P1) Amy, a TA in 421 is asked to grade n sheets of exams, $1, \dots, n$. Let t_i be the time that takes her to grade the i-th sheet (perhaps, t_i depends on how long the i-th proof is). Whenever she grades the i-th sheet the grade will be published in Gradescope right away. Say she finishes grading i-th sheet at time f_i. Then, the average grading time of all exams is $\frac{1}{n}(f_1 + \cdots + f_n)$. To make the students happy, Shayan asked Amy to minimize the average grading time, that is the average time for a 421 student to see his/her grade. We want to help Amy by figuring out the optimal order to grade these sheets. Design an efficient algorithm which outputs the minimum average grading time (note that your algorithm just needs to output a number).

For example, if $t_1 = 3, t_2 = 2, t_3 = 4$ then the optimal order to grade is, $2, 1, 3$. Then, we have $f_2 = 2, f_1 = 2 + 3 = 5, f_3 = 2 + 3 + 4 = 9$ and the average grading time is $16/3$.

P2) You are given a connected undirected weighted graph $G = (V, E)$ with n vertices and m edges, their weights, w_e for all edges $e \in E$, a minimum spanning tree $T \subseteq E$ together with an edge $f \in E$ and a new weight w'_f for f, design an algorithm that runs in time $O(n + m)$ to test if T still remains the minimum spanning tree of the graph if we change the weight of f to w'_f. Your algorithm should output "yes" if T is still the MST and "no" otherwise. Note that the edge f may or may not belong to T and we may decrease or increase weight of f. For simplicity, assume that all edge weights are distinct both before and after the update. For example in the following picture the MST is colored in blue. If we update the weight of the edge $(3, 1)$ to 1.5 the blue tree is no longer the MST.