⓪ Closest points $\boxed{\text{Geometric algorithms}}$

Given: n points c[i] with cordinates
c[i].x , c[i].y $0 \le i \le n-1$

Find: points i, j

Such that: distance (c[i], c[j]) is minimized over all other pairs of points

simplify: find the min distance.

Question: can we do beter than n^2

$\boxed{\text{ToDo}}$ write the n^2 algorithm:

Geometric algorithms

① Try divide and conquer

To Do closest Distance(point[] c, int low, int high)

? base case:

problem decomposition:

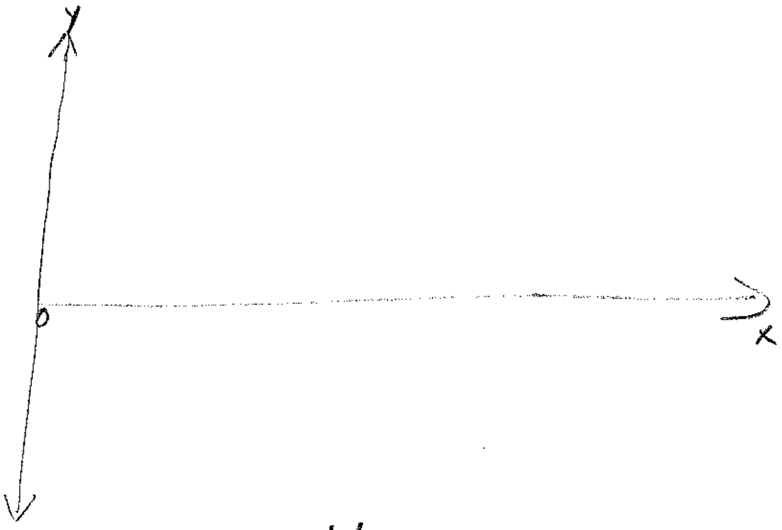
ⓐ just split $c[0 \ldots n/2-1]$ $c[n/2 \ldots n-1]$

To Do write the recurrance relation:

ⓑ sort by x first?

To Do draw the points

WLOG
assume
$\forall_i \; c[i].x > 0$



To Do draw the split

② Now consider solution construction

```
left ← closestDistance(C, low, high/2
right ← closestDistance(C, high/2+1, high)
```

my answer is? to Do

Complexity? write the recurrance ToDo

Is this algorithm correct?
what points need to be considered?
ToDo draw Principle: "compute once use many times"

(3) Complexity?

write the recurance relation ToDo

new method

Can we do beter?

sort by Y

linear scan? how many other points?

Draw the packing density TOdo

(4) what is the recurrance relation? To do

Final optimization:

principle "apply what you have already done
to do less"

Merge sort!

Final algorithm:

Final complexity

Expected performance graph

Book keeping: