HW 05 - Static Code Analysis

Arun Rao Nayineni

Summary: The changes are made to the original program to make it more legible, readable, and executable by eliminating any extraneous indentation, spaces, and variable renaming, and removing redundant return statements after the code analyzer was performed on the original program. Once this was done static value reached full. Thus, achieving 100% coverage.

1. **The GitHub URL containing the code that was analyzed**

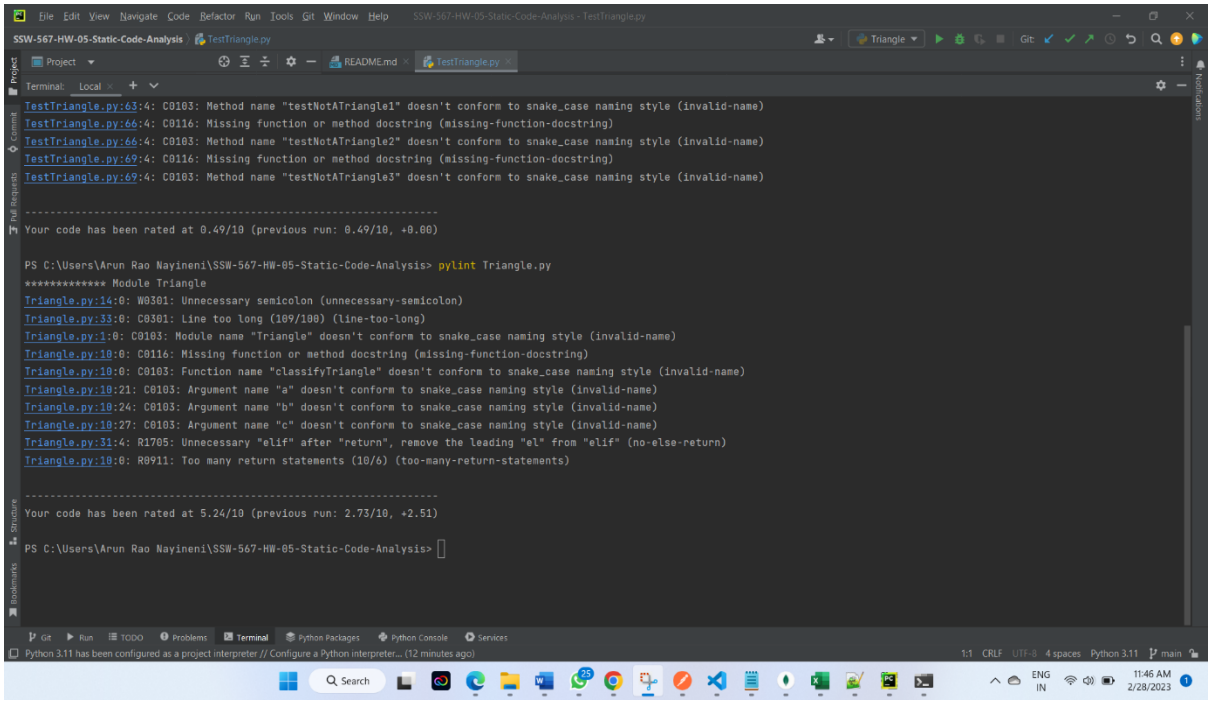
URL: https://github.com/ArunRao1997/SSW-567-HW-05-Static-Code-Analysis

2. **The name and output of the static code analyzer tool you used;**

The tool I used for the static code analysis is **Pylint**

Initial Output (Before making the changes to the code):

