8-3 Ambulance Dispatch (100 分)

Given the map of a city, with all the ambulance dispatch centers (救护车派遣中心) and all the pick-up spots marked. You are supposed to write a program to process the emergency calls. It is assumed that the callers are waiting at some pick-up spot. You must inform the nearest (that is, to take the minimum time to reach the spot) dispatch center if that center has at least one ambulance available. Note: a center without any ambulance must not be considered.

In case your options are not unique, inform the one with the largest number of ambulances available. If there is still a tie, choose the one that can pass the least number of streets to reach the spot, which is guaranteed to be unique.

Input Specification:

Each input file contains one test case. For each case, the first line contains two positive integers *N****s* (≤103) and *N****a* (≤10), which are the total number of pick-up spots and the number of ambulance dispatch centers, respectively. Hence the pick-up spots are numbered from 1 to *N****s*, and the ambulance dispatch centers are numbered from *A*−1 to *A*−*N****a*.

The next line gives *N****a* non-negative integers, where the *i*-th integer is the number of available ambulances at the *i*-th center. All the integers are no larger than 100.

In the next line a positive number *M* (≤104) is given as the number of streets connecting the spots and the centers. Then *M* lines follow, each describes a street by giving the indices of the spots or centers at the two ends, followed by the time taken to pass this street, which is a positive integer no larger than 100.

Finally the number of emergency calls, *K*, is given as a positive integer no larger than 103, followed by *K* indices of pick-up spots.

All the inputs in a line are separated by a space.

Output Specification:

For each of the *K* calls, first print in a line the path from the center that must send an ambulance to the calling spot. All the nodes must be separated by exactly one space and there must be no extra space at the beginning or the end of the line. Then print the minimum time taken to reach the spot in the next line. It is assumed that the center will send an ambulance after each call. If no ambulance is available, just print `All Busy` in a line. It is guaranteed that all the spots are connected to all the centers.

Sample Input:

```
7 3
3 2 2
16
A-1 2 4
A-1 3 2
3 A-2 1
4 A-3 1
A-1 4 3
```

```
6 7 1
1 7 3
1 3 3
3 4 1
6 A-3 5
6 5 2
5 7 1
A-2 7 5
A-2 1 1
3 5 1
5 A-3 2
8
6 7 5 4 6 4 3 2
```

Sample Output:

```
A-3 5 6
4
A-2 3 5 7
3
A-3 5
2
A-2 3 4
2
A-1 3 5 6
5
A-1 4
3
A-1 3
2
All Busy
```

Grading Policy:

```
Chapter 1: Introduction (6 pts.)
Chapter 2: Algorithm Specification (12 pts.)
Chapter 3: Testing Results (20 pts.)
Chapter 4: Analysis and Comments (10 pts.)
Write the program (50 pts.) with sufficient comments.
Overall style of documentation (2 pts.)
```