Question 5 by Dan Nguyen (z5206032)

There is a given array, A, of size n, which stores the number of weeks, $A[i] = t_i$ required to complete course i.

There is another given array, B, of size m, where stores a pair of *unordered* pairs of *distinct* courses, p, and number of weeks, t_p, required to complete either course if one has been completed i.e. for some b in B:

$$b = \{p, t_p\}$$

$$p = \{i, j\}, t_p \leq \min(t_i, t_j)$$

Construct a weighted undirected graph, G, represented as an adjacency list with $|E|$ edges and $|N|$ nodes. Nodes are each course in A and edges are p for each b in B with the weighting of t_p, therefore $|E| = m$ and $|N| = n$. Graph construction has an expected time complexity of $O(|N| + |E|) = O(n + m)$.

To complete all n courses with the minimal number of weeks, use Kruskal's algorithm to find the minimum spanning forest (MSF) of G which returns a list of edges, E_i, for each minimum spanning tree (MST), i, within the MSF. This has an expected time complexity of $O(|E|log|V|) = O(mlog(n))$.

For each i, find the node with the minimum number of weeks required to complete any course i.e. M_i, and sum E_i for S_i. The minimum number of weeks required to complete all n courses is thus the sum of all M_i and S_i in the MSF. Across all MSTs, this has an expected time complexity of $O(n)$.

The final expected time complexity is therefore:

$$O(n + m) + O(mlog(n)) + O(n) \in O((n + m)log(n + m))$$

as required.