78.Closest-Pair

Program:

```
import math
def distance(p1, p2):
   return math.sqrt((p1[0] - p2[0]) ** 2 + (p1[1] - p2[1]) ** 2)
def closest_pair(points):
   def closest_pair_rec(px, py):
      n = len(px)
      if n <= 3:
         return min((distance(px[i], px[j]), (px[i], px[j])) for i in range(n) for j in range(i + 1, n))
      mid = n // 2
      mid_x = px[mid][0]
      Qx, Rx = px[:mid], px[mid:]
      Qy, Ry = [p for p in py if p[0] <= mid_x], [p for p in py if p[0] > mid_x]

      (dl, pair_l) = closest_pair_rec(Qx, Qy)
      (dr, pair_r) = closest_pair_rec(Rx, Ry)
      d, mn_pair = (dl, pair_l) if dl < dr else (dr, pair_r)

      strip = [p for p in py if mid_x - d <= p[0] <= mid_x + d]
      min_strip = min(((distance(strip[i], strip[j]), (strip[i], strip[j]))
                       for i in range(len(strip)) for j in range(i + 1, min(i + 7, len(strip)))),
                      default=(d, mn_pair))

      return min_strip if min_strip[0] < d else (d, mn_pair)

   px = sorted(points, key=lambda p: p[0])
   py = sorted(points, key=lambda p: p[1])
   _, pair = closest_pair_rec(px, py)
   return pair


# Example usage:
points = [(2, 3), (12, 30), (40, 50), (5, 1), (12, 10), (3, 4)]
closest_points = closest_pair(points)
print(f"Closest pair: {closest_points} with distance {distance(*closest_points)}")
```

Output:

```
"C:\Program Files\Python312\python.exe" "C:\Work Space\DAA\DAA COADS.PYTHON\program 78.py"
Closest pair: ((2, 3), (3, 4)) with distance 1.4142135623730951

Process finished with exit code 0
```

Time complexity:
O(n.m)