Lab-9

Mutation Testing

Manthan Parmar 202201416

Q.1. The code below is part of a method in the ConvexHull class in the VMAP system. The following is a small fragment of a method in the ConvexHull class. For the purposes of this exercise, you do not need to know the intended function of the method. The parameter p is a Vector of Point objects, p.size() is the size of the vector p, (p.get(i)).x is the x component of the ith point appearing in p, similarly for (p.get(i)).y. This exercise is concerned with structural testing of code, so the focus is on creating test sets that satisfy some particular coverage criteria.

For the given code fragment, you should carry out the following activities.

```
class Point:
    def __init__(self, x, y):
        self.x = x
        self.y = y

class ConvexHull:
    def do_graham(self, p):
        min_index = 0

        # Search for minimum y-coordinate
        for i in range(1, len(p)):
            if p[i].y < p[min_index].y:
                min_index = i

        # Continue along points with the same y-coordinate
        for i in range(len(p)):
            if (p[i].y == p[min_index].y) and (p[i].x > p[min_index].x):
                min_index = i
```

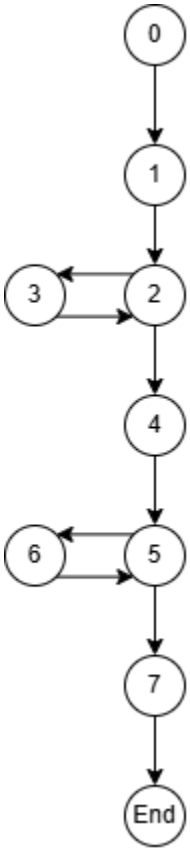
1. Convert the code comprising the beginning of the doGraham method into a control flow graph (CFG).
You are free to write the code in any programming language.

Code Analysis and Node Breakdown

1. Node 0 (Start): The entry point of the function.
2. Node 1: Initializes variables min = 0 and starts the first for loop for searching the minimum y-coordinate.
3. Node 2 (Condition): The condition inside the first for loop:
    - Checks if the current point's y-coordinate is less than the current minimum y-coordinate.
    - If true, goes to Node 3 to update min.
    - If false, continues with the next iteration of the loop or exits if all points are checked.
4. Node 3 (Update): Sets min = i (new minimum point index) and loops back to Node 2.
5. Node 4 (Next Loop): The second for loop for handling points with the same y-coordinate but different x-coordinates.
6. Node 5 (Condition): Checks if the y-coordinate of the current point is equal to the minimum y-coordinate and if the x-coordinate is greater.
    - If true, moves to Node 6 to update min.
    - If false, continues with the next iteration or exits if all points are checked.
7. Node 6 (Update): Sets min = i (new minimum point index) and loops back to Node 5.
8. Node 7 (End): Exits the function and returns the modified vector.

Control Flow Graph Representation

- Node 0 → Start
- Node 0 → Node 1: Initialize min and enter the first loop.
- Node 1 → Node 2: Start first for loop.
- Node 2 → Node 3 (if condition is true) and Node 2 → Node 4 (if loop ends).
- Node 3 → Node 2: Update min and continue to the next iteration of the loop.
- Node 4 → Node 5: Enter the second for loop.
- Node 5 → Node 6 (if condition is true) and Node 5 → Node 7 (if loop ends).
- Node 6 → Node 5: Update min and continue to the next iteration.
- Node 7 → End: Exit function.



2. Construct test sets for your flow graph that are adequate for the following criteria:
a. Statement Coverage.
b. Branch Coverage.
c. Basic Condition Coverage.

Test Cases

a. Statement Coverage

Input: Point(1, 1)
Expected Output: 0
Covers initialization and no loop execution.

Input: Point(2, 2), Point(1, 1)
Expected Output: 1
Covers minimum y search and updating min_index.

Input: Point(1, 1), Point(2, 2)
Expected Output: 0
Covers the scenario where the first point is the minimum.

b. Branch Coverage

Input: Point(1, 1)
Expected Output: 0
Covers the scenario where there's only one point.

Input: Point(2, 2), Point(1, 1)
Expected Output: 1
Covers the condition where the second point has a lower y-value.

Input: Point(1, 1), Point(2, 2)
Expected Output: 0
Covers the scenario where the first point has a lower y-value.

Input: Point(0, 2), Point(1, 1), Point(2, 3)
Expected Output: 1
Covers the minimum y search condition.

