Problem 2. Closest Pair of Points using Divide and Conquer algorithm

Given an array of n points in the plane, and the problem is to find out the distance of closest pair of points in the array. For example, in air-traffic control, you may want to monitor planes that come too close together, since this may indicate a possible collision. Recall the following formula for distance between two points p and q. $||\mathrm{pq}|| = \sqrt{(\mathrm{p_x} - \mathrm{q_x})^2 + (p_y - q_y)^2}$. The Brute force solution is $O(\mathrm{n}^2)$, compute the distance between each pair and return the smallest. We can calculate the smallest distance in $O(\mathrm{n} * \log(\mathrm{n}))$ time using Divide and Conquer strategy. Complete the function "def closest_util" for the function "def closestPair(points: list[tuple[int, int]]) -> float". Decimal numbers will be rounded at the third decimal place. For example, 11.2226 would be rounded to 11.223, and 11.2224 would be rounded to 11.222.

Example:

Input: points = [(0, 0), (1, 1), (2, 2), (5, 5), (1, 2)]

Output = closestPair(points)

Output: 1.0

Constraints:

0 <= the range of each axis <= 1,000

0<= len(points) <= 100

Submission guidelines for Gradescope are as follows:

1. Do Not Rename Files: Do not change the file names when submitting.
2. Do Not Change Function Names: Keep the original function names as specified.
3. Follow Input and Output Formats Accurately: Ensure that the input and output formats match the specified requirements exactly.