Grattis! You've been hired to assist the packaging operations at Ikea. Your first task is to evaluate a bid from a box manufacturer. Roughly speaking, the ideal box manufacturer is one whose boxes can fit as many Ikea products as possible. In particular, you are given a list P of n products $p_1, \ldots, p_n$ where product p_i has length $\texttt{length}(p_i)$ and width $\texttt{width}(p_i)$, and a list B of m boxes $b_1, \ldots, b_m$ where box b_j has length $\texttt{length}(b_j)$ and width $\texttt{width}(b_j)$. We say that product p_i *fits* into box b_j if $\texttt{length}(p_i) \leq \texttt{length}(b_j)$ and $\texttt{width}(p_i) \leq \texttt{width}(b_j)$. (Unfortunately, the packing machine is unable to rotate products due to damage, so no rotations are allowed.) The *total fit* is the total number of product-box pairs (p_i, b_j) such that p_i fits in b_j. Note that this is different from the total number of products that can fit in *at least one* box. The goal is to compute the total fit.

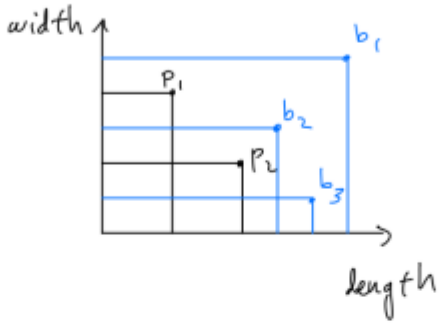

Figure 1: p_1 and p_2 fit in box b_1; p_2 fits in box b_2 so the total fit is 3.

Note that there may be multiple products and/or boxes that share the same length and/or width.

Task 1: Keep It Simple, Single-Dimensional [5 points]

We start with the simple case where products and boxes are one-dimensional, i.e. they only have a length. (Believe it or not, thinking about this is actually useful for the other parts of the assignment.) Here, we are given a list P of n products $p_1, \ldots, p_n$ where product p_i has length $\texttt{length}(p_i)$, and a list B of m boxes $b_1, \ldots, b_m$ where box b_j has length $\texttt{length}(b_j)$. We say that product p_i *fits* into box b_j if $\texttt{length}(p_i) \leq \texttt{length}(b_j)$. The *total fit* is the total number of product-box pairs (p_i, b_j) such that p_i fits in b_j.

Your task is to design an algorithm that computes the total fit in $O(N \log N)$ time where $N = m + n$ and implement your algorithm on Ed. (You are not required to submit a description of the algorithm and proof of correctness and time complexity, but you should also practice describing your algorithm in plain English and proving correctness and time complexity as it will help with Task 2.)

Task 2: Two-Dimensional [95 points]

We can think of the products and boxes as points in two-dimensional space with length as the x-coordinate and width being the y-coordinate. Thus, it is natural to divide the input using a horizontal or vertical line, i.e. divide the products and boxes according to their width or length, respectively. Before deciding on exactly what the line should be, it is useful to design the combine step.

(a) **Combine step** In this subtask, we will design the combine step subroutine assuming that the divide step divides products and boxes into those whose length are at most D and those that are at least D, for some D. Formally, the inputs to the combine step are:

- 4 lists P_L, P_R, B_L, B_R. The list P_L consist of products p_i with $\texttt{length}(p_i) \leq D$ and P_R consist of products p_i with $\texttt{length}(p_i) \geq D$. Ditto for B_L and B_R. Moreover, each list is sorted in ascending order of width,
- F_L, the total fit of products P_L with boxes B_L, i.e. the number of pairs (p_i, b_j) where p_i is in P_L and b_j is in B_L and p_i fits in b_j.
- F_R, the total fit of products P_R with boxes B_R, i.e. the number of pairs (p_i, b_j) where p_i is in P_R and b_j is in B_R and p_i fits in b_j.

The output of the combine step is the total fit of products in both P_L and P_R with all boxes in B_L and B_R, i.e. the number of pairs (p_i, b_j) where p_i is in P_L or P_R, b_j is in B_L or B_R, and p_i fits in b_j. Your task is to design an $O(N)$ time algorithm for the combine step, where N is the total number of products and boxes, i.e. $N = |P_L| + |P_R| + |B_L| + |B_R|$. (Note: You may choose to solve a different version of the combine step. If you choose to do so, make sure you clearly define what the inputs and outputs of your combine step are.)

(i) Description of how your algorithm works in plain English.

(ii) Prove that your algorithm is correct.

(iii) Prove an upper bound on the time complexity of your algorithm.

(iv) Implement your algorithm on Ed

(b) Use the combine step algorithm above to construct a divide-and-conquer algorithm for the problem. For full marks, your algorithm should take time $O(N \log N)$ where $N = m + n$, the total number of products and boxes.

(i) Description of how your algorithm works in plain English. Make sure you describe

- your pre-processing step (if needed)
- the problem your recursive algorithm is solving, i.e. its input and output.
- your divide step (subproblems, base cases)
- your delegate step
- how you use the combine step subroutine

(ii) Prove that your algorithm is correct

(iii) Prove an upper bound on the time complexity of your algorithm

(iv) Implement your algorithm on Ed

Guidelines

To make it easier for you to write and for us to mark, you can simply assume the following without further explanation/proof when describing/analyzing your algorithm.

- Computing the min or max of an unsorted list of n values takes $O(n)$ time. You can simply write "Let A be the max of the input list". Do not tell us the for loop to do this.
- Sorting a list of n values takes $O(n \log n)$ time. You can simply write "Sort list L in ascending order of blah" and assume it takes $O(n \log n)$ time without specifying what sort you are using. Please do not re-write or re-prove mergesort or any other sorting algorithm.
- Check whether a product p_i fits in a box b_j in $O(1)$ time. You can simply write "If p_i fits in b_j then blah" instead of writing "if $\texttt{length}(p_i) \leq \texttt{length}(b_j)$ and $\texttt{width}(p_i) \leq \texttt{width}(b_j)$ then blah".