For Triangle.py

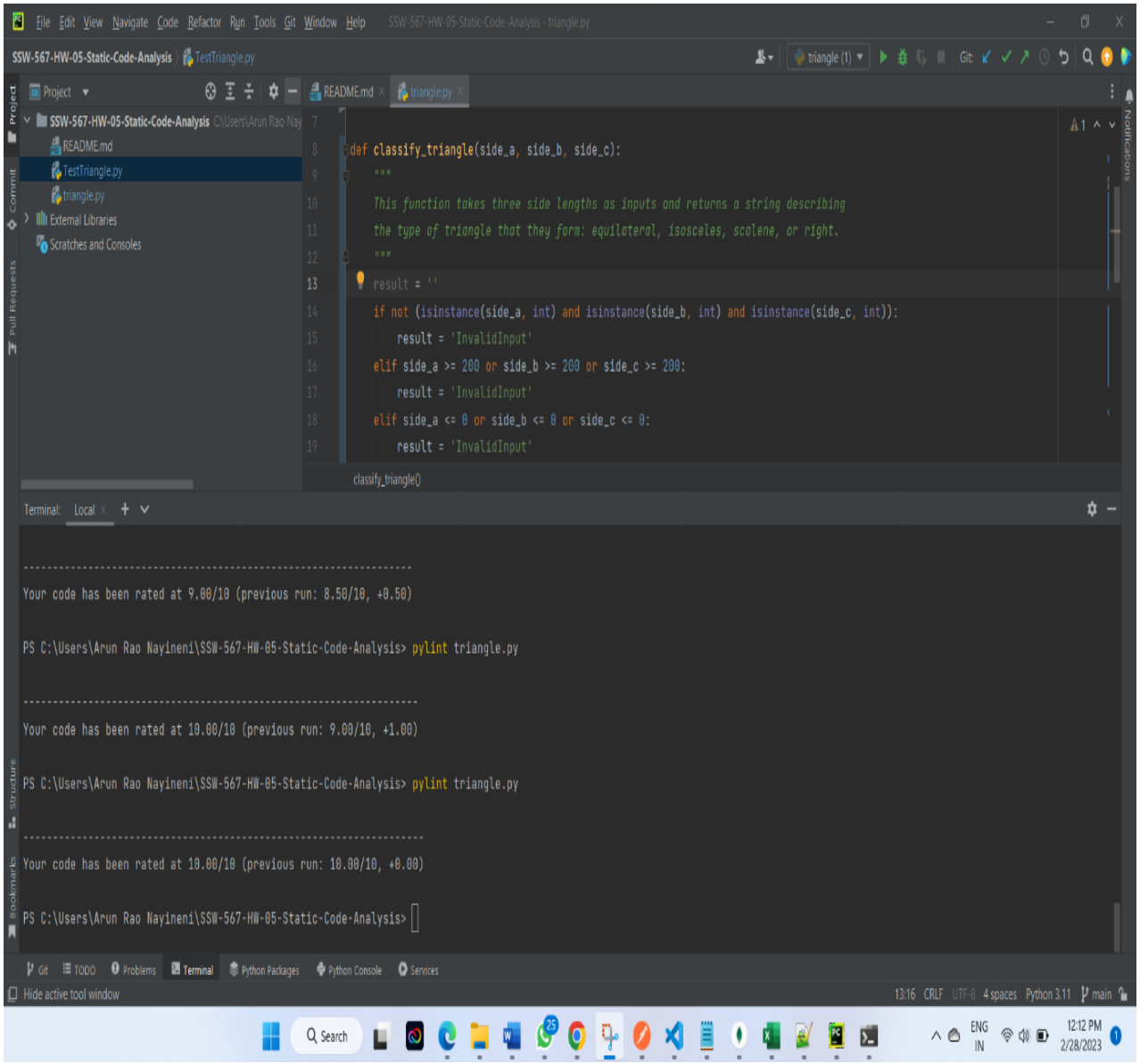


For TestTriangle.py

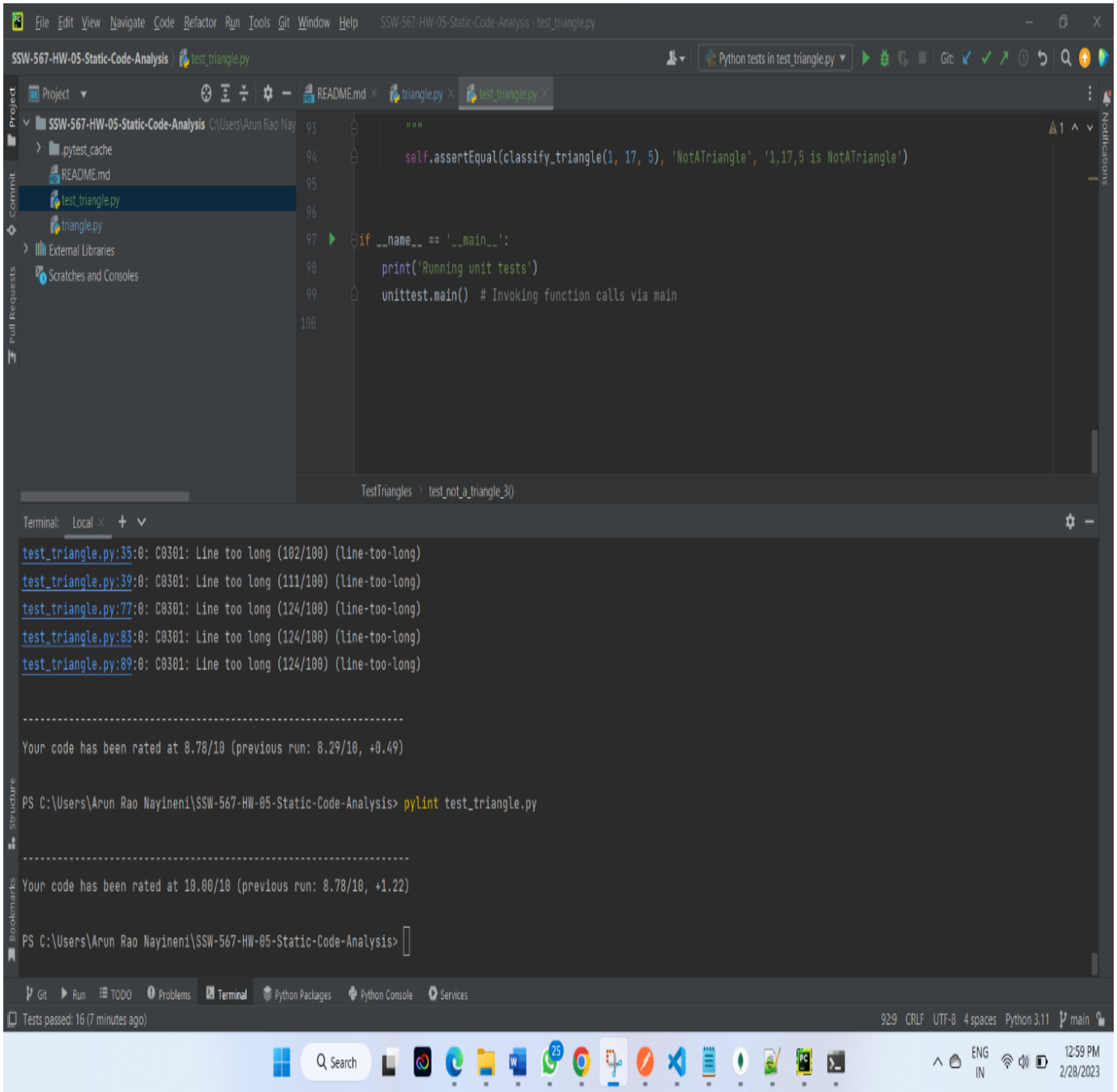
```
PS C:\Users\Arun Rao Nayineni\SSW-567-HW-05-Static-Code-Analysis> pylint TestTriangle.py
************* Module TestTriangle
TestTriangle.py:35:0: C0301: Line too long (101/100) (line-too-long)
TestTriangle.py:38:0: C0301: Line too long (110/100) (line-too-long)
TestTriangle.py:55:0: C0301: Line too long (118/100) (line-too-long)
TestTriangle.py:58:0: C0301: Line too long (104/100) (line-too-long)
TestTriangle.py:61:0: C0301: Line too long (110/100) (line-too-long)
TestTriangle.py:1:0: C0103: Module name "TestTriangle" doesn't conform to snake_case naming style (invalid-name)
TestTriangle.py:17:0: C0115: Missing class docstring (missing-class-docstring)
TestTriangle.py:20:4: C0116: Missing function or method docstring (missing-function-docstring)
TestTriangle.py:20:4: C0103: Method name "testRightTriangleA" doesn't conform to snake_case naming style (invalid-name)
TestTriangle.py:24:4: C0116: Missing function or method docstring (missing-function-docstring)
TestTriangle.py:24:4: C0103: Method name "testRightTriangleAndScalene" doesn't conform to snake_case naming style (invalid-name)
TestTriangle.py:28:4: C0116: Missing function or method docstring (missing-function-docstring)
TestTriangle.py:28:4: C0103: Method name "testEquilateralTriangleA" doesn't conform to snake_case naming style (invalid-name)
TestTriangle.py:31:4: C0116: Missing function or method docstring (missing-function-docstring)
TestTriangle.py:31:4: C0103: Method name "testEquilateralTriangleB" doesn't conform to snake_case naming style (invalid-name)
TestTriangle.py:63:4: C0103: Method name "testNotATriangle1" doesn't conform to snake_case naming style (invalid-name)
TestTriangle.py:66:4: C0116: Missing function or method docstring (missing-function-docstring)
TestTriangle.py:66:4: C0103: Method name "testNotATriangle2" doesn't conform to snake_case naming style (invalid-name)
TestTriangle.py:69:4: C0116: Missing function or method docstring (missing-function-docstring)
TestTriangle.py:69:4: C0103: Method name "testNotATriangle3" doesn't conform to snake_case naming style (invalid-name)

------------------------------------------------------------------
Your code has been rated at 0.49/10 (previous run: 0.49/10, +0.00)

PS C:\Users\Arun Rao Nayineni\SSW-567-HW-05-Static-Code-Analysis>
```

