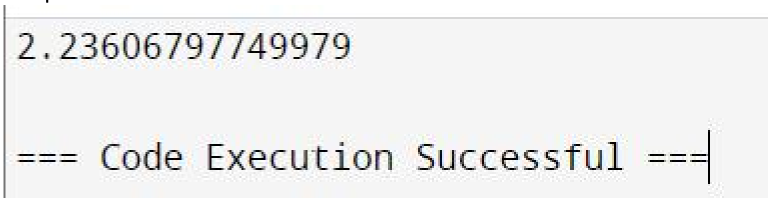
85. Closest pair of points using divide and conquer

Program:

```
import math

def closest_pair(points):
    points.sort(key=lambda x: x[0])
    return min_distance(points)

def min_distance(points):
    n = len(points)
    if n <= 3:
        return brute_force(points)

    mid = n // 2
    left = points[:mid]
    right = points[mid:]

    min_left = min_distance(left)
    min_right = min_distance(right)
    min_dist = min(min_left, min_right)

    strip = [point for point in points if abs(point[0] - points[mid][0]) < min_dist]
    strip.sort(key=lambda x: x[1])

    return min(min_dist, strip_distance(strip, min_dist))

def brute_force(points):
    min_dist = math.inf
    for i in range(len(points)):
        for j in range(i+1, len(points)):
            dist = math.dist(points[i], points[j])
            min_dist = min(min_dist, dist)
    return min_dist

def strip_distance(strip, min_dist):
    min_strip = min_dist
    for i in range(len(strip)):
        j = i + 1
        while j < len(strip) and (strip[j][1] - strip[i][1]) < min_strip:
            min_strip = min(min_strip, math.dist(strip[i], strip[j]))
            j += 1
    return min_strip

# Example Usage
points = [(1, 1), (2, 3), (5, 4), (6, 7), (8, 9)]
print(closest_pair(points))
```

Output:



```
2.23606797749979


=== Code Execution Successful ===
```

Time complexity:O(nlogn)