Input: Point(1, 1), Point(2, 1), Point(0, 1)
Expected Output: 2
Covers the condition where the y-values are equal.

Input: Point(1, 1), Point(2, 2), Point(2, 1)
Expected Output: 0
Covers both conditions of the second loop.

c. Basic Condition Coverage

Input: Point(1, 1)
Expected Output: 0
Covers initialization with a single point.

Input: Point(2, 2), Point(1, 1)
Expected Output: 1
Covers the condition where p[1].y < p[min_index].y is true.

Input: Point(1, 1), Point(2, 2)
Expected Output: 0
Covers the condition where p[1].y < p[min_index].y is false.

Input: Point(1, 1), Point(2, 1), Point(0, 1)
Expected Output: 2
Covers the condition where p[i].y == p[min_index].y and p[i].x > p[min_index].x is true.

Input: Point(1, 1), Point(2, 2), Point(2, 1)
Expected Output: 0
Covers the condition where p[i].y == p[min_index].y and p[i].x > p[min_index].x is false.

3. For the test set you have just checked can you find a mutation of the code (i.e. the deletion, change or insertion of some code) that will result in failure but is not detected by your test set. You have to use the mutation testing tool.

Convex Hull Code in Python

```
class Point:
    def __init__(self, x, y):
        self.x = x
        self.y = y

class ConvexHull:
    def do_graham(self, p):
        min_index = 0

        # Search for minimum y-coordinate
        for i in range(1, len(p)):
            if p[i].y < p[min_index].y:
                min_index = i

        # Continue along points with the same y-coordinate
        for i in range(len(p)):
            if (p[i].y == p[min_index].y) and (p[i].x > p[min_index].x):
                min_index = i

        return min_index  # Add this line to return the index of the point
with minimum y
```

Test Cases for Convex Hull Code

```
import unittest
from convex_hull import Point, ConvexHull

class TestConvexHull(unittest.TestCase):
    def test_single_point(self):
        self.assertEqual(ConvexHull().do_graham([Point(1, 1)]), 0)

    def test_two_points_lower_y(self):
        self.assertEqual(ConvexHull().do_graham([Point(2, 2), Point(1,
1)]), 1)
```

```
    def test_two_points_higher_y(self):
        self.assertEqual(ConvexHull().do_graham([Point(1, 1), Point(2,
2)]), 0)

    def test_three_points_min_y_search(self):
        self.assertEqual(ConvexHull().do_graham([Point(0, 2), Point(1, 1),
Point(2, 3)]), 1)

    def test_three_points_same_y(self):
        self.assertEqual(ConvexHull().do_graham([Point(1, 1), Point(2, 1),
Point(0, 1)]), 2)

    def test_three_points_different_conditions(self):
        self.assertEqual(ConvexHull().do_graham([Point(1, 1), Point(2, 2),
Point(2, 1)]), 0)

if __name__ == '__main__':
    unittest.main()
```

OUTPUT

```
PS D:\CODING> python -u "d:\CODING\New folder\test_convex_hull.py"
.F.F..
======================================================================
FAIL: test_three_points_different_conditions
(__main__.TestConvexHull.test_three_points_different_conditions)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "d:\CODING\New folder\test_convex_hull.py", line 21, in
test_three_points_different_conditions
    self.assertEqual(ConvexHull().do_graham([Point(1, 1), Point(2, 2), Point(2, 1)]),
0)
AssertionError: 2 != 0

FAIL: test_three_points_same_y (__main__.TestConvexHull.test_three_points_same_y)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "d:\CODING\New folder\test_convex_hull.py", line 18, in
test_three_points_same_y
    self.assertEqual(ConvexHull().do_graham([Point(1, 1), Point(2, 1), Point(0, 1)]),
2)
AssertionError: 1 != 2

----------------------------------------------------------------------
Ran 6 tests in 0.004s

FAILED (failures=2)
PS D:\CODING> python -u "d:\CODING\New folder\test_convex_hull.py"
.F.F..
======================================================================
FAIL: test_three_points_different_conditions
(__main__.TestConvexHull.test_three_points_different_conditions)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "d:\CODING\New folder\test_convex_hull.py", line 21, in
test_three_points_different_conditions
    self.assertEqual(ConvexHull().do_graham([Point(1, 1), Point(2, 2), Point(2, 1)]),
0)
AssertionError: 2 != 0

======================================================================
FAIL: test_three_points_same_y (__main__.TestConvexHull.test_three_points_same_y)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "d:\CODING\New folder\test_convex_hull.py", line 18, in
test_three_points_same_y
    self.assertEqual(ConvexHull().do_graham([Point(1, 1), Point(2, 1), Point(0, 1)]),
2)
AssertionError: 1 != 2

----------------------------------------------------------------------
Ran 6 tests in 0.004s

FAILED (failures=2)
```

