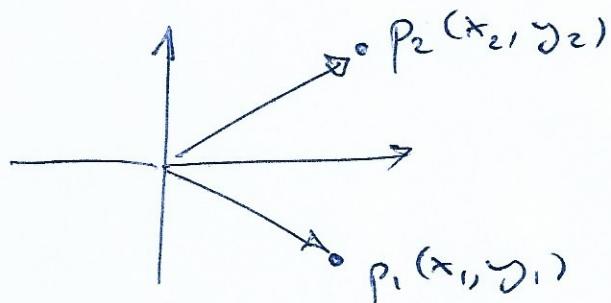


33.1-1

We can separate the proof into a few cases:
case 1: Both \vec{p}_1 and \vec{p}_2 are in the right half-plane:



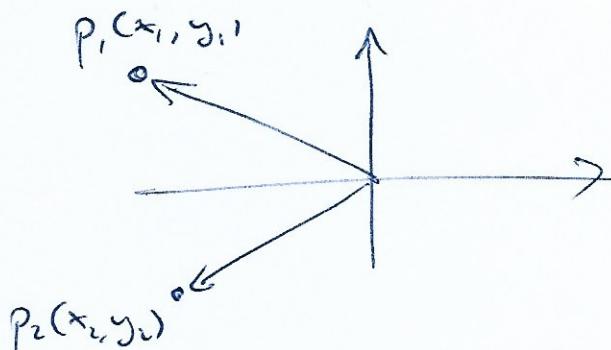
\vec{p}_1 is clockwise from \vec{p}_2

\Leftrightarrow gradient of $\vec{p}_1 <$ gradient of \vec{p}_2

$$\Leftrightarrow \frac{y_1}{x_1} < \frac{y_2}{x_2}$$

$$\Leftrightarrow x_1 y_2 - x_2 y_1 > 0 \quad (\text{Note: } x_1, x_2 > 0)$$

case 2: Both \vec{p}_1 and \vec{p}_2 are in the left half-plane.



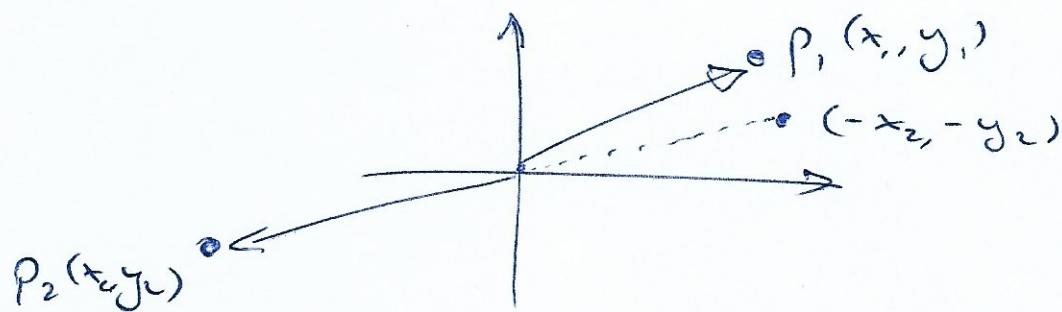
\vec{p}_1 is clockwise from \vec{p}_2

\Leftrightarrow gradient of $\vec{p}_1 <$ gradient of \vec{p}_2

$$\Leftrightarrow \frac{y_1}{x_1} < \frac{y_2}{x_2}$$

$$\Leftrightarrow x_1 y_2 - x_2 y_1 > 0 \quad (\text{Note: both } x_1, x_2 < 0 \\ \text{so multiplying reverses the inequality twice.})$$

Case 3: p_1 is in right half-plane and p_2 is in left half-plane.



\vec{p}_1 is clockwise from \vec{p}_2

$\Leftrightarrow \vec{p}_1$ is ^{counter-}clockwise from $(-x_2, -y_2)$

\Leftrightarrow gradient $\vec{p}_1 >$ gradient of $(-x_2, -y_2)$

$$\Leftrightarrow \frac{y_1}{x_1} > \frac{-y_2}{-x_2}$$

$$\Leftrightarrow x_1 y_2 - x_2 y_1 > 0 \quad \square$$

Case 4 p_1 is in left half-plane and p_2 is in right half-plane.

