1 True. e^* would be the 1st edge considered by Kruskal's algorithm after sorting.

2 a) True. If we feed the costs c_e^2 into Kruskal's algorithm, it will sort them in the same order, and hence put the same subset of edges in the MST.

b) False. After squaring the costs of all the edges, the shortest path may change.

3 Say n boxes arrive in the order $b_1, \ldots, b_n$.
Say each box b_i has a positive weight w_i, and the maximum weight each truck can carry is W.
To pack the boxes into K trucks preserving the order in which the boxes arrived is to assign each box to one of the trucks $1, \ldots, K$ so that:

- No truck is overloaded;
- The order of arrivals is preserved.

Our Greedy solution stays ahead. We prove this by considering any other solution and show the following.
If the greedy algorithm fits boxes $b_1, \ldots, b_J$ into the first K trucks, and the other solution fits $b_1, \ldots, b_i$ into the first K trucks, then $i \leq J$.

We will prove this claim by induction on K.
The case $K = 1$ is clear; the greedy algo fits as many boxes as possible into the first truck.

Now, assuming it holds for $k-1$:
the greedy algo fits j' boxes into the first $k-1$ trucks, and the other solution fits $i' \leq j'$.
Now, for the k^{th} truck, the alternate solution packs in $b_{i'+1}, \ldots, b_i$. Thus, since $j' \geq i'$, the greedy algo is able at least to fit all the boxes $b_{j'+1}, \ldots, b_i$ into the k^{th} truck, and it can potentially fit more.

4

```
i = j = 1

while i ≤ n and j ≤ m
    if S_i == S'_j
        k_j = i
        i += 1 and j += 1
    else
        i += 1
endwhile
    if j = m + 1
        return S ⊆ S'
```

O(n)

5 Push the 1st base station as right as possible and remove the covered houses. Repeat.
We can prove the optimality (greedy stays ahead) by induction and by exchange argument, comparing the set of base stations placed by our antenna, to the set of base stations placed by any other solution.

6 We can arrange the participants in order of decreasing (cycling + running) time, and send them out in this

order. (greedy stays ahead)

We can prove the optimality of our greedy rule by exchange argument, considering any other solution which sends the partecipants out in another order (maybe in order of increasing time).

- Partecipants numbered from 1 to n;
- S_i, B_i, R_i times.

Consider an OPT solution that, as previously said, sends them out in another order.
For example: a pair of 2 partecipants (i, j).
j is being sent out after i, but $B_i + R_i < B_j + R_j$.
This is an inversion.

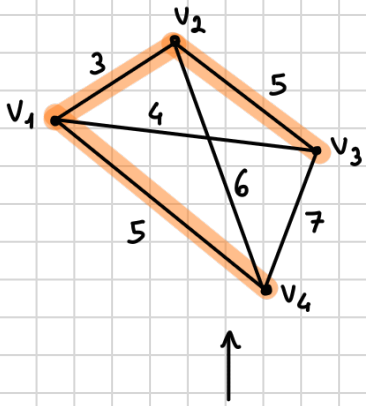
7 Run jobs in order of decreasing time.
We can prove the optimality of our greedy solution (greedy stays ahead) by exchange argument.
Consider another solution $\neq$ from ours and we say that if we swap jobs that form a pair, because actually they're in an increasing running time order, then, after making all the possible inversions of this solution, it won't have a completion time better or worse than our solution, so our greedy solution is OPT.

8 Let's do a contradiction.
Suppose that T and T' are 2 distinct MST of G.
Since T and T' have the same n. of edges, but are not equal, there's some edge e' in T' but not in T.

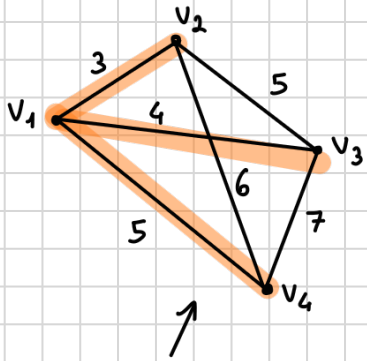
If we add T' to T, we form a cycle C.
Let e be the most expensive edge of this cycle.
Then, by the cycle property, e doesn't belong to any MST.

9 a) False.

minimum bottleneck tree



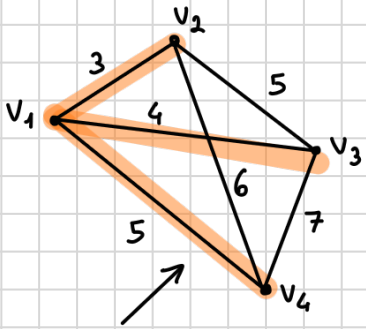
This is a min. bottleneck tree, but not a MST.



This a MST and also a min. bottleneck tree.

In my opinion the solution of this exercise is wrong.

b) True



This is the unique MST of this graph, and it's for sure also a min. bottleneck tree.

11 Let δ be the minimum difference between any two non-equal edge weights; subtract $\delta i/n^3$ from the weight of edge i. Then Kruskal's algo will sort the edges and return the MST, between the ST

whose total weight has been reduced by the most.

13 An optimal algorithm is to schedule jobs in decreasing order of w_i / t_i.
We can prove the optimality of this algo by an exchange argument (inversions).

15 – 1st the one that ends earliest and unmarked.
– 2nd pick one that overlaps with 1st and ends latest. Add this to the set of solutions and mark the other shifts that overlap with this.

17

22 We can't conclude that T itself must be a MST in G.