Final output: After the changes have been made.

For triangle.py



For test_triangle.py



3. The name and output of the code coverage tool I used:

The tool used is **coverage.py**

Initial: The initial coverage was 92%.

```
PS C:\Users\Arun Rao Nayineni\SSW-567-HW-05-Static-Code-Analysis> coverage run -m unittest discover
Right angle and Scalene
................
----------------------------------------------------------------------
Ran 16 tests in 0.002s

OK
PS C:\Users\Arun Rao Nayineni\SSW-567-HW-05-Static-Code-Analysis> coverage report -m
Name                Stmts   Miss  Cover   Missing
-----------------------------------------------
test_triangle.py       42      2    95%   77-78
triangle.py            22      3    86%   36-38
-----------------------------------------------
TOTAL                  64      5    92%
PS C:\Users\Arun Rao Nayineni\SSW-567-HW-05-Static-Code-Analysis>
```

Final: The final coverage is 100%, covering all the test cases.

```
PS C:\Users\Arun Rao Nayineni\SSW-567-HW-05-Static-Code-Analysis> coverage run test_triangle.py
Running unit tests
................
----------------------------------------------------------------------
Ran 16 tests in 0.003s

OK
PS C:\Users\Arun Rao Nayineni\SSW-567-HW-05-Static-Code-Analysis> coverage report -m
Name                Stmts   Miss  Cover   Missing
-----------------------------------------------
test_triangle.py       41      0   100%
triangle.py            21      0   100%
-----------------------------------------------
TOTAL                  62      0   100%
PS C:\Users\Arun Rao Nayineni\SSW-567-HW-05-Static-Code-Analysis>
```

4. **Identify both your original test cases and new test cases that you created to achieve at least 80% code coverage.**

As part of the initial request, the aim was to make the code 100% efficient, I fixed the code to 100% efficiency and post that when I ran the test cases against the new code, and was able to achieve coverage of more than 80%.

I achieved an efficiency of 100%. However, it is important to note that achieving 100% efficiency is often not possible in practice. There are often trade-offs between performance and other factors such as code readability, maintainability, and extensibility.

I tested the program with a lot of test cases in the Assignment and there was no need to add more test cases. The thing that worked for me was to make the correction to the code and post that everything was working well.

Overall, I was successful in optimizing the program for efficiency and thoroughly testing it to achieve a high level of coverage. However, there is always room for improvement and refinement.

5. Attach screenshots of the output of the static code analyzer as well as code coverage. You should show a screenshot of the analysis results both before and after any changes that you make to your programs:

I have attached the screenshot of the static code analysis and code coverage above for before and after changes. I have also uploaded the new versions of the code to the git URL.