

Lab 4

Student name: Khalid Nimri (2140145)

Exercise 1: (Distance between closest pair of points) using divide and conquer.

The complete code:

```
import math

def distance(p1, p2):
    """
    Returns the Euclidean distance between two points.
    """
    return math.sqrt((p1[0] - p2[0])**2 + (p1[1] - p2[1])**2)

def brute_force(points):
    """
    Finds the closest pair of points using brute force.
    """
    n = len(points)
    min_dist = float('inf')
    for i in range(n-1):
        for j in range(i+1, n):
            dist = distance(points[i], points[j])
            if dist < min_dist:
                min_dist = dist
    return min_dist

def closest_pair(points):
    """
    Finds the closest pair of points using divide and conquer.
    """
    n = len(points)
    if n <= 3:
        return brute_force(points)

    # Divide the points into two halves along the x-axis
    mid = n // 2
    left_points = points[:mid]
    right_points = points[mid:]

    # Recursively find the closest pair of points in each half
    left_min = closest_pair(left_points)
    right_min = closest_pair(right_points)

    # Find the minimum distance between the two halves
    min_dist = min(left_min, right_min)
```

```

# Find the points within the strip
strip = []
for i in range(n):
    if abs(points[i][0] - points[mid][0]) < min_dist:
        strip.append(points[i])

# Find the minimum distance between points in the strip
strip_min = brute_force(strip)

return min(min_dist, strip_min)

# Example usage
points = [(3, 2), (3, 6), (7, 12), (3, 1), (9, 18), (8, 56)]
min_distance = closest_pair(points)
print("The minimum distance between any two points is:", min_distance)

```

PART B: Run your code for the following inputs: $p = [(3, 2), (3, 6), (7, 12), (3, 1), (9, 18), (8, 56)]$; and **attach screenshot of final** output of your code

The screenshot shows the OnlineGDB IDE interface. The code is written in Python and is being executed. The output of the program is displayed in the console at the bottom.

```

39 # Find the minimum distance between the two halves
40 min_dist = min(left_min, right_min)
41
42 # Find the points within the strip
43 strip = []
44 for i in range(n):
45     if abs(points[i][0] - points[mid][0]) < min_dist:
46         strip.append(points[i])
47
48 # Find the minimum distance between points in the strip
49 strip_min = brute_force(strip)
50
51 return min(min_dist, strip_min)
52
53 # Example usage
54 points = [(3, 2), (3, 6), (7, 12), (3, 1), (9, 18), (8, 56)]
55 min_distance = closest_pair(points)
56 print("The minimum distance between any two points is:", min_distance)
57
58
59

```

Input

```

The minimum distance between any two points is: 1.0

...Program finished with exit code 0
Press ENTER to exit console.

```

The minimum distance between any two points is: 1.0