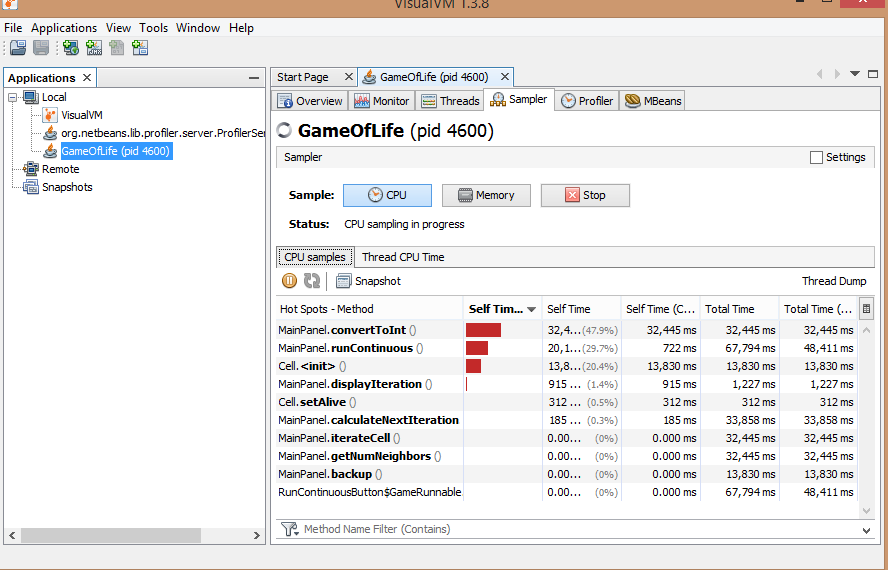
IS 2545 – DELIVERABLE 5: Performance Testing Conway’s Game of Life

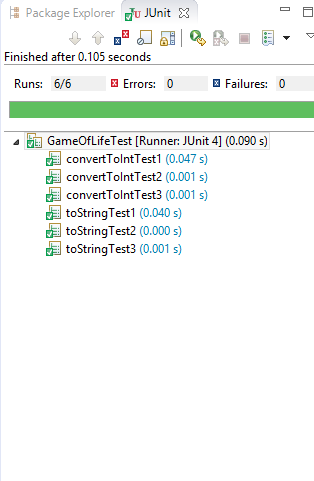
Chase Raymond

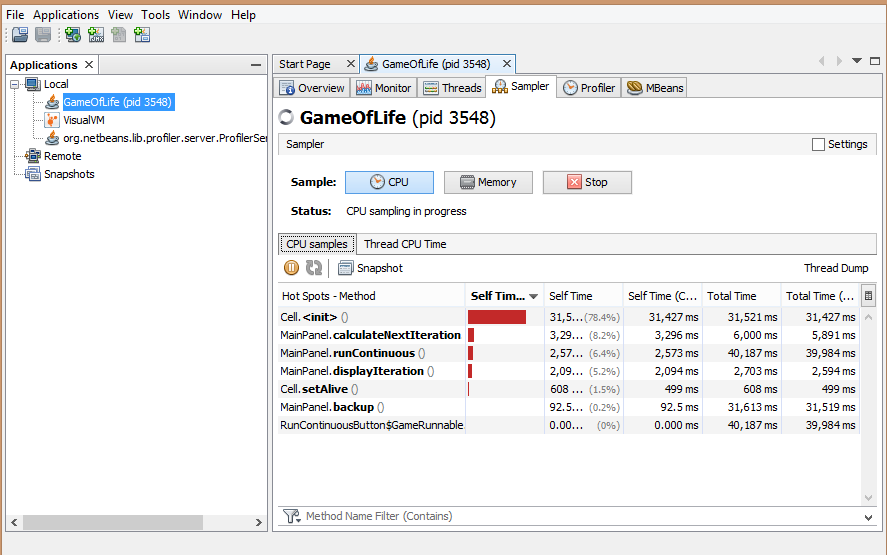
Deliverable 5 was one of the more difficult ones of the semester. This time the code was already there and it was able to run, but there were improvements that were possible to be made. I also had assistance from the professor and had to submit late because of myself unaware of what would be needed in this assignment. I wanted to submit an assignment that I was satisfied with and the tests working rather than something that was not compared to the previous assignment. In this deliverable, I was able to successfully have tests green when running. It was at task just setting up the code to run in VisualVM. At first I was having some errors, but I was able to run the code and profile running the code in VisualVM. While in VisualVM, I selected the Sampler option where I sampled the CPU. This revealed the GameOfLife CPU samples. The main methods that were experiencing strong CPU use included MainPanel.convertToInt, MainPanel.runContinuous, and Cell.toString. After discovering this, the two java files where these methods were located were MainPanel and Cell. The convertToInt method seemed to be written to cause performance issues. The method had excess code which could be removed and replaced with one line of code. The next method in the MainPanel was runContinuous that caused performance issues. This method included a loop that was not needed and could be removed. Finally in the Cell, there was Cell.toString. This method doubled the running of itself and had a loop that was not needed. Now with less code, the methods can run fasters and in the after screenshot, the results are shown. The main CPU Cell that is running is the main function of the program.