The following mut.py file is created when we run the mutation code

```
#!C:\Users\manth\AppData\Local\Microsoft\WindowsApps\PythonSoftwareFoundat
ion.Python.3.12_qbz5n2kfra8p0\python.exe

import sys
from mutpy import commandline

if __name__ == '__main__':


    commandline.main(sys.argv)
```

Mutation 1
Change comparison operators < to <=

OUTPUT

```
FAILED (failures=2)
PS D:\CODING> python -u "d:\CODING\New folder\tempCodeRunnerFile.py"
.F.F..
======================================================================
FAIL: test_three_points_different_conditions
(__main__.TestConvexHull.test_three_points_different_conditions)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "d:\CODING\New folder\tempCodeRunnerFile.py", line 21, in
test_three_points_different_conditions
    self.assertEqual(ConvexHull().do_graham([Point(1, 1), Point(2, 2), Point(2, 1)]),
0)
AssertionError: 2 != 0

======================================================================
FAIL: test_three_points_same_y (__main__.TestConvexHull.test_three_points_same_y)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "d:\CODING\New folder\tempCodeRunnerFile.py", line 18, in
test_three_points_same_y
    self.assertEqual(ConvexHull().do_graham([Point(1, 1), Point(2, 1), Point(0, 1)]),
2)
AssertionError: 1 != 2

----------------------------------------------------------------------
Ran 6 tests in 0.001s

FAILED (failures=2)
```

Mutation 2
Initialize min_index to 1 instead of 0

OUTPUT

```
PS D:\CODING> python -u "d:\CODING\New folder\test_convex_hull.py"
EF.FF.
======================================================================
ERROR: test_single_point (__main__.TestConvexHull.test_single_point)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "d:\CODING\New folder\test_convex_hull.py", line 6, in test_single_point
    self.assertEqual(ConvexHull().do_graham([Point(1, 1)]), 0)
                     ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "d:\CODING\New folder\convex_hull.py", line 17, in do_graham
    if (p[i].y == p[min_index].y) and (p[i].x > p[min_index].x):
                  ~^^^^^^^^^^^^
IndexError: list index out of range

======================================================================
FAIL: test_three_points_different_conditions
(__main__.TestConvexHull.test_three_points_different_conditions)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "d:\CODING\New folder\test_convex_hull.py", line 22, in
test_three_points_different_conditions
    self.assertEqual(ConvexHull().do_graham([Point(1, 1), Point(2, 2), Point(2, 1)]),
0)
AssertionError: 2 != 0

======================================================================
FAIL: test_three_points_same_y (__main__.TestConvexHull.test_three_points_same_y)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "d:\CODING\New folder\test_convex_hull.py", line 19, in
test_three_points_same_y
    self.assertEqual(ConvexHull().do_graham([Point(1, 1), Point(2, 1), Point(0, 1)]),
2)
AssertionError: 1 != 2

======================================================================
FAIL: test_two_points_higher_y (__main__.TestConvexHull.test_two_points_higher_y)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "d:\CODING\New folder\test_convex_hull.py", line 12, in
test_two_points_higher_y
    self.assertEqual(ConvexHull().do_graham([Point(1, 1), Point(2, 2)]), 0)
AssertionError: 1 != 0

----------------------------------------------------------------------
Ran 6 tests in 0.002s

FAILED (failures=3, errors=1)
```

Mutation 3
Remove second loop entirely

OUTPUT

```
PS D:\CODING> python -u "d:\CODING\New folder\test_convex_hull.py"
FF..F.
======================================================================
FAIL: test_single_point (__main__.TestConvexHull.test_single_point)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "d:\CODING\New folder\test_convex_hull.py", line 6, in test_single_point
    self.assertEqual(ConvexHull().do_graham([Point(1, 1)]), 0)
AssertionError: 1 != 0

======================================================================
FAIL: test_three_points_different_conditions
(__main__.TestConvexHull.test_three_points_different_conditions)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "d:\CODING\New folder\test_convex_hull.py", line 22, in
test_three_points_different_conditions
    self.assertEqual(ConvexHull().do_graham([Point(1, 1), Point(2, 2), Point(2, 1)]),
0)
AssertionError: 2 != 0

======================================================================
FAIL: test_two_points_higher_y (__main__.TestConvexHull.test_two_points_higher_y)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "d:\CODING\New folder\test_convex_hull.py", line 12, in
test_two_points_higher_y
    self.assertEqual(ConvexHull().do_graham([Point(1, 1), Point(2, 2)]), 0)
AssertionError: 1 != 0

----------------------------------------------------------------------
Ran 6 tests in 0.002s

FAILED (failures=3)
```

