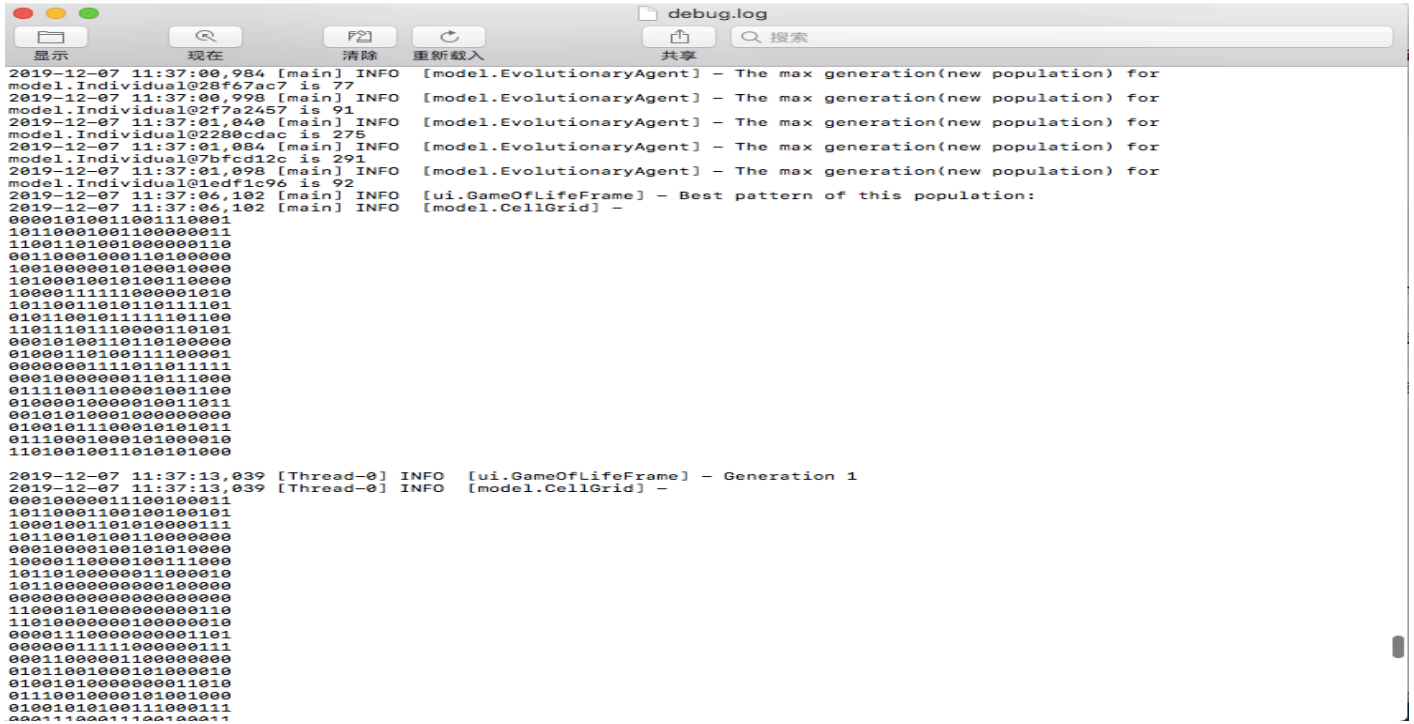
- Survival Function

During our selection, we choose the top20 individuals based on the fitness rules for next new population and the rest part of the new population will be selected randomly.