When writing tests, I focused my tests around runContinuous, convertToInt, and Cell.toString. The first three tests were for the method runContinuous. I compared the original method (convertToIntO) to the modified method (convertToInt). The tests were focused on getting the same results between the two that any number x would pass through. The three tests passed. Next I created three tests for toString. I tested the new cell state and if the cell is alive or not. The three tests passed. When it came to testing runContinuous, I went with manual testing. I created three cases where I tested the before and after of the method modification of cells between 15\*15, 16\*16, and 17\*17. I tried various inputs and results were confirmed true. The reason behind manual testing was because I was able to verify that the GameOfLife was continuously was running. I was able to compare the expectation and actual outcome and find if there were any differences.

Screenshots are included of the before and after via VisualVM and the manual testing is included after this.

BEFORE



AFTER

**Manual Testing:**

**IDENTIFIER:** CaseOne

**TEST CASE:** Manual test to determine if the runContinuous() method will produce the same output before and after modification of code

**PRECONDITIONS:** GameOfLife with a 15\*15 world is created with the pre-modified runContinuous() method

**EXECUTION STEPS: 1.** Select cells: [7, 3] [8, 3] [9, 3] [6, 4] [10, 4] [6, 5] [10, 5] [6, 6] [10, 6] [6, 7] [10, 7] [7, 8] [8, 8] [9, 8] [8, 9] [8, 10] [8, 11] [8, 12] [8, 13] [8, 14] [8, 15]

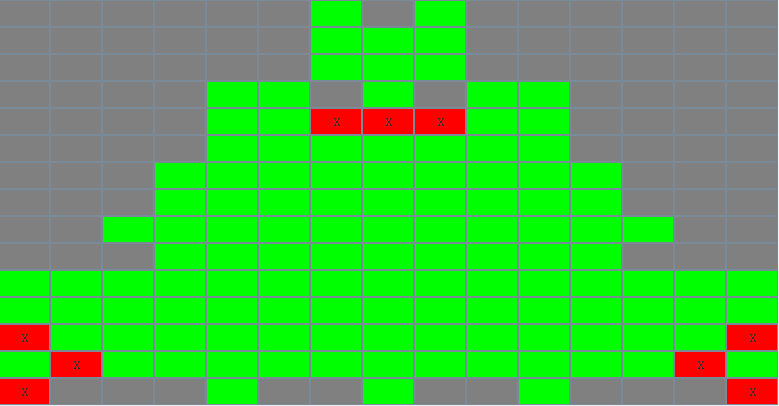
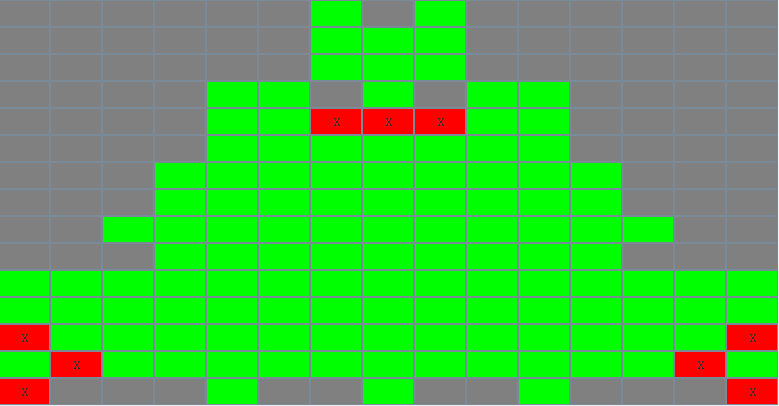
**2.** Select Run Continuous, then stop after 33 seconds

**3.** Note the pattern, then exit

**4.** Change runContinuous() method and run GameOfLife

**5.** Repeat steps 1-3

**POSTCONDITIONS:** The pattern that was noted before and after is the same after changes to runContinuous()



**IDENTIFIER:** CaseTwo

**TEST CASE:** Manual test to determine if the runContinuous() method will produce the same output before and after modification of code

