**Description of Tests:**

Testing values with one test only:

In these series of tests, I want to see the performance of both the programs just using one test per value. Testing out simple tests before the more complex ones should be able to give me a general outline on the performance of the programs. I’ll be using four values (1, 10, 100, 1000), the only constant is that I will be using is one test only.

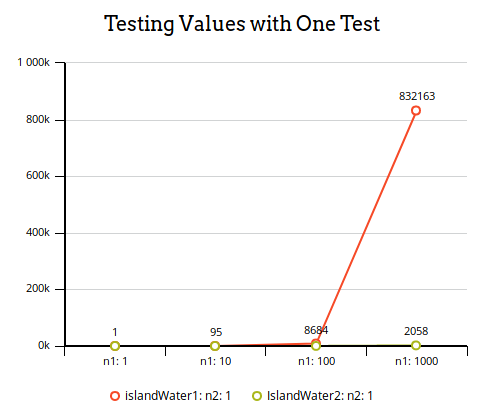
Testing with very large values (stress testing):

In these series of tests, I wanted to stress test the both programs to see if they can handle very large numbers. I believe stress testing these programs is a good indicator on how well a program performs. A good program should be able to allocate resources properly under load. I will using the value 100,000 (as the constant), and will be using testing values of (1, 2, 3).

Testing different values with different amounts of tests:

In these series of tests, I used three values (10, 50, 100), and tested them with different testing values (10, 50, 100). This is more of an “across the board” kind of testing. Testing various values and test values will give me a more concrete view of the performance of the programs because there will be data to back it up. From here I should be able to determine the performance of both programs more clearly.

**Report:**

Testing values with one test:

islandWater1:

Value: Tests: Step count:

1 1 1.00 steps

10 1 95.00 steps

100 1 8684.00 steps

1000 1 832163.00 steps

islandWater2:

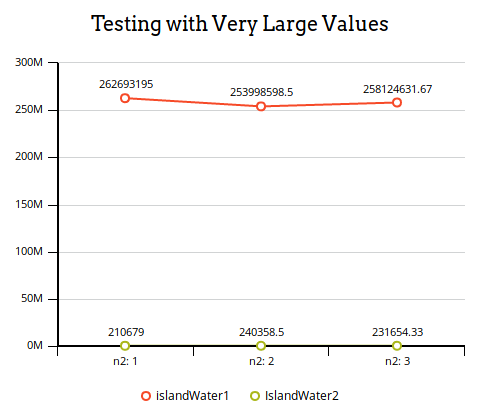
Value: Tests: Step count:

1 1 0.00 steps

10 1 15.00 steps

100 1 224.00 steps

1000 1 2058.00 steps

Testing very large values:

islandWater1:

Value: Tests: Step count:

100000 1 262693195.00 steps

100000 2 253998598.50 steps

100000 3 258124631.67 steps

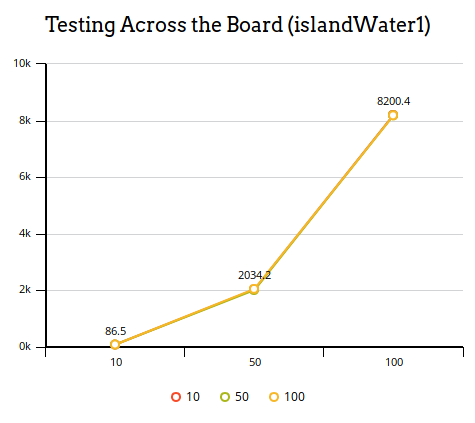
islandWater2:

Value: Tests: Step count:

100000 1 210679.00 steps

100000 2 240358.50 steps

100000 3 231654.33 steps

Testing different values with different amounts of tests:

islandWater1:

Value: Tests: Step count:

10 10 86.50 steps

10 50 87.88 steps

10 100 87.95 steps

50 10 2034.20 steps

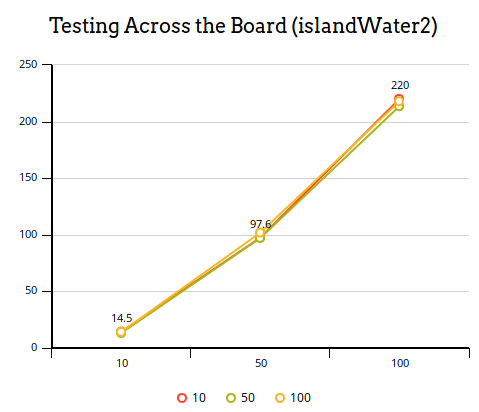
50 50 2011.12 steps

50 100 2053.19 steps

100 10 8200.40 steps

100 50 8187.52 steps

100 100 8186.09 steps

islandWater2:

Value: Tests: Step count:

10 10 14.50 steps

10 50 13.22 steps

10 100 14.22steps

50 10 97.60 steps

50 50 97.34steps

50 100 101.98 steps

100 10 220.00 steps

100 50 213.62 steps

100 100 218.02 steps

**Performance Indication:**

islandWater1:

This program had the overall slowest runtime and higher step count than islandWater2. In the first round of testing (testing with one test with different values), islandWater1 step count exponentially increased when going to 10, 100, and 1000. In the second round of testing (testing with a large number), islandWater1 ran incredibly slow (runtimes lasting over a minute) with very large step counts. In the final round of testing (across the board tests), the tests showed an overall higher step count than islandWater2. It fell behind every test compared to islandWater2.

islandWater2:

This program had the overall best runtime and step count than islandWater1. In the first round of testing (testing with one test with different values), islandWater2 lower step counts compared islandWater1 with step counts only increasing around the factor of ten. The second round of testing (testing with a large number), islandWater2 performed better than islandWater1 with reduced runtimes and much lower step counts. In the final round of testing, islandWater2 performed overall better than islandWater1 with lower step counts in every test, beating it in every test.

**Recommended Program:**

The program that I recommend is islandWater2. The program overall performed better than islandWater1 in every test. Lower step counts and runtimes makes it a clear winner. Even with large integers like 100,000, the program did much better at computing the value in less time (islandWater1 often took a minute to compute this value). The program islandWater2 is the most efficient and fastest code.

**Problems and Pitfalls of Testing:**

An overarching problem with this type of blackbox testing is that not being able to view the mechanics of the code. So weird errors and odd behavior cannot be effectively diagnosed. One of these problems are inconsistent step counts and runtimes in both programs. Running this code on older hardware (like my laptop) may yield slower runtimes and different step counts. Also, unpredicted behavior like having the program hang up and freeze could interfere with testing.