IT314
Lab-09

Name: Stavan Ravisaheb
ID: 202201436

1. Control Flow Graph (CFG)

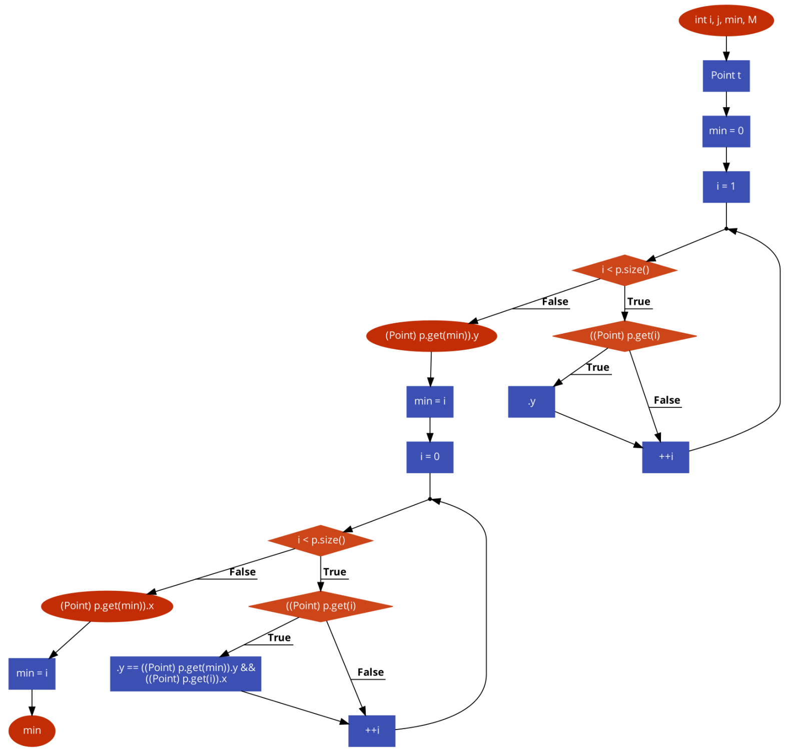
Here is the control flow of the `doGraham` method:

1. **Start Node**: Entry point to the method.
2. **Initialization**: `int i, j, min, M;` and setting `min = 0`.
3. **First Loop**: `for(i = 1; i < p.size(); ++i)`
    - **Condition**: `if (((Point) p.get(i)).y < ((Point) p.get(min)).y)`
        - **True Path**: `min = i;`
        - **False Path**: Continue to next iteration.
4. **Second Loop**: `for(i = 0; i < p.size(); ++i)`
    - **Condition**: `if (((Point) p.get(i)).y == ((Point) p.get(min)).y && ((Point)`

`p.get(i)).x > ((Point)`
`p.get(min)).x)`

  - **True Path**: `min = i;`
  - **False Path**: Continue to next iteration.

5. **End Node**: End of the method.



2. Test Sets for Coverage Criteria:

a. Statement Coverage

To cover every statement in the code, we need at least one test case that:

Executes both for loops.

Satisfies the conditions in both if-statements at least once.

Test Case for Statement Coverage:

Input vector p with points that satisfy the y < y and y == y && x > x conditions.

b. Branch Coverage

To cover every branch, we need test cases that:

Cause both true and false outcomes for each if-condition.

Test Cases for Branch Coverage:

1. Test Case 1: Input where ((Point) p.get(i)).y < ((Point) p.get(min)).y is true in the first loop.

2. Test Case 2: Input where ((Point) p.get(i)).y == ((Point) p.get(min)).y && ((Point) p.get(i)).x > ((Point) p.get(min)).x is true in the second loop.

3. Test Case 3: Input where both conditions are false, causing the loops to iterate without setting min.

c. Basic Condition Coverage

For compound conditions, we need test cases that evaluate each part of the conditions independently.

Test Cases for Basic Condition Coverage:

1. Test Case 1: Satisfies ((Point) p.get(i)).y < ((Point) p.get(min)).y in the first loop.

2. Test Case 2: Fails ((Point) p.get(i)).y < ((Point) p.get(min)).y but satisfies ((Point) p.get(i)).y == ((Point) p.get(min)).y && ((Point) p.get(i)).x > ((Point) p.get(min)).x in the second loop.

3. Test Case 3: Both conditions are false, so min is not updated.

---

3. Mutation Testing:

Mutations:

1. Deletion Mutation:

Delete min = i; in one of the if-statements to check if the test cases can detect the incorrect minimum assignment.

2. Insertion Mutation:

Insert an additional line min = 0; before the second loop to reset min, which might interfere with the logic.

3. Modification Mutation:

Modify ((Point) p.get(i)).y == ((Point) p.get(min)).y to ((Point) p.get(i)).y != ((Point) p.get(min)).y to see if tests detect the change in behavior.

---

4. Path Coverage Test Case:

For path coverage, ensure each loop is executed zero, one, and two times.

Test Cases for Path Coverage:

1. Test Case 1: Empty vector p to check if both loops can execute zero times.

2. Test Case 2: Vector p with only one point, causing each loop to execute exactly once.

3. Test Case 3: Vector p with multiple points, ensuring loops can execute multiple times.