ComS417 Assignment 4

**Question 1:**

1. A diagram of a triangle with black lines and numbers

   Description automatically generatedA group of white circles with black text

   Description automatically generatedGraph (left is what I drew, right is from the textbook website):

1. There are 36 simple paths: [1,2,3], [1,2,4], [1,2,6], [2,3,2], [2,4,5], [2,4,6], [2,6,2], [3,2,3], [3,2,4], [3,2,6], [4,5,6], [4,6,2], [5,6,2], [6,2,3], [6,2,4], [6,2,6], [1,2,4,5], [1,2,4,6], [2,4,5,6], [2,4,6,2], [3,2,4,5], [3,2,4,6], [4,5,6,2], [4,6,2,3], [4,6,2,4], [5,6,2,3], [5,6,2,4], [6,2,4,5], [6,2,4,6], [1,2,4,5,6], [2,4,5,6,2], [3,2,4,5,6], [4,5,6,2,3], [4,5,6,2,4], [5,6,2,4,5], [6,2,4,5,6]
2. There are 21 prime paths: [3,2,4,5,6], [2,4,5,6,2], [1,2,4,5,6], [4,5,6,2,3], [6,2,4,5,6], [5,6,2,4,5], [4,5,6,2,4], [2,4,6,2], [3,2,4,6], [1,2,4,6], [6,2,4,6], [4,6,2,3], [4,6,2,4], [2,6,2], [3,2,3], [2,3,2], [1,2,3], [1,2,6], [6,2,6], [3,2,6], [4,4]
3. Edge coverage: there are edges [4,4], [4,6], [6,2] that are not covered

Edge pair coverage: all edge pairs are covered by the existing test paths

Node coverage: all nodes are covered by the existing test paths

To satisfy the edge coverage, add these new test paths:

* [1,2,4,4,6]
* [1,2,4,6,2]

**Question 2:**

Control flow graph:

A black and white image of a cross

Description automatically generated

**Question 3:**

1. Downloaded tar file
2. mvn verify command run
3. Number of total mutants generated: 51, killed: 29. output report:

A screenshot of a computer program

Description automatically generated

1. target/pit-reports/index.html:

A graph of coverage report

Description automatically generated

1. com.example -> triangle.java lines 8-22:

A screenshot of a computer program

Description automatically generated

1. The “changed conditional boundary” mutation on line 8 survived. This mutation changed the conditional boundary within the if statement on line 8. It most likely altered the range of valid inputs that the program accepts, ensuring that the triangle classification function considers a broader range of inputs as valid triangles.

This mutation survived because it maintained the logical structure of the code and didn’t introduce any errors or inconsistencies. By adjusting the conditional boundary, it potentially improved the robustness of the triangle classification function by allowing it to handle caases that were previously considered invalid.

1. I added three new test cases:

  @Test

  public void testIsoceles2(){

    final TriangleType type = Triangle.classify(3, 4, 3);

    assertEquals(TriangleType.ISOSCELES, type);

  }

  @Test

  public void testIsoceles3(){

    final TriangleType type = Triangle.classify(5, 3, 5);

    assertEquals(TriangleType.ISOSCELES, type);

  }

  @Test

  public void testIsocelesBoundary(){

    final TriangleType type = Triangle.classify(2147483647, 2147483647, 2147483647);

    assertEquals(TriangleType.EQUILATERAL, type);

  }

Here is the target/pit-report/index.html which is improved from part 4:

A screenshot of a graph

Description automatically generated

**Question 4:**

Team members for the final project are Paige Rolling, Logan Ellsworth, Saiyara Iftekharuzzaman, and Jade Seiler. We are doing our final project on Large Language Models for Test Generation.