3.Implementation Details

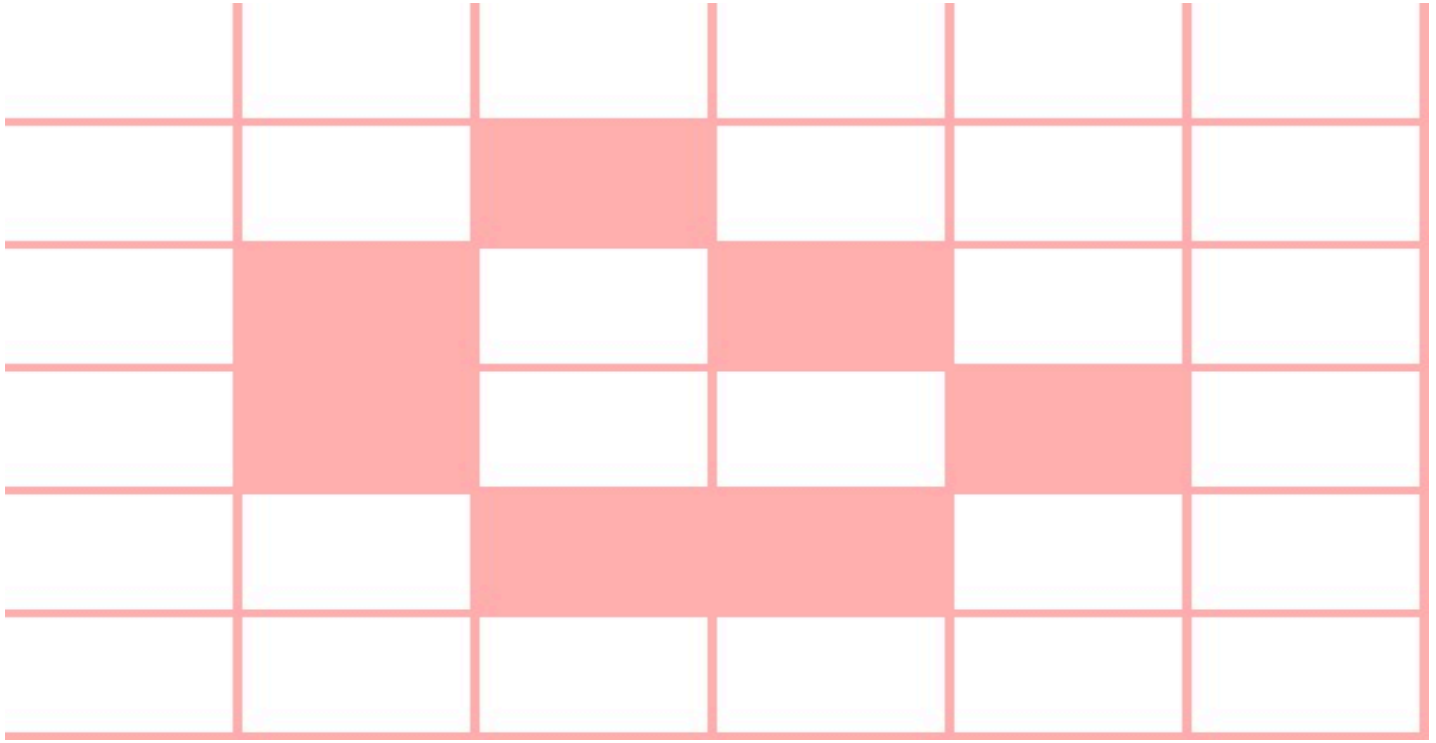
- Log

We add a logging function to keep track of the progress of the evolution, including the best candidate from the final generation.



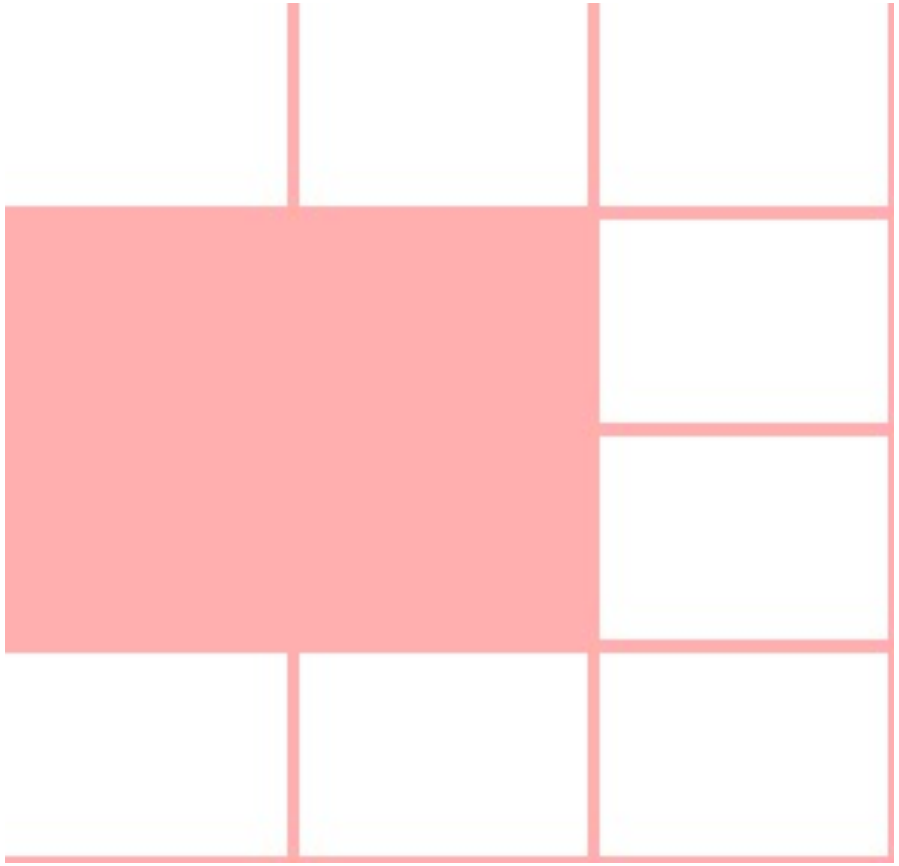
4.Results and Conclusion

- We did many experiments with our project and we found several patterns that can result in a steady state of the game.

