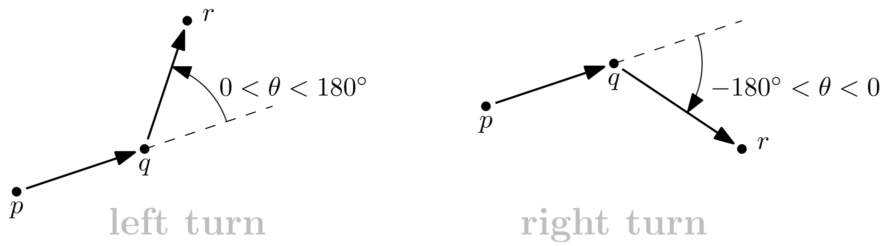
T1. **(24-July-2018)[Turn checking]** Explain how to check whether three points on xy-plane make a right turn. Prove that your procedure always gives computationally correct result when the points have integer coordinates.



Solution See the above figure. Let p, q, r be three points in order to define a turn at q. Take the 2D vectors $\vec{pq}$ and $\vec{qr}$, and compute their cross product $\vec{r} = \vec{pq} \times \vec{qr}$. The vector $\vec{r}$ is a vector perpendicular to xy-plane. If $\vec{r}$ is directed along $+z$-axis, then we have a left turn, otherwise it's a right turn. The calculations are as follows.

$$\begin{aligned}\vec{r} &= \Big(\big(x_q - x_p\big)\hat{\imath} + \big(y_q - y_p\big)\hat{\jmath}\Big) \times \Big(\big(x_r - x_q\big)\hat{\imath} + \big(y_r - y_q\big)\hat{\jmath}\Big)\\ &= \Big(\big(x_q - x_p\big)\big(y_r - y_q\big) - \big(y_q - y_p\big)\big(x_r - x_q\big)\Big)\hat{k}\\ &= d\hat{k},\ \text{where } d = \big(x_q - x_p\big)\big(y_r - y_q\big) - \big(y_q - y_p\big)\big(x_r - x_q\big).\end{aligned}$$

If $d < 0$, then $\vec{r}$ is directed along $-z$-axis, and we get a right turn.

T2. **(24-July-2018)[Convex hull]** Prove that $O(n \log n)$ is optimal for convex hull computation.

Solution Let $A[1..n]$ be a 1D array having n real numbers, namely $a_1, a_2, \ldots, a_n$. Construct a 2D point set $P = \{p_1, p_2, \ldots, p_n\}$ such that $p_i = (a_i, a_i^2)$ for $i = 1, 2, \ldots, n$. Let there be an algorithm AlgoCHull that can compute the convex hull of a 2D point set in less than $O(n \log n)$ time. As all points of P are lying on a parabola—a convex curve, the vertices of the convex hull of P will be all these n points. As a result, these n points of P will be reported by AlgoCHull in anticlockwise order as the vertices of the convex hull of P. We will consider the x-coordinates in that order and hence get the sorted sequence of A in less than $O(n \log n)$ time, which means sorting is doable in less than $O(n \log n)$ time—a contradiction.

T3. **(6-Aug-2018)[k-regular]** Prove that any k-regular bipartite graph always admits a perfect matching.

Solution Let (X, Y) be the vertex-set partition of V in $G(V, E)$, i.e., $x \in X$ and $y \in Y\ \forall\ (x, y) \in E$. As G is k-regular, $|X| = |Y|$. Convert G to a flow network $G'(V', E')$ with the newly added source s and sink t, $V' = V \cup \{s, t\}$, $E' = E \cup \{(s, x) : x \in X\} \cup \{(y, t) : y \in Y\}$, $c(u, v) = 1\ \forall\ (u, v) \in E'$. Apply a flow as follows: $f(s, x) = 1\ \forall\ x \in X$, $f(y, t) = 1\ \forall\ y \in Y$, $f(x, y) = 1/k\ \forall\ (x \in X, y \in Y)$. It is easy to verify that f is a flow, as it satisfies the capacity constraint and flow conservation. Clearly, the value of this flow is $|f| = \sum\limits_{x \in X} f(s, x) = |X|$, and hence it is the max-flow. Its value indicates a perfect matching, since its cardinality is $|V|/2 = |X|$.