IT - 314 Software Engineering

Assignment 9 Mutation Testing



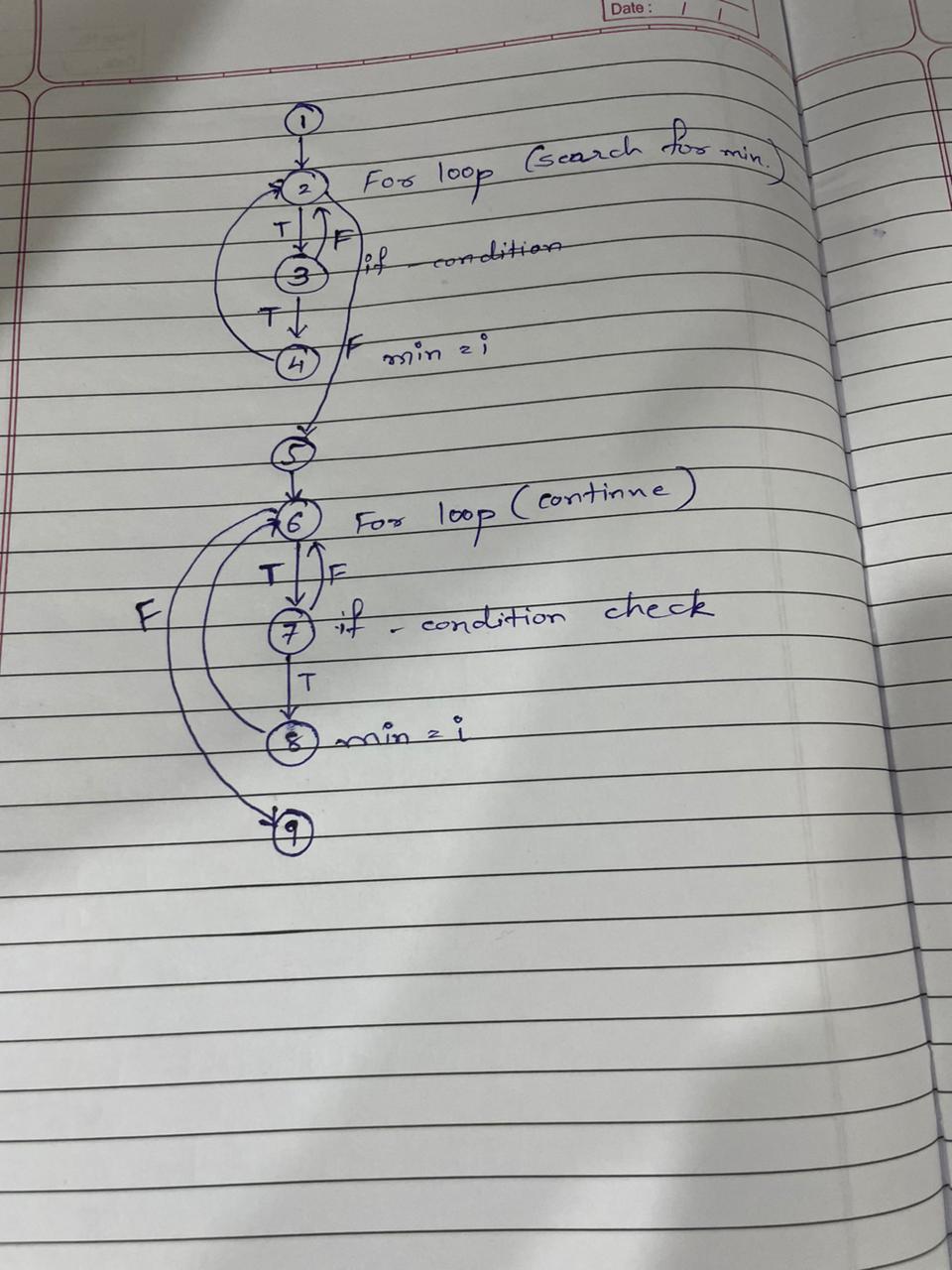
Prof. : Saurabh Tiwari

Name: Divyakumar Tandel

SID: 202201469

November 4, 2024

**Control flow graph:**



**Python Executable code:**

class Point:

def \_\_init\_\_(self, x, y):

self.x = x

self.y = y

def do\_graham(points):

min\_index = 0 # Start with the first point as the minimum

# Search for the minimum y-coordinate

for i in range(1, len(points)):

if points[i].y < points[min\_index].y:

min\_index = i

# Continue along values with the same y component

for i in range(len(points)):

if points[i].y == points[min\_index].y and points[i].x > points[min\_index].x:

min\_index = i

return min\_index

**Test case code:**

import unittest

from graham import Point, do\_graham

class TestGrahamFunction(unittest.TestCase):

def test\_single\_point(self):

points = [Point(1, 2)]

result = do\_graham(points)

self.assertEqual(result, 0)

def test\_multiple\_points(self):

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

result = do\_graham(points)

self.assertEqual(result, 2)

def test\_same\_y\_coordinate(self):

points = [Point(1, 1), Point(2, 1), Point(3, 1)]

result = do\_graham(points)

self.assertEqual(result, 2)

def test\_negative\_coordinates(self):

points = [Point(-1, -2), Point(-3, -1), Point(-2, -3)]

result = do\_graham(points)

self.assertEqual(result, 2)

def test\_mixed\_coordinates(self):

