Fancy Fence Solution

Topics

Sorting, (DSU), Maths, Sieve

Problem author

Máté Busa

Subtask 2

$N, h_i \leq 50, w_i = 1$

There are at most 50^4 different rectangles.
For all of them, we can check if they are fancy or not. This can be done in constant time with some precomputation.

Subtask 3

$h_i = 1$ or $h_i = 2$ for all i.

Consider a rectangle with height 1 and width K.
Lemma: There are $\binom{K+1}{2}$ fancy rectangles in it.
Proof: There are $K - p + 1$ fancy rectangles with width p:

$$\sum_{p=1}^{K}(K - p + 1) = \binom{K+1}{2}$$

Now we can solve subtask 3: There are 2 types of fancy rectangles, the ones with height 1, and the ones with height 2.

We can easily calculate the answer, applying the previous lemma in $O(N)$ time.

A helpful observation

Consider a rectangle with height A and width B.
Let's denote the number of fancy rectangles contained within this big rectangle by $T_{A,B}$. Now we have

$$T_{A,B} = \binom{A+1}{2} \cdot \binom{B+1}{2}$$

Proof:
We can chose $\binom{A+1}{2}$ different horizontal lines to form the horizontal side of a fancy rectangle. The same holds for the vertical side.

Note that $\binom{X}{2} = \frac{X(X-1)}{2}$, where $X(X-1)$ is always divisible by 2.

Subtask 4

The solution follows easily from the previous lemma.

This subtask can be solved in $O(N)$ time.

Subtask 5

The heights are in increasing order.

Let W_i be the sum of section widths from the ith to the Nth section.
The answer is given by the formula:

$$\sum_{i=1}^{N} T_{h_i,W_i} - T_{h_{i-1},W_i},$$

where $h_0 = 0$.

This way, the subtask can be solved in $O(N)$ time.

Subtask 6

$N \leq 1000$

For all $1 \leq i \leq j \leq N$, we calculate the number of fancy rectangles whose left side is part of the ith section and right side is part of the jth section.
Let H be the minimum of section heights from the ith to the jth section.
Let W be the sum of section widths from the ith to the jth section.
The number of fancy rectangles is (if $i \neq j$):

$$T_{H,W} - T_{H,W-w_i} - T_{H,W-w_j} + T_{H,W-w_i-w_j},$$

which can be precomputed for all H.

This subtask can be solved in $O(N^2)$.

Subtask 7

Original constraints.

Sorting

Let's sort the sections in decreasing order according to their heights.

Let us denote the original index of the i section by p_i. In the ith step, we calculate the number of fancy rectangles lying exclusively on the first i sections.

Let x be the smallest index for which the xth, $x+1$th ... $p_i - 1$th sections preceed the p_ith section.

Let y be the biggest index for which the $p_i + 1$th, $p_i + 2$th ... yth sections succeed the p_ith section.
Write

$$X_i = \sum_{j=x}^{p_i-1} w_j, \qquad Y_i = \sum_{j=p_i+1}^{y} w_j.$$

In the ith step, the answer increases by $T_{h_{p_i},X_i+Y_i+w_{p_i}} - T_{h_{p_i},X_i} - T_{h_{p_i},Y_i}$. To calculate values X_i and Y_i efficiently, we have to combine consecutive intervals (e.g. using DSU or similar, more simple methods). At the start, there are N intervals, each one consists of exactly one section. In the ith step, we shall combine the sections from the xth to the yth. For all intervals, the sum of widths of sections contained by the interval has to be stored as well.

This subtask can be solved in $O(NlogN)$ time.

Linear

Let's iterate through the sectionss from left to right maintaining a stack of sections with the following property: from bottom to top the height of sections are increasing and after the ith section is processed every fancy rectangle not present in the stack is already counted.

When at the ith section three cases are possible, let the top of the stack contain a section with dimensions $H \times W$:

- if $h_i = H$ we can easily modify W and increase it by w_i not hurting the invariant described above
- if $h_i > H$ we can just push a $h_i \times w_i$ rectangle to the top of the stack
- if $h_i < H$ then we have to pop some elements from the stack until h_i will be greater or equal to the height of the section on the top of the stack. While doing the popping we accumulate the width of the new top element (i.e. the sum of widths of all elements popped plus w_i) and also with a similar strategy to subtask 5 the number of fancy rectangles that will not be present in the stack should be calculated.

After processing every section we can use the solution to subtask 5 to count the remaining fancy rectangles or even better just create a 0×0 dummy section after the ones in the input.

Overall the time complexity of this solution is $O(N)$ since every rectangle is pushed and popped exactly once while doing a constant amount of operations.