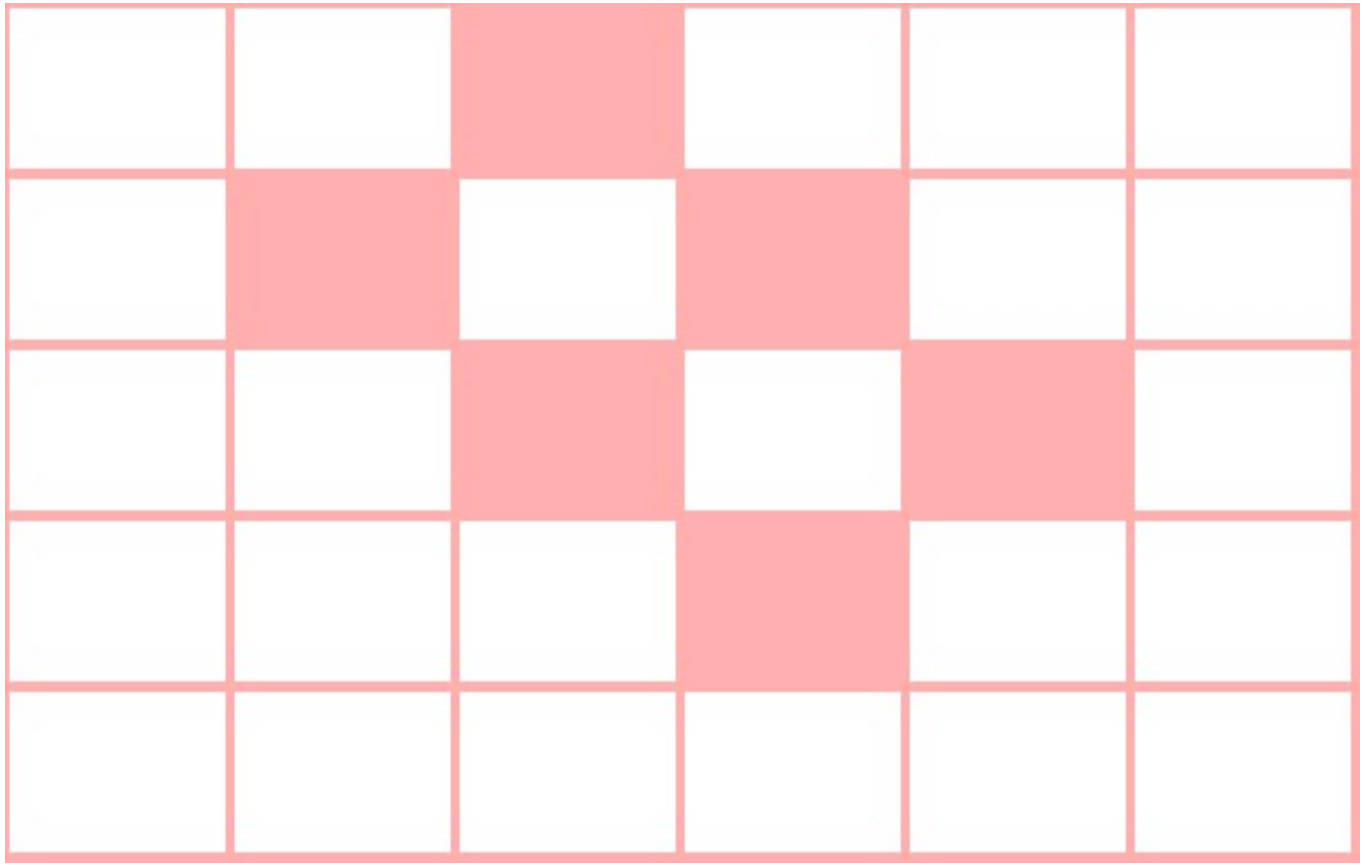


Picture 1

4.Results & Conclusion

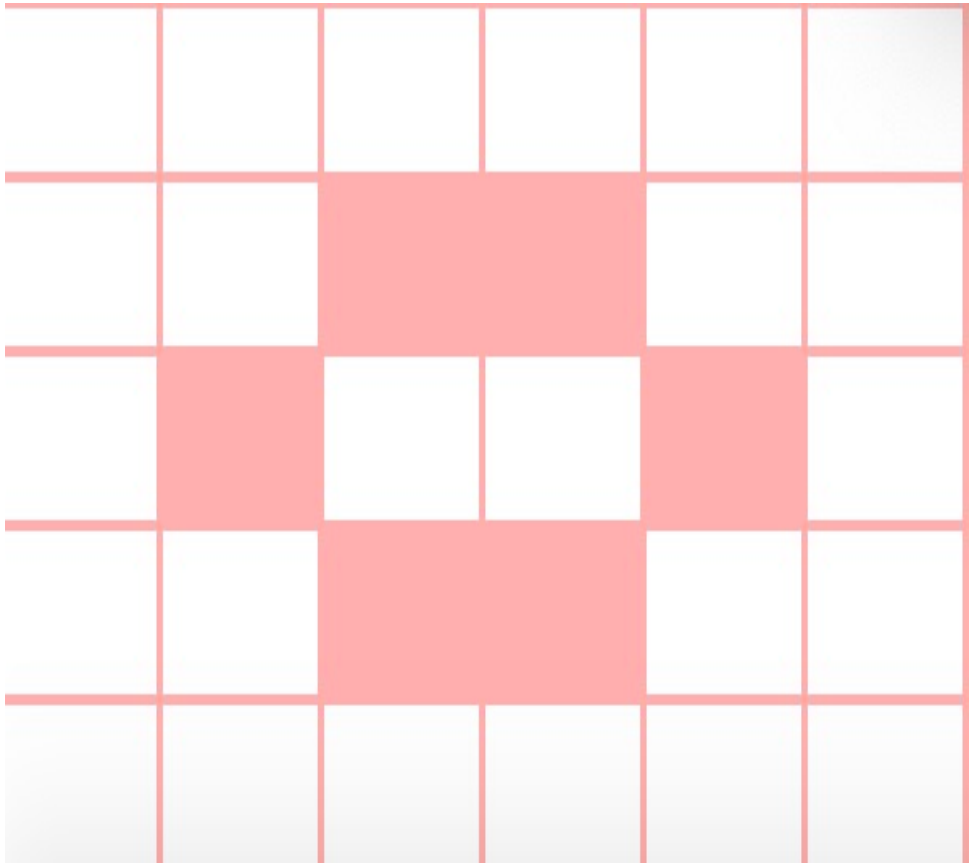


Picture 2

