Go-Gitters Deliverable 4: Celestia

By: Lara Brooksbank, Jacob Mattox, Kyle Cooper, and Alexander Swanson

Overview:

For this deliverable, we continue with the creation of test cases by hashing out the remaining twenty. All test cases are listed below:

Test Cases:

1. Test 1:
    1.1. Since Celestia was such a difficult project for us to build we decided that automating the check for the correct packages needed for the build.

2. Test 2:
    2.1. Likewise this test makes sure the makefile required to build the project is found after executing the cmake (Celestia devs changed this on 11/26 - it previously configured in a different way)

3. Test 3:
    3.1. Method Tested: degToRad()
    3.2. Input: 720.0
    3.3. Expected Output: 12.5663706144
    3.4. Actual Output: 12.5663706144

4. Test 4:
    4.1. Method Tested: degToRad()
    4.2. Input: 0.0
    4.3. Expected Output: 0.0
    4.4. Actual Output: 0.0

5. Test 5:
    5.1. Method tested: degToRad()
    5.2. Input: -180.0
    5.3. Expected Output: -3.14159265358979323846
    5.4. Actual Output: -3.14159265358979323846

6. Test 6:
    6.1. Method tested: circleArea()
    6.2. Input: 6.0
    6.3. Expected Output: 113.097335529
    6.4. Actual Output: 113.097335529

7. Test 7:
    7.1. Method tested: circleArea()
    7.2. Input: 0.0
    7.3. Expected Output: 0.0
    7.4. Actual Output: 0.0

8. Test 8:
    8.1. Method tested: circleArea()
    8.2. Input: -1.0
    8.3. Expected Output: Error
    8.4. Actual Output: Error

9. Test 9:
    9.1. Method tested: radToDeg()
    9.2. Input: 3.14159265358979323846
    9.3. Expected Output: 180.0
    9.4. Actual Output: 180.0

10. Test 10:
    10.1. Method tested: radToDeg()
    10.2. Input: 0.0
    10.3. Expected Output: 0.0
    10.4. Actual Output: 0.0

11. Test 11:
    11.1. Method tested: radToDeg()
    11.2. Input: -2 * 3.14159265358979323846
    11.3. Expected Output: -360.0
    11.4. Actual Output: -360.0

12. Test 12:
    12.1. Method tested: square()
    12.2. Input: -1.0
    12.3. Expected Output: 1
    12.4. Actual Output: 1

13. Test 13:
    13.1. Method tested: square()
    13.2. Input: 0.0
    13.3. Expected Output: 0
    13.4. Actual Output: 0

14. Test 14:
    14.1. Method tested: square()
    14.2. Input: 53.0
    14.3. Expected Output: 2809
    14.4. Actual Output: 2809

15. Test 15:
    15.1. Method tested: clamp()
    15.2. Input: -2.0
    15.3. Expected Output: 0
    15.4. Actual Output: 0

16. Test 16:
    16.1. Method tested: clamp()
    16.2. Input: 0.0
    16.3. Expected Output: 0
    16.4. Actual Output: 0

17. Test 17:
    17.1. Method tested: clamp()
    17.2. Input: 0.5
    17.3. Expected Output: 0.5
    17.4. Actual Output: 0.5

18. Test 18:
    18.1. Method tested: cube()
    18.2. Input: -2.0
    18.3. Expected Output: -8.0
    18.4. Actual Output: -8.0

19. Test 19:
    19.1. Method tested: cube()
    19.2. Input: 0.0
    19.3. Expected Output: 0.0
    19.4. Actual Output: 0.0

20. Test 20:
    20.1. Method tested: cube()
    20.2. Input: 10.5
    20.3. Expected Output: 1157.625
    20.4. Actual Output: 1157.625

21. Test 21:
    21.1. Method tested: sign()
    21.2. Input: -2.0
    21.3. Expected Output: -1.0
    21.4. Actual Output: -1.0

22. Test 22:
    22.1. Method tested: sign()
    22.2. Input: 0.0
    22.3. Expected Output: 0.0
    22.4. Actual Output: 0.0

23. Test 23:
    23.1. Method tested: sign()
    23.2. Input: 5.5
    23.3. Expected Output: 1.0
    23.4. Actual Output: 1.0
24. Test 24:
    24.1. Method tested: sphereArea()
    24.2. Input: 10.0
    24.3. Expected Output: 1256.63706144
    24.4. Actual Output: 1256.63706144

25. Test 25:
    25.1. Method tested: sphereArea()
    25.2. Input: 0.0
    25.3. Expected Output: 0.0
    25.4. Actual Output: 0.0

26. Test 26:
    26.1. Method tested: sphereArea()
    26.2. Input: -5.0
    26.3. Expected Output: Error
    26.4. Actual Output: Error

27. Test 27:
    27.1. Method tested: pfmod()
    27.2. Input: 10.0, 5.0
    27.3. Expected Output: 0
    27.4. Actual Output: 0

28. Test 28:
    28.1. Method tested: pfmod()
    28.2. Input: 1.0, 2.0
    28.3. Expected Output: 1.0
    28.4. Actual Output: 1.0

29. Test 29:
    29.1. Method tested: pfmod()
    29.2. Input: 17.0, 0.0
    29.3. Expected Output: nan
    29.4. Actual Output: nan

Report:

This was a very challenging deliverable for our team. None of us have worked with C++ before and we struggled to find code to test that we understood. It was even more difficult to implement the code that we did find in a way that was repeatable with a script. Eventually we began to learn enough C++ to get our test code pulled out of the original source files and into a main function to test from. Getting the output to an html file proved to be challenging also, but we were able to combine similar cases into individual files and run “make” on them to more easily create the necessary executables while also keeping the access to output files.

Overall, this assignment has been extremely time consuming and we have not done a great job of managing our time effeciently. We have struggled to meet every deadline, and our project was a mess until recently. On top of all the problems we have internally as a group, externally the Celestia development team pushed a good many changes to the repository that changed many things about the install and build of Celestia. When the changes went live, we had to spend numerous hours fixes all the bugs in our own framework caused by the Celestia update.