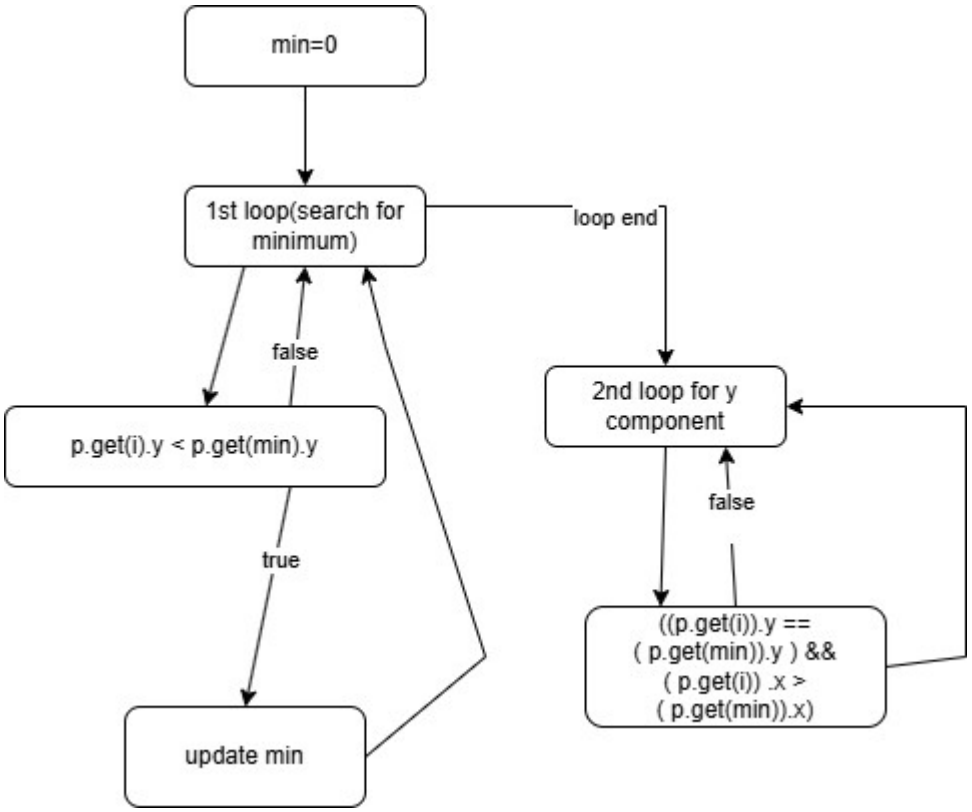
NAME – SAHIL CHAUDHARI

ID- 202201171

Q1.



Q2.

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 the second loop executes (multiple same y)	[(0, 0), (1, 0), (2, 0)]

Branch Coverage

Ensures every decision point is tested for both true and false outcomes.

Test Case Description	Input Points
Evaluates the condition finding minimum y	[(0, 1), (1, 0), (2, 2)]
Evaluates condition with multiple same y, checks x	[(0, 0), (1, 0), (2, 0)]
Evaluates condition where no two points share the same y	[(1, 1), (2, 2), (0, 3)]

Basic Condition Coverage

Ensures every 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 of same y condition	[(0, 0), (1, 0), (2, 0)]
Covers false evaluation of minimum y condition	[(1, 1), (2, 2), (0, 3)]

Q3

1. **Deletion Mutation**
   Remove the line that initializes min to 0
   Effect:
   Without initializing min, the code will likely reference an uninitialized variable, leading to unpredictable behavior. This failure might not be detected if the test cases do not check for uninitialized variable handling.

2. **Change Mutation**
   Change the condition in the first loop from < to <=.

Effect:
This change would cause the first loop to update min even when encountering points with equal y values. If the input has multiple points with the same minimum y, this could lead to incorrect results being processed in the subsequent logic, especially if the input list contains points with the same y value but different x values.

3. **Insertion Mutation**
Insert a line to reset min at the end of the first loop.
Effect:
Resetting min to 0 after finding the minimum could cause the subsequent logic (the second loop) to fail in identifying the correct minimum, as it discards the result of the first loop. If the test cases do not explicitly validate the final value of min, this error could go unnoticed.

Q4

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: Verifies that the function can handle an empty input without errors.

Test Case 2: Ensures the function can handle a single point input.
Test Case 3: Tests the condition where multiple points have the same y value, triggering the second loop.
Test Case 4: Checks mixed conditions with various y and x values.
Test Case 5: Validates that the function correctly identifies the minimum point when there are no points with the same y value.

AFTER LAB EXECUTION

Q1

Control Flow Graph Factory Tool :- Yes
Eclipse flow graph generator :- Yes

Q2

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)

Q3

Mutation Type	**Description**	**Effect of Mutation**
Deletion	Delete 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	Insert a line that resets 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.

Q4

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 with the same y as others; checks handling of y value.