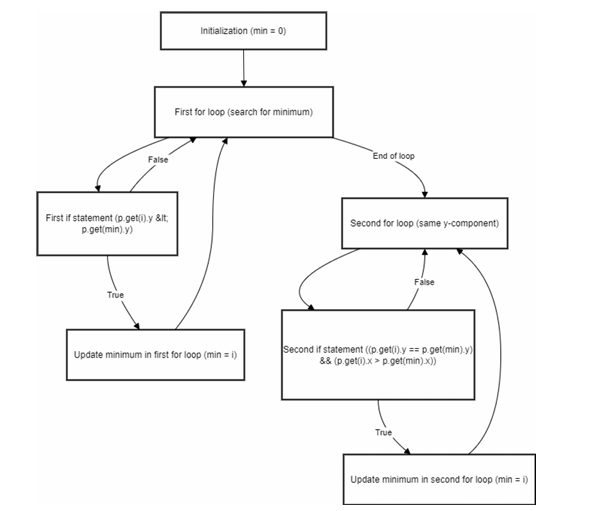
IT313: Software Engineering Lab Session–

Mutation Testing

Archi Jariwala 202201450

Q1. Convert the code comprising the beginning of the doGraham method into a control flow graph (CFG). You are free to write the code in any programming language



Q2.Construct test sets for your flow graph that are adequate for the following criteria:

1.Statement Coverage.

Statement coverage requires that each statement in the code is executed at least once. We need to ensure that all lines (1 to 7 in the code) are executed in at least one of the test cases.

Test Set for Statement Coverage:

Test Case 1: Single Point Input: points = [Point(0, 0)]

Expected Result: min\_index = 0

Path: Entry → 1 → 2 (False) → 5 → 6 (False) → Exit

Test Case 2: Multiple Points with Unique Minimum y-Coordinate Input: points = [Point(1, 3), Point(2, 2), Point(0, 1)]

Expected Result: min\_index = 2

Path: Entry → 1 → 2 (True) → 3 (True) → 4 → 2 (False) → 5 → 6 (False) → Exit

2. Branch Coverage

Branch Coverage requires that each possible branch from a decision point (if statement) is taken at least

once.

This means that both the true and false outcomes of each condition need to be tested.

Test Set for Branch Coverage

● TestCase1: Input vector with only one point (e.g., [(0, 0)])

○ Truebranch: The first loop exits immediately, covering the loop without changes.

● TestCase2: Input vector with two points, one lower y value (e.g., [(1, 1), (2, 0)])

○ Truebranch: The minimum is updated to the second point.

● TestCase3: Input vector with points having the same y but different x values (e.g., [(1, 1),

(3, 1), (2, 1)])

○ Truebranch for the second condition.

● TestCase4: Input vector with all points having the same y value (e.g., [(1, 1), (1, 1),

(1, 1)])

○ False branch: Ensure the second loop is executed without changing the minimum.

3.Basic Condition Coverage

Basic Condition Coverage requires that each individual condition within every decision is evaluated as both True and False at least once.

Test Set for Basic Condition Coverage:

Test Case 1: Single Point (ensures points[i].y < points[min\_index].y is False)

Input: points = [Point(0, 0)]

Expected Result: min\_index = 0

Path: Entry → 1 → 2 (False) → 5 → 6 (False) → Exit

Test Case 2: Unique Minimum y-Coordinate (ensures points[i].y < points[min\_index].y is True)

Input: points = [Point(0, 3), Point(1, 2), Point(2, 1)]

Expected Result: min\_index = 2

Path: Entry → 1 → 2 (True) → 3 (True) → 4 → 2 (False) → 5 → 6 (False) → Exit

Q4.Create a test set that satisfies the path coverage criterion where every loop is explored at least zero, one or two times.

import unittest

from point import Point, find\_min\_point

class TestFindMinPointPathCoverage(unittest.TestCase): def test\_no\_points(self):

points = [ ]

with self.assertRaises(IndexError): # Expect an IndexError due to empty list find\_min\_point(points)

def test\_single\_point(self): points = [Point(0, 0)]

result = find\_min\_point(points)

self.assertEqual(result, points[0]) # Expect the point (0, 0)

def test\_two\_points\_unique\_min(self): points = [Point(1, 2), Point(2, 3)] result = find\_min\_point(points)

self.assertEqual(result, points[0]) # Expect the point (1, 2)

def test\_multiple\_points\_unique\_min(self): points = [Point(1, 4), Point(2, 3), Point(0, 1)] result = find\_min\_point(points)

self.assertEqual(result, points[2]) # Expect the point (0, 1)

def test\_multiple\_points\_same\_y(self):

points = [Point(1, 2), Point(3, 2), Point(2, 2)] result = find\_min\_point(points)

self.assertEqual(result, points[1]) # Expect the point (3, 2)

def test\_multiple\_points\_minimum\_y\_ties(self):

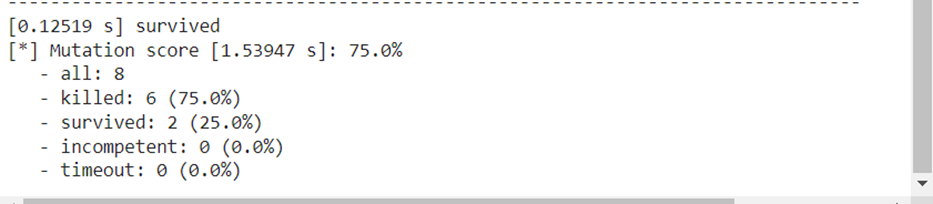
points = [Point(1, 2), Point(2, 2), Point(3, 1), Point(4, 1)] result = find\_min\_point(points)

self.assertEqual(result, points[3]) # Expect the point (4, 1)

# Run the tests if this file is executed if name == " main ":

unittest.main()

Mutation Testing using mut.py tool



1. After generating the control flow graph, check whether your CFG match with the CFG generated by Control Flow Graph Factory Tool and Eclipse flow graph generator. (In your submission document, mention only “Yes” or “No” for each tool).

**Yes**