SHUBHAM GUPTA
180749
Theo Ass - 1

7) pseudocode:

```
sort (A)
sort (B)
min dist ← +∞
for (i, m-1) {
    start ← 0
    end ← m-1
    mid ← ((start + end) / 2)

    if ( dist (A[mid], B[i]) < min dist) {
        min dist ← dist (A[mid], B[i])
        ans ← (A[mid], B[i])
    } else if ( dist (A[mid], B[i]) > dist (A[mid - 1], B[i]))
        start = mid - 1
    } else {
        end = mid + 1
    }
    if ( min dist > dist A[mid], B[i]) {
        min dist ← dist (A[mid], B[i])
        ans ← (A[mid], B[i])
    }
}
return ans.

dist ( A[], B[]) {
    return (xA, yA) . (xB, yB)
}
```

algorithm: we will sort A & B, and then for every point in B we will find particular point in A such that distance by A[j] & B[i] is minimum, this could be done by dividing and conquer. we do this for all values of B.

time complexity : $O(m \times \log n)$

Proof of correctness:
after i^{th} iteration optimal sol^n should lie b/w $start_i$ & end_i

after i-1 iteration let $mid = \frac{start + end}{2}$
we will now have 3 cases.
C1: dist (A[mid], B[i]) > dist (A[mid-1], B[i])
we will move to the left half of mid to find minima.
C2: dist (A[mid], B[i]) > dist (A[mid+1], B[i])
we will move to the right half of mid to find minima
C3: if both RHS distances are greater than dist (A[mid], B[i])
the sol^n is found.