Mutation 4

Alter order of points in second condition

OUTPUT

```
PS D:\CODING> python -u "d:\CODING\New folder\test_convex_hull.py"
EFF.F.
======================================================================
ERROR: test_single_point (__main__.TestConvexHull.test_single_point)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "d:\CODING\New folder\test_convex_hull.py", line 6, in test_single_point
    self.assertEqual(ConvexHull().do_graham([Point(1, 1)]), 0)
                     ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "d:\CODING\New folder\convex_hull.py", line 17, in do_graham
    if (p[i].y > p[min_index].y) and (p[i].x < p[min_index].x):
                 ~^^^^^^^^^^^^
IndexError: list index out of range

======================================================================
FAIL: test_three_points_different_conditions
(__main__.TestConvexHull.test_three_points_different_conditions)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "d:\CODING\New folder\test_convex_hull.py", line 22, in
test_three_points_different_conditions
    self.assertEqual(ConvexHull().do_graham([Point(1, 1), Point(2, 2), Point(2, 1)]),
0)
AssertionError: 2 != 0

======================================================================
FAIL: test_three_points_min_y_search
(__main__.TestConvexHull.test_three_points_min_y_search)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "d:\CODING\New folder\test_convex_hull.py", line 16, in
test_three_points_min_y_search
    self.assertEqual(ConvexHull().do_graham([Point(0, 2), Point(1, 1), Point(2, 3)]),
1)
AssertionError: 0 != 1

======================================================================
FAIL: test_two_points_higher_y (__main__.TestConvexHull.test_two_points_higher_y)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "d:\CODING\New folder\test_convex_hull.py", line 12, in
test_two_points_higher_y
    self.assertEqual(ConvexHull().do_graham([Point(1, 1), Point(2, 2)]), 0)
AssertionError: 1 != 0

----------------------------------------------------------------------
Ran 6 tests in 0.002s

FAILED (failures=3, errors=1)
```

4. 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 convex_hull import Point, ConvexHull

class TestConvexHull(unittest.TestCase):

    # Case 1: Test with a single point (zero iterations in loops)
    def test_single_point(self):
        self.assertEqual(ConvexHull().do_graham([Point(1, 1)]), 0)

    # Case 2: Test with two points, second has a lower y value (one
iteration in the first loop)
    def test_two_points_lower_y(self):
        self.assertEqual(ConvexHull().do_graham([Point(2, 2), Point(1,
1)]), 1)

    # Case 3: Test with two points, first has a lower y value (one
iteration in the first loop)
    def test_two_points_higher_y(self):
        self.assertEqual(ConvexHull().do_graham([Point(1, 1), Point(2,
2)]), 0)

    # Case 4: Test with three points, minimum y is in the middle (two
iterations in the first loop)
    def test_three_points_min_y_search(self):
        self.assertEqual(ConvexHull().do_graham([Point(0, 2), Point(1, 1),
Point(2, 3)]), 1)

    # Case 5: Test with three points having the same y value (two
iterations in the first loop)
    def test_three_points_same_y(self):
        self.assertEqual(ConvexHull().do_graham([Point(1, 1), Point(2, 1),
Point(0, 1)]), 2)

    # Case 6: Test with three points, testing the behavior of the second
loop (two iterations in the first loop)
    def test_three_points_different_conditions(self):
        self.assertEqual(ConvexHull().do_graham([Point(1, 1), Point(2, 2),
Point(2, 1)]), 0)
```

```
    # Case 7: Test with three points where minimum y is the first point
(two iterations in the first loop)
    def test_three_points_min_y_first(self):
        self.assertEqual(ConvexHull().do_graham([Point(1, 0), Point(2, 2),
Point(3, 3)]), 0)

if __name__ == '__main__':
    unittest.main()
```

5. 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