Given a graph $G = (V, E)$, a triangle is a set of three vertices $\{u, v, w\}$ that are pairwise adjacent, that is, such that the edges $(u, v)$, $(u, w)$, and $(v, w)$ all belong to $E$. Develop an algorithm that tests whether a given connected graph $G$ contains a triangle and, if so, outputs it. The running time of the algorithm should be in $O(nm)$. Your algorithm should not use any data structures other than the adjacency list representation of $G$ and possibly arrays, doubly linked lists and/or an adjacency matrix representation of $G$. You may use any sorting algorithm as a black box and, for the second bonus question below, use the fact that $n$ integers in the range $[1, n]$ can be sorted in $O(n)$ time. The input is given in adjacency list representation. Prove that your algorithm is correct and that it achieves the desired running time.

**Bonus 1:** Prove that the running time of your algorithm is in fact in $O(dm)$, where $d$ is the maximum vertex degree in $G$. (This is better than $O(nm)$ for all graphs with $d \in o(n)$). Of course, this requires that you have the right algorithm.

**Bonus 2:** Prove that the running time of your algorithm is in fact in $O(m \cdot \min(d, \sqrt{m}))$, where $d$ is the maximum vertex degree in $G$. (This is better than $O(nm)$ for all graphs with $m \in o(n^2)$). Of course, this requires that you have the right algorithm.

Notes:

- If you do the bonus version, choose only one of them. Obviously, Bonus 2 simply asks you to prove a stronger claim than Bonus 1, which in turn asks you to prove a stronger claim than the vanilla version of the assignment. You won’t get any marks for Bonus 1 if you also do Bonus 2.
- Don’t overcomplicate things. The basic strategy of your algorithm should be the natural one: Given an edge, test efficiently whether its two endpoints share a neighbour. Do this for every edge.

Marks:

- 3 marks for a correct algorithm.
- If the algorithm is correct, then 3 marks for achieving the desired running time.
- 2 marks for a correct proof of correctness of the algorithm.
- 2 marks for a correct analysis of its running time. If you do Bonus 1, this becomes 4 marks. If you do Bonus 2, this becomes 6 marks. Note that you will get these 2 marks even if the running time of your algorithm exceeds $O(nm)$, as long as you correctly recognize this fact and provide a correct analysis of your algorithm.