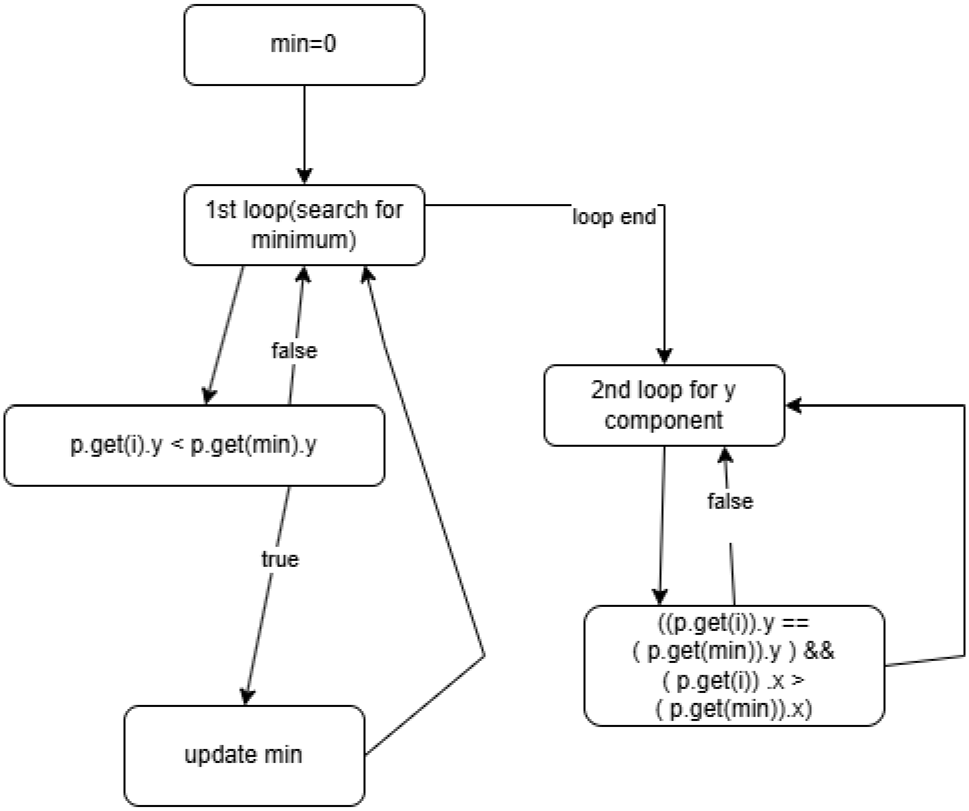
Name :- Nirva Patel

Student id :- 202201071

Question-1.


min=0
1st loop(search for minimum)
loop end
p.get(i).y < p.get(min).y
false
true
update min
2nd loop for y component
false
((p.get(i)).y == (p.get(min)).y) && (p.get(i)) .x > (p.get(min)).x)

Question-2.

⇒ **Statement Coverage**

It Ensures every line of code is executed at least once**.**

Test Case Description	**Input Points**
Covers both loops executing once	[(0, 1), (1, 0), (2, 2)]
Ensures repeated execution of the second loop (multiple identical y-values)	[(0, 0), (1, 0), (2, 0)]

⇒ **Branch Coverage**

Branch coverage checks that each decision point is tested for both true and false outcomes.

Test Case Description	**Input Points**
Tests condition for finding the minimum y-value	[(0, 1), (1, 0), (2, 2)]
Tests condition for cases with multiple identical y-values, validates x-coordinates	[(0, 0), (1, 0), (2, 0)]
Tests condition where no two points have the same y-value	[(1, 1), (2, 2), (0, 3)]

⇒ **Basic Condition Coverage**

Basic condition coverage ensures that each individual condition in decision statements is tested for both true and false outcomes.

Test Case Description	**Input Points**
Covers true evaluation of minimum y condition	[(0, 1), (1, 0), (2, 2)]
Covers true evaluation for identical y condition	[(0, 0), (1, 0), (2, 0)]
Covers false evaluation of minimum y condition	[(1, 1), (2, 2), (0, 3)]

Question-3.

1. **Deletion Mutation**
   Mutation: Remove the line that initializes min to 0.
   Effect: Without this initialization, the code may attempt to use an uninitialized min variable, leading to unpredictable results. This issue might go undetected if the test cases don’t account for handling uninitialized variables.

