Algorithm Design Laboratory with Applications

Prof. Stefano Leucci

Problem: *Computing Power.*

You often need to run some very computationally intensive experiments within a very tight deadline. There are k cloud computing companies, each of which rents n machines with different computing power and prices. Any of these machines can be used to run your experiments, but the time needed will depend on its computing power. More precisely, the j-th machine of the i-th type requires $d_{i,j}$ days to complete your experiments and costs $c_{i,j}$ dollars.

Design a data structure that can preprocess all values $d_{i,j}$ and $c_{i,j}$ to answer the following query: given a deadline D (in days) report, for each company, the cost of the cheapest machine that is able to complete the experiments in at most D days.

Input. The input consists of a set of instances, or *test-cases*, of the previous problem. The first line contains the number T of test-cases. The first line of each test-case contains three integers k, n, and q, where n and k are as described above and q is the number of queries. The i-th of the next k lines describes the machines offered by the i-th company a consists of the $2k$ integers $d_{i,1}, c_{i,1}, d_{i,2}, c_{i,2}, \ldots, d_{i,k}, c_{i,k}$. Finally, the next line contains q integers $D_1, \ldots, D_q$, where D_h represents the deadline D of the h-th query.

Output. The output consists of q lines for each test-case. The h-th of these q lines is the answer to the h-th query and contains k integers $r_1, \ldots, r_k$ separated by a space, where the generic integer r_i is the cost of the cheapest machine offered by the i-th company among those that require at most D_h days to complete the experiments. If all the machines of the i-th company require more than D_h days, then $r_i = -1$.

Assumptions. $1 \leq T \leq 10$; $1 \leq k \leq 2^{13}$; $1 \leq n \leq 2^{13}$; $1 \leq q \leq 2^{18}$;
$\forall i = 1, \ldots, k \; \forall j = 1, \ldots, n-1, \; d_{i,j} < d_{i,j+1}$ and $c_{i,j} > c_{i,j+1}$.

Example.

Input:

```
1
3 4 5
2 500 6 300 12 100 15 80
1 800 3 400 10 150 13 100
4 200 8 150 9 110 14 90
8 3 1 11 14
```

Output:

```
300 400 150
500 400 -1
-1 800 -1
300 150 110
100 100 90
```

Requirements. Your algorithm should have a preprocessing time of $O(nk)$, a query time of $O(k + \log n)$, and must be able to answer queries *online*, i.e., the answer to a query must be returned before the next query is read.

Notes. A reasonable implementation should not require more than 2 seconds for each input file.