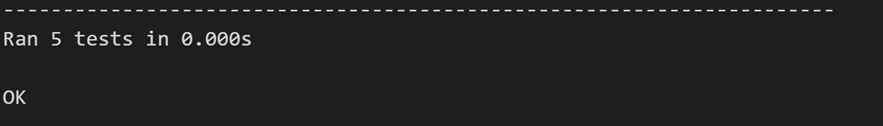
points = [Point(1, 2), Point(-1, -1), Point(0, 0), Point(2, -2)]

result = do\_graham(points)

self.assertEqual(result, 3)

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()



**Mutant 1:**

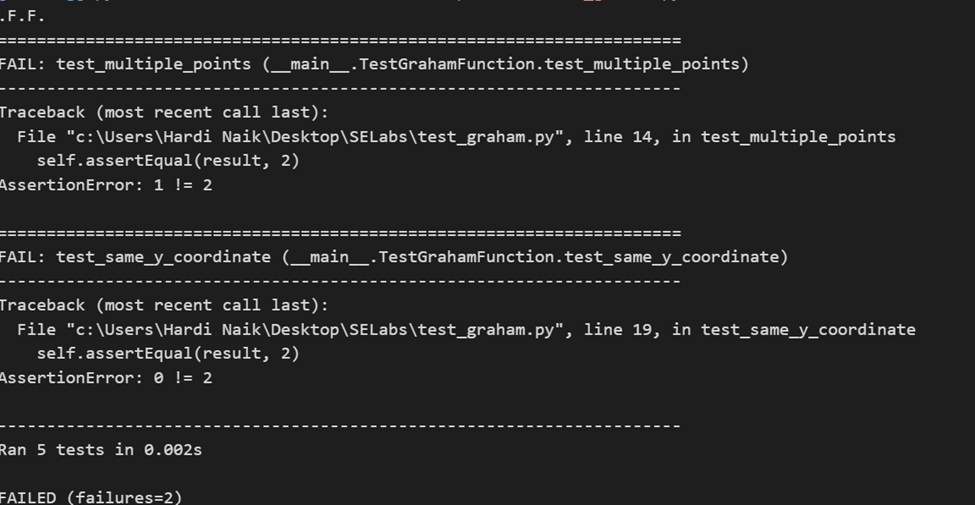
**changes**

for i in range(len(points)):

if points[i].y == points[min\_index].y and points[i].x < points[min\_index].x:

min\_index = i

**result :**



**Mutant 2:**

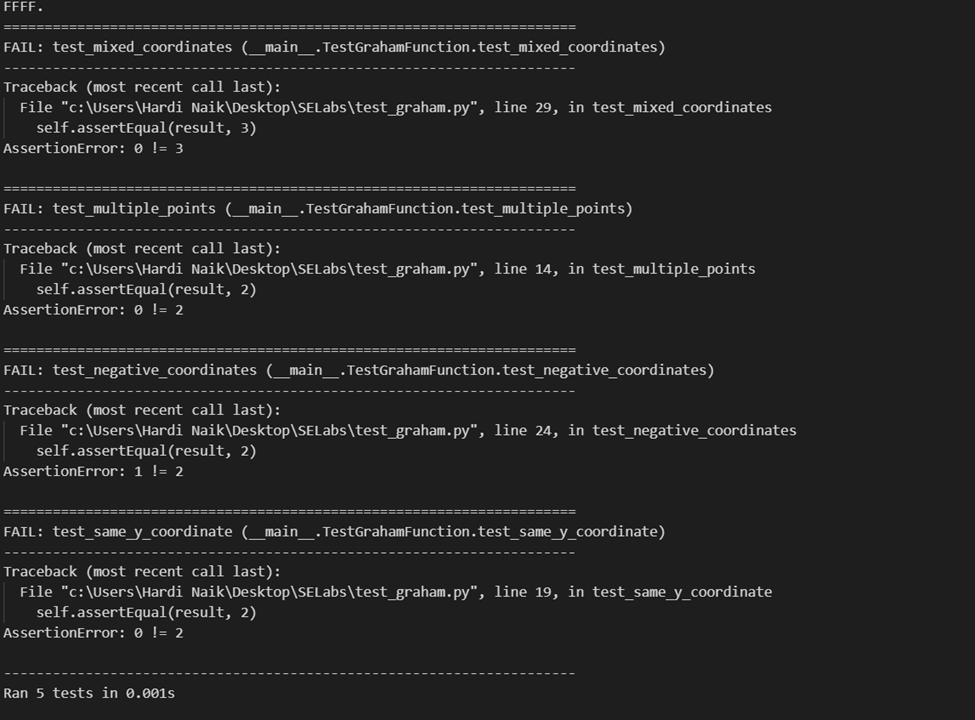
**Changes :**

for i in range(1,len(points)):

if points[i].y < points[min\_index].y:

min\_index = i

**Result :**



**Mutant 3:**

**Changes :**

for i in range(len(points)):

if points[i].y != points[min\_index].y and points[i].x < points[min\_index].x:

min\_index = i

**Result :**

