Pseudocódigo Closest Pair of Point

```
vectur<tuple<x1, y1, x2, y2, dist>>> cpp (vector<pair<int,int>> p)

    p.sort(x) // Ordenar arreglo por coordenada x

    int n = (p.size() / 2) // Valor medio

    vector<pair<int, int>> l = p[0, n]
    vector<pair<int, int>> r = p[n+1, p.size() - 1]

    // Comparar coordenadas de cada lado
    vector<tuple<int, int, int, int, dist>> d_l = comparar_coord(l)
    vector<tuple<int, int, int, int, dist>> d_r = comparar_coord(r)

    // Margen para el centro
    int m = p.size() / 3 // Divide en 3

    // Tomamos 2/3
    vector<pair<int,int>> mid = p[m, m+n]
    vector<tuple<int,int,int,int, dist>> d_mid = comparar_coord(mid)

    // Encontrar el segmento mínimo
    int min = ∞

    if (d_l.dist < min) min = d_l.dist
    if (d_r.dist < min) min = d_r.dist
    if (d_mid.dist < min) min = d_mid.dist

    // Almacenar coordenadas mínimas

    vector<tuple<int, int, int, int, dist>> res

    for (int i=0 ; i< d_l.size() ; i++) {
        if (d_l[i] == min) res.push_back(d_l[i])
    }

    for (int j=0 ; j < d_r.size(); j++ ) {
        if (d_r[i] == min) res.push_backs(d_r[j])
    }

    for (int k=0; k < mid ; k++) {
        if (mid[k] == min) res.push_backs(mid[k])
    }

    return res;
}
```

```
vector<tuple> comparar coord (vector<tuple> v) {

    int min = INT_MAX
    vector<tuple> res

    for (i=0 ; i<= v.size()-2) {

       for (j=i+1 ; j<= v.size()-1) {

            d = sqrt ((v[i][0]- v[j][0])^2 + (v[i][1]- v[j][1])^2)

            if (d< min) min = d
                res.clear()
                res.push_backs(v[i], v[j], d]

            } else if (d == min)
                res.push_backs (v[i], v[j], d]
            }
       }
    }

    return res;
}
```