The Parking Lot Problem

Let's assume that we have a parking lot with fixed dimension, and there is a large variety of cars, busses, motorcycles and various other vehicles that need to be parked on it for long term. In the long term, we do not care where the individual vehicles are, and if they are accessible or not, all that we care about is to fit all of the vehicles on the lot.

1. Introduction

Given a matrix sizes $P \in N^{L \times W}$ which shows, the position of the individual cars on the lot. Let us also have a set of vehicles V= $\{(l_i, w_i)\}_{i=1}^{N}$, where l_i is the length of vehicle i., and w_i is the width of the vehicle i. For example, on a 5x7 sized parking lot, and set of vehicles:

$$V = \{(4,2),(3,2),(1,2),(2,5),(2,2),(2,1),(3,1)\}$$

A possible arrangement of vehicles can be placed as follows:

P=

1	1	1	1	4	4	6
1	1	1	1	4	4	6
2	2	3	3	4	4	7
2	2	5	5	4	4	7
2	2	5	5	4	4	7

Our goal is to fit all object in O into the P matrix. You can rotate vehicles 90 degrees, in fact, in most cases it will be necessary. The problem can easily be demonstrated to be NP hard, yet there still are efficient algorithms that can find a solution reasonably quickly even on large problems.

2. Assignment

Implement the filling up of the parking lot in Java or Python. The algorithm can be chosen freely, but you are not allowed to use any code that you yourself have not written.

2.1. Java

The code must contain a Main class, and within this, a main() function. It will receive all inputs on the standard input, and should output the solution to the standard output. Upload the zipped source code files of your application to the BME MIT HomeWork portal. (https://hf.mit.bme.hu).

2.2. Python

The code must be a single python file, that will be run and receives all inputs onto the standard input, and it should write the solution to the standard output. Upload the zipped single python file to the BME MIT Homework portal. Use Python3.x, and only standard libraries are available (e.g. no numpy!). (https://hf.mit.bme.hu).

2.3. Inputs

The input is multiple lines of text, with the individual elements being tab separated. The first line contains the length and width of the parking lot. The second line contains the number of vehicles. Each subsequent row contains the length and width of a single vehicle. So for the above example, the input would look like this:

```
5	7
7
4	2
3	2
1	2
2	5
2	2
2	1
3	1
```

2.4. Output

Output the entire P matrix to the standard output, with values on each line separated by tabs. (A common mistake is to have extra tabs at the end of the lines, and we will not accept solutions that make this error).

```
1	1	1	1	4	4	6
1	1	1	1	4	4	6
2	2	3	3	4	4	7
2	2	5	5	4	4	7
2	2	5	5	4	4	7
```

3. Evaluation

The evaluation is performed with problems of increasing difficulty, in which case the size of the parking lot is increased (from 3x3 parking lots up to 45x45 parking lots), as is the number of vehicles. Only solutions that place all vehicles correctly will be accepted. The final score will be the number of successful tests. If your solution places the vehicles incorrectly, you will be notified of any errors. You may then edit and resubmit your solution. You have about 5 seconds of CPU time to come up with the solution, and if your solution is incorrect or times out, then your solution will not be given the next parking lot.