**PRECONDITIONS:** GameOfLife with a 16\*16 world is created with the pre-modified runContinuous() method

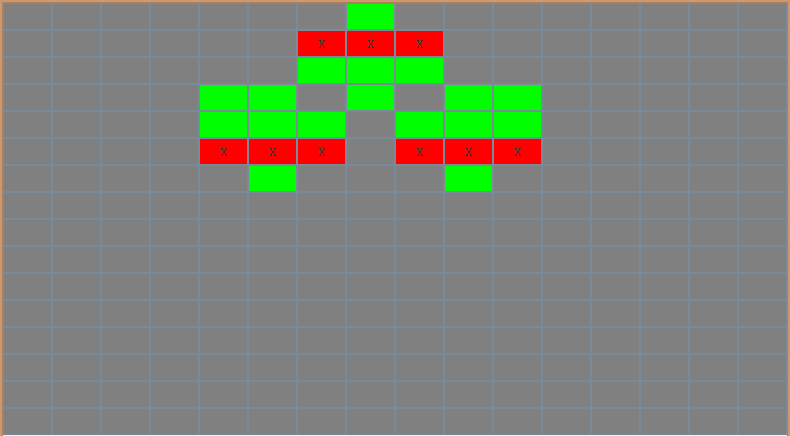
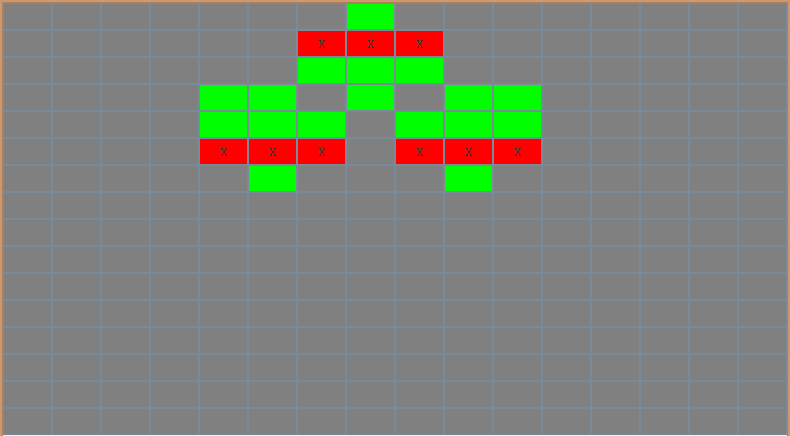
**EXECUTION STEPS: 1.** Select cells: [7, 3] [8, 3] [9, 3] [6, 4] [10, 4] [6, 5] [10, 5] [6, 6] [10, 6] [6, 7] [10, 7]

**2.** Select Run Continuous, then stop after 33 seconds

**3.** Note the pattern, then exit

**4.** Change runContinuous() method and run GameOfLife

**5.** Repeat steps 1-3

**POSTCONDITIONS:** The pattern that was noted before and after is the same after changes to runContinuous()

**IDENTIFIER:** CaseThree

**TEST CASE:** Manual test to determine if the runContinuous() method will produce the same output before and after modification of code

**PRECONDITIONS:** GameOfLife with a 17\*17 world is created with the pre-modified runContinuous() method

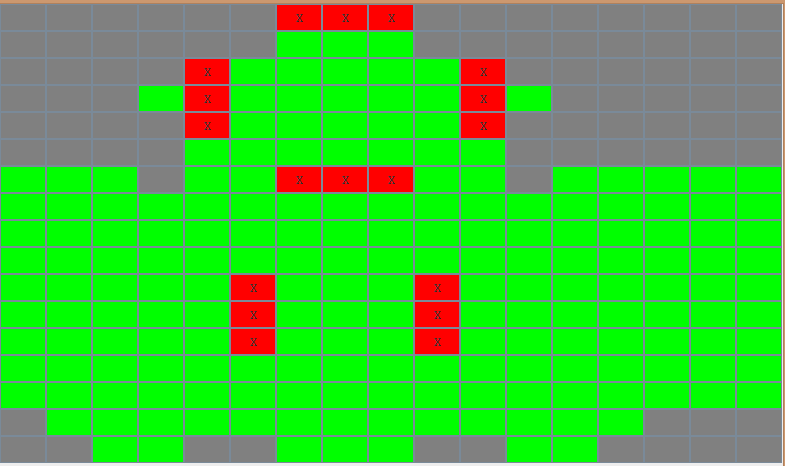
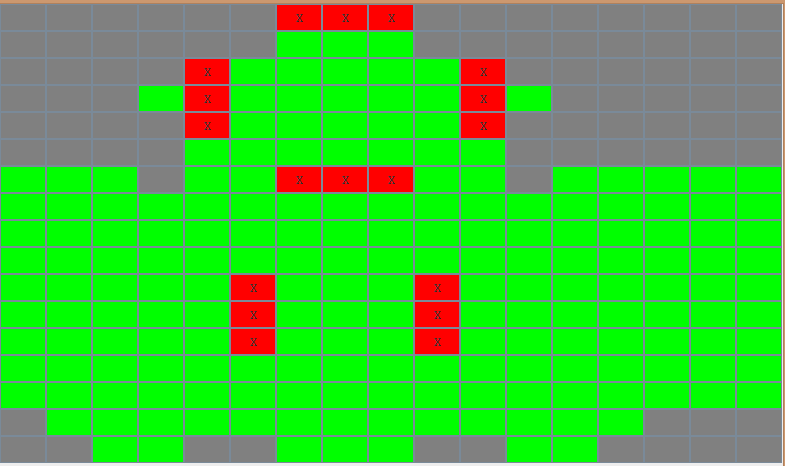
**EXECUTION STEPS: 1.** Select cells: [7, 8] [8, 8] [9, 8] [8, 9] [8, 10] [8, 11] [8, 12] [8, 13] [8, 14] [8, 15]

**2.** Select Run Continuous, then stop after 33 seconds

**3.** Note the pattern, then exit

**4.** Change runContinuous() method and run GameOfLife

**5.** Repeat steps 1-3

**POSTCONDITIONS:** The pattern that was noted before and after is the same after changes to runContinuous()