Input: List P of points (x, y)
Output: A pair of 2 points that have the smallest distance
Note: Assume we have some distance function $d(x, y)$ which returns the distance between points x and y. An example of such function would be the Euclidean distance.

procedure CLOSEST_PAIR(P)
    $P_x \leftarrow$ list of points in P sorted by x coordinate
    $P_y \leftarrow$ list of points in P sorted by y coordinate
    return $Closest\_Pair(P_x, P_y)$

procedure CLOSEST_PAIR(P_x, P_y)
    if $|P_x| \leq 3$ **then** ▷ For the base case, we will use brute force
        ▷ Yields constant time, since $|P_x|$ is very small
        return $Closest\_Pair\_Brute\_Force(P_x)$

    $mid \leftarrow \lfloor |P|/2 \rfloor$
    $L_x \leftarrow$ elements in index range $[1, mid]$ from P_x
    $L_y \leftarrow$ elements from L_x sorted by y coordinate (construct from P_y)
    $R_x \leftarrow$ elements in index range $[mid + 1, |P_x|]$ from P_x
    $R_y \leftarrow$ elements from R_x sorted by y coordinate (construct from P_y)

    $(l_1, l_2) \leftarrow Closest\_Pair(L_x, L_y)$
    $(r_1, r_2) \leftarrow Closest\_Pair(R_x, R_y)$

    $x_{mid} \leftarrow P[mid]$
    $\delta \leftarrow \min(d(l_1, l_2), d(r_1, r_2))$
    $C \leftarrow$ points from P with $|x_{mid} - x| < \delta$, sorted by y coordinate (construct from P_y)

    $c \leftarrow (l_1, l_2)$
    if $d(r_1, r_2) = \delta$ **then**
        $c \leftarrow (r_1, r_2)$

    for $i \leftarrow 0$ **to** $|C| - 1$ **do** ▷ The inner loop will actually only check at most 6 points (proven, but not in this document), so we have constant time for the inner loop.
        $j \leftarrow i + 1$
        while $j < |C|$ AND $((C[j].y - C[i].y) < \delta)$ **do**
            if $d(C[j], C[i]) < \delta$ **then**
                $\delta \leftarrow d(C[j], C[i])$
                $c \leftarrow (C[i], C[j])$

return c

```
procedure CLOSEST_PAIR_BRUTE_FORCE(P)
    d ← ∞
    c ← UNDEFINED
    for p1 ∈ P do
        for p2 ∈ P ≠ p1 do
            d12 ← d(p1, p2)

            if d12 < d then
                d ← d12
                c ← (p1, p2)
    return c
```