2. **Change Mutation**
   Mutation: Change the condition in the first loop from **<** to **<= .**
   Effect: This adjustment would make the loop update min even for points with identical y values. If the input contains multiple points with the same lowest y value, it could cause incorrect values to be used in subsequent steps, particularly if points have the same y value but different x values.

3. **Insertion Mutation**

   Mutation*:* Insert a line to reset min to 0 at the end of the first loop.

   Effect:Resetting min after finding the minimum could disrupt the logic in the second loop, preventing it from accurately identifying the correct minimum value. If test cases do not verify the final value of min, this error could remain undetected.

Question-4

Test Case	**Input Points**	**Path Exploration**
Test Case 1	[] (empty list)	Path with zero iterations of both loops.
Test Case 2	[(0, 1)]	Path with one iteration of the first loop, zero iterations of the second loop.
Test Case 3	[(0, 0), (1, 0)]	Path with two iterations of the first loop, one iteration of the second loop (two points with the same y).
Test Case 4	[(0, 0), (1, 1), (2, 0)]	Path with two iterations of the first loop and one iteration of the second loop. It checks the condition when y is equal and x is different.
Test Case 5	[(1, 1), (2, 2), (0, 3)]	Path with three iterations of the first loop and zero iterations of the second loop (no points share the same y).

- **Test Case 1:** Confirms that the function can process an empty input without producing errors.
- **Test Case 2:** Verifies that the function works correctly with a single point as input.
- **Test Case 3:** Checks the behaviour when multiple points share the same y-value, activating the second loop.
- **Test Case 4:** Assesses the function's handling of varied conditions with different y and x values.
- **Test Case 5:** Ensures the function accurately finds the minimum point when no points have matching y-values.

AFTER LAB EXECUTION

Question-1

Control Flow Graph Factory Tool ->Yes

Eclipse flow graph generator -> Yes

Question-2.

Test Case	Input Points	Expected Behavior	Coverage Achieved
TC1	[] (empty list)	Should handle empty input without errors.	Statement Coverage (initial setup)
TC2	[(0, 0)]	Single point; first loop executes once, second does not.	Statement Coverage, Branch Coverage (both ifs false)
TC3	[(1, 1), (0, 0)]	Minimum at (0, 0); tests behavior with two points.	Statement Coverage, Branch Coverage (first if true, second if false)
TC4	[(0, 1), (1, 0)]	Minimum at (1, 0); tests behavior with equal y-values.	Statement Coverage, Branch Coverage (first if true, second if true)
TC5	[(1, 1), (2, 2), (0, 3)]	Minimum at (1, 1); tests multiple distinct points.	Statement Coverage, Branch Coverage (first if true, second if false)
TC6	[(2, 1), (2, 1), (3, 1)]	Minimum at (2, 1); checks behavior with duplicates and same y-values.	Statement Coverage, Branch Coverage (first if false, second if true)

Question-3.

Mutation Type	Description	Effect of Mutation
Deletion	Remove the initialization of the min variable.	Without initializing min, accessing its value may lead to undefined behavior (e.g., accessing an invalid index) if the input is empty.
Insertion	Add a line to reset the min variable after the first loop.	• Resetting min back to 0 may cause the program to incorrectly identify the minimum point after the first loop, leading to errors. • Test cases may pass but do not confirm that min points to the correct minimum after both loops.
Modification	Change the comparison operator in the first loop from <to <= .	• Changing the comparison operator may lead to incorrect behavior when multiple points have the same y value, resulting in wrong outcomes. • Existing test cases may pass but do not check the implications of this change, especially in cases of equal y values.

Question-4.

Test Case ID	Input	Expected Output	Description

TC1	p = []	Undefined behavior or error	Test with an empty list to check how the function handles it.
TC2	p = [(0, 1), (1, 2), (2, 3)]	Minimum point at (0, 1)	Minimum is the first element; checks correct identification of the minimum.
TC3	p = [(1, 2), (0, 1), (2, 3)]	Minimum point at (0, 1)	Minimum found later in the list; tests if the function correctly updates min.
TC4	p = [(1, 1), (1, 1), (1, 1)]	Minimum point at (1, 1)	All elements have the same y value; checks if the function handles ties correctly.
TC5	p = [(2, 2), (3, 3), (1, 1)]	Minimum point at (1, 1)	Minimum is the last element; tests if the function correctly identifies the minimum.
TC6	p = [(2, 3), (1, 3), (0, 3)]	Minimum point at (0, 3)	Minimum is the first element withthe same y as others; checks handling of y value.