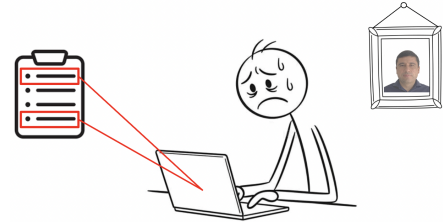


Procrastination Optimization

You want to write a program that will allow you to optimize the Kattis problems you select when completing your COMP 321 assignments. For each assignment, you only need to answer a subset of the listed questions to reach full marks. Each problem has a:



- Point value (some positive integer)
- Difficulty rating (an integer from 1.0 to 10.0)
- Topic (like trees, graphs,...)
- Text length (number of words)

Although you're a lazy student, you still care about your grades, and so you want to achieve full marks on your assignment while putting in the least amount of effort possible. On a given assignment, you need to attain at least M points, and you want to do so while trying to:

1. Minimize the total difficulty of the chosen problems first, then
2. Minimize the number of problems solved. Then,
3. Align the topics of the problems with concepts you have the best understanding of. Then finally,
4. Minimize the total length of text read

These requirements are listed in preferential order. That is, you first and foremost want to minimize the difficulty of questions you select. If there are still some choices for question subsets, then you can narrow it down further by minimizing the number of points solved, and work down to the least pressing requirement, which is minimizing the total length of text you read.

Input

The first line contains two integers M and N , which are the minimum total points needed for full score and the number of problems on an assignment, respectively. Note that M can, and will be quite large, prepare for inputs where $M \geq 10^{20}$.

The next line contains a list of distinct strings: a variety of problem topics ranked from your best understanding to your worst understanding.

The following N lines describes the problems on the assignment in the following format:

Problem_No.	Points	Difficulty	Topic	Text_Length
-------------	--------	------------	-------	-------------

Where "Problem_No" $\in [1, N]$, is the problem number on the list of questions, "Points" $\in [0, M]$ is the number of points that problem is worth, "Difficulty" $\in [1.0, 10.0]$ is the difficulty rating of the problem (larger rating means more difficult), "topic" is a non-empty string without spaces (assume exactly one topic is assigned to each problem) which is the topic the problem is assigned to, and finally, $1 \leq \text{Text_Length} \leq 10^4$ is the number of words in the problem statement.

Output

One line, containing the problem numbers, separated by spaces, that you choose in order to complete your assignment under the above conditions. If no subset of problems can achieve at least M points, output -1 .

Sample Input 1

```
10 4
dp graphs arrays
1 5 3 dp 120
2 6 5 graphs 200
3 4 1 arrays 50
4 8 4 dp 300
```

Sample Output 1

```
3 4
```

Sample Input 2

```
10 3
stacks queues trees
1 2 1 stacks 100
2 2 1 stacks 100
3 10 9 trees 100
```

Sample Output 2

```
3
```

Sample Input 3

```
1000000000000000 4
greedy dijkstra strings
1 600000000000000 3 greedy 5000
2 500000000000000 2 dijkstra 8000
3 400000000000000 1 strings 2000
4 700000000000000 5 greedy 10000
```

Sample Output 3

```
1 3
```