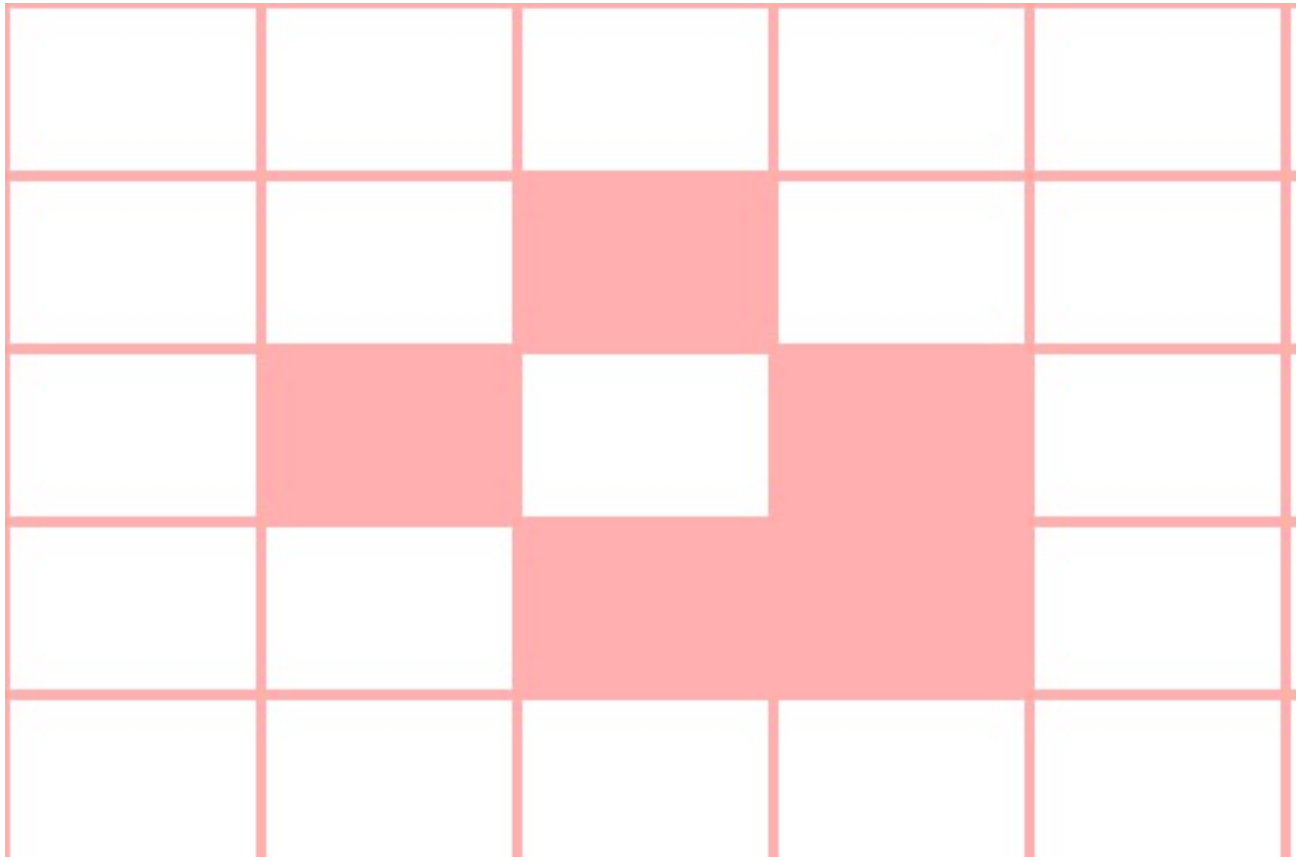


Picture 3

4.Results & Conclusion



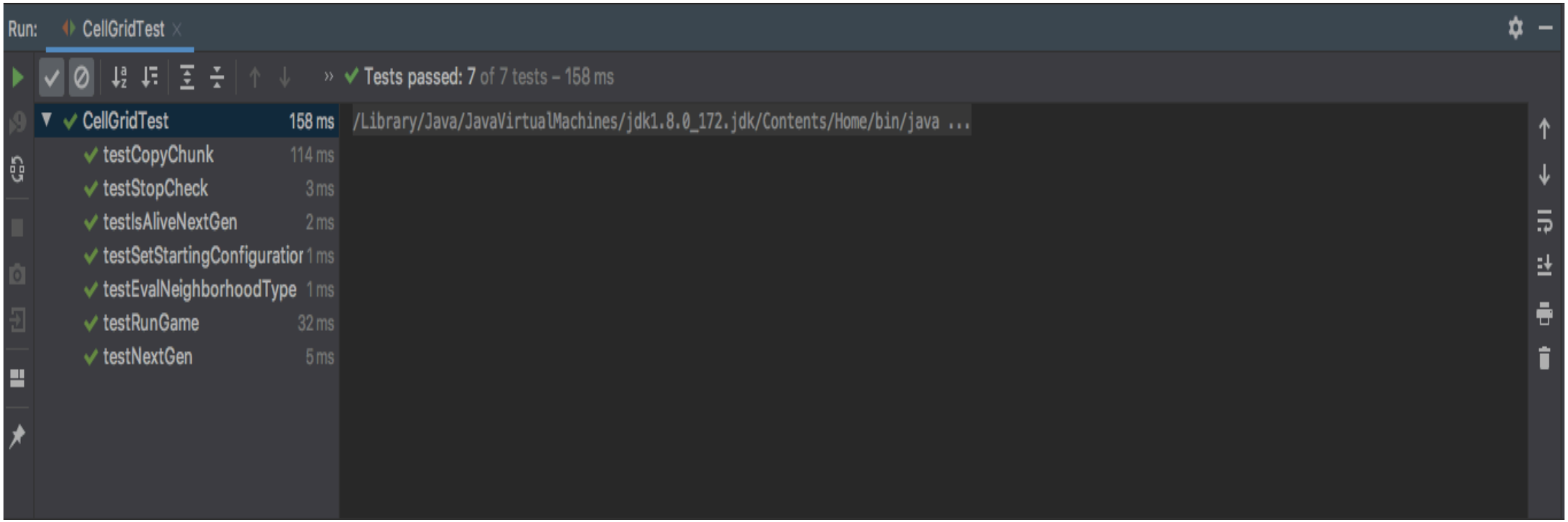
Picture 4



Picture 5

5.Unit Test

- CellGridTest



5.Unit Test

- ConfigurationTest

6.Bonus Part

- We build a UI for our project.