POKLON

Let $a_1, a_2, ..., a_{N-1}$ be the prices of the gifts in decreasing order. Set the phony price to 0 and calculate the difference between the costs of gifts of both grandchildren. Now increase the phony price by 1 until it is equal to a_{N-1}.

Notice that the difference also changes by 1. The difference will decrease if N is even, and increase if N is odd. Considering phony prices in the interval $[0, a_{N-1}>$ we can calculate the exact interval of possible values for the difference. The intersection of this interval with the interval [A, B] gives the interval in which both grandchildren are happy, so we add the size of this intersection to the output.

Repeat this procedure for the interval $[a_{N-1}, a_{N-2}>$ as possible phony prices. This price is larger than that of the last gift so it will be chosen second to last. Because of this, the difference will now change opposite of how it did in the first case, again depending on whether N is even or odd.

We repeat the same procedure until the interval $[a_0, \infty>$ when the gift with the phony price tag becomes the most expensive.

TVRTKA

There are multiple ways to assign a new command structure satisfying all three conditions. We will describe one here.

For each employee E in the initial command structure, sort the list of his direct reportess in decreasing order of IQ. The first employee keeps E as his manager, while every other reportee gets its predecessor in the list as the new manager.

Every team has at most 3 members because every employee has at most 2 direct reportees, one he managed in the initial structure and one which was his peer, so the first condition is satisfied. The second condition is satisfied because only the first of these two can have an IQ greater than his manager. Finally, the third condition is satisfied, which is obvious from the way in which we created the structure.

SLON

Sort the plants in decreasing order of x-coordinates. Plants with equal x-coordinates are sorted in decreasing order of y-coordinates. In the sorted sequence, from some plant it is only possible to jump to plants that come later in the sequence. This allows us to use dynamic programming to solve the problem.

We proceed along the sorted sequence, and for each plant we calculate the length of the longest sequence of jumps ending on that plant. Let dp(i) be that length. Let L(a) be the list for all plants for which dp(i) = a. Let L(a).last be the last plant in L(a).

Observe that, as the list L(a) grows, new plants are added in decreasing order of y-coordinates. If that were not so, it would be possible to jump from one plant to another an obtain a sequence of jumps of length a+1.

Also, if we observe the plants L(a).last in increasing order of a, we will notice that their y-coordinates increase with a.

The number dp(i) can be calculated by choosing, of all plants L(a).last, the one with the largest y-coordinate not greater than the y-coordinate of plant i. Let that plant be $L(a_0)$.last. Because the y-

coordinate increases with a, the number a_0 can be efficiently found with binary search. From this plant it is possible to jump to plant i, so $dp(i) = a_0 + 1$. We add plant i to the end of list $L(a_0 + 1)$.

To calculate the number of different sequences of length dp(i) ending on plant i, we need to find all plants in the list $L(a_0)$ from which it is possible to jump to plant i. Because the list $L(a_0)$ is sorted in decreasing order of y-coordinates, all such plants will be at the end of $L(a_0)$ and we can find the first such plant again using binary search.