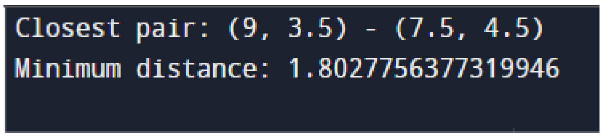
151. Write a program to find the closest pair of points in a given set using the brute force approach. Analyze the time complexity of your implementation. Define a function to calculate the Euclidean distance between two points. Implement a function to find the closest pair of points using the brute force method. Test your program with a sample set of points and verify the correctness of your results. Analyze the time complexity of your implementation. Write a brute-force algorithm to solve the convex hull problem for the following set S of points? P1 (10,0)P2 (11,5)P3 (5, 3)P4 (9, 3.5)P5 (15, 3)P6 (12.5, 7)P7 (6, 6.5)P8 (7.5, 4.5).How do you modify your brute force algorithm to handle multiple points that are lying on the sameline?
Given points: P1 (10,0), P2 (11,5), P3 (5, 3), P4 (9, 3.5), P5 (15, 3), P6 (12.5, 7), P7 (6, 6.5), P8 (7.5, 4.5).
output: P3, P4, P6, P5, P7, P1

AIM:To find the closest pair of points in a given set using the brute force approach

PROGRAM:

```
import math



def euclidean_distance(p1, p2):

    """ Calculate the Euclidean distance between two points p1 and p2. """

    return math.sqrt((p1[0] - p2[0])**2 + (p1[1] - p2[1])**2)



def closest_pair_brute_force(points):

    """ Find the closest pair of points in a set of points using brute force. """

    n = len(points)

    if n < 2:

        return None, float('inf')



    min_distance = float('inf')

    closest_pair = None



    for i in range(n):

        for j in range(i + 1, n):

            distance = euclidean_distance(points[i], points[j])
            if distance < min_distance:
                min_distance = distance
                closest_pair = (points[i], points[j])

    return closest_pair, min_distance


points = [(10, 0), (11, 5), (5, 3), (9, 3.5), (15, 3), (12.5, 7), (6, 6.5), (7.5, 4.5)]
closest_pair, min_distance = closest_pair_brute_force(points)


if closest_pair:
    print(f"Closest pair: {closest_pair[0]} - {closest_pair[1]}")
    print(f"Minimum distance: {min_distance}")
else:
    print("No points or less than 2 points provided.")
```

OUTPUT:



```
Closest pair: (9, 3.5) - (7.5, 4.5)
Minimum distance: 1.8027756377319946
```

TIME COMPLEXITY: O(n^2)