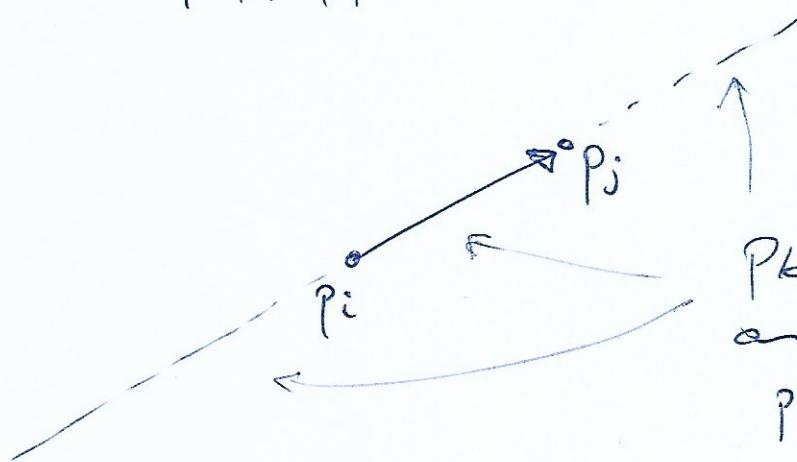
This is similar to case 3.

| There are also special cases where p_1 and/or p_2 lie on the x - or y -axis.

33.1 - 2

On-Segment (p_i, p_j, p_k) is used if

$\text{Direction}(p_i, p_j, p_k) = 0$, as in :



p_k can lie anywhere
on the line through
 p_i & p_j .

p_k lies on the segment $p_i p_j$ iff

$$x_i \leq x_k \leq x_j \quad \& \quad y_i \leq y_k \leq y_j$$

In this case it is enough to just check
for x 's or just for y 's.

But consider the case where $\vec{p_i p_j}$ is
vertical : and p_k lies below $\vec{p_i p_j}$



The ~~condition~~

$$x_i = x_k = x_j \quad \text{so}$$

$$\min(x_i, x_j) \leq x_k \leq \max(x_i, x_j)$$

is satisfied.

So we have to check for
 $\min(y_i, y_j) \leq y_k \leq \max(y_i, y_j)$

33.1-3 A is an array of points $P_i = (x_i, y_i)$.

Mergesort(A, p, r)

If $p < r$

$$g = \lfloor \frac{p+r}{2} \rfloor$$

MergeSort(A, p, g)

MergeSort($A, g+1, r$)

Merge(A, p, g, r)

==

Merge(A, p, g, r)

Create arrays L & R with elements from
 $A[p, \dots, g]$ and $A[g+1, \dots, r]$.

$i = 1$

$j = 1$

for k from p to r

if point at $L[i]$ lies clockwise from point at $R[j]$
or the lie in same direction from P_0 ,
but $L[i]$ is closer to P_0

then $A[k] = L[i]$

$i = i + 1$

else $A[k] = R[j]$

$j = j + 1$

==

33.1-3

Note: In \otimes we are checking if
the point at $L[i]$ comes before
the point at $R[j]$ in the counter-clockwise
ordering.

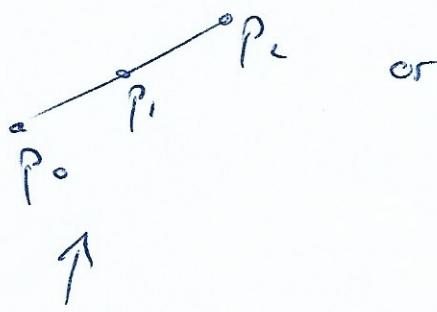
If point at $L[i]$ is $p_1 = (x_1, y_1)$
& point at $R[j]$ is $p_2 = (x_2, y_2)$

we calculate $p_1 \times p_2 = x_1 y_2 - x_2 y_1$,

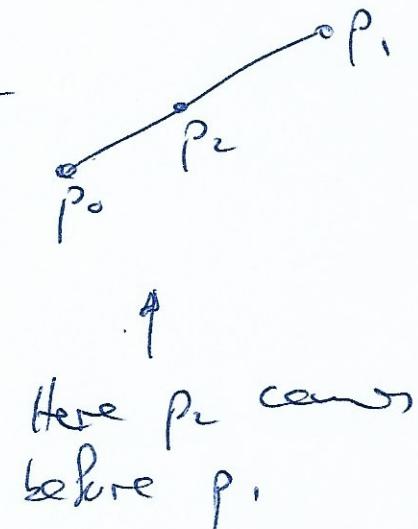
if $p_1 \times p_2 > 0$ then p_1 is clockwise from p_2
if $p_1 \times p_2 < 0$ then p_2 is clockwise from p_1

if $p_1 \times p_2 = 0$ then we have the

situation



Here p_1 comes
before p_2



Here p_2 comes
before p_1

To determine which situation occurs
we can calculate distance from p_0 to p_1
and p_0 to p_2 . Or you can determine
on-Segment(p_0, p_1, p_2) and on-Segment(p_0, p_2, p_1).