Bloomberg CODECON

Memory Limit: 1350 MB
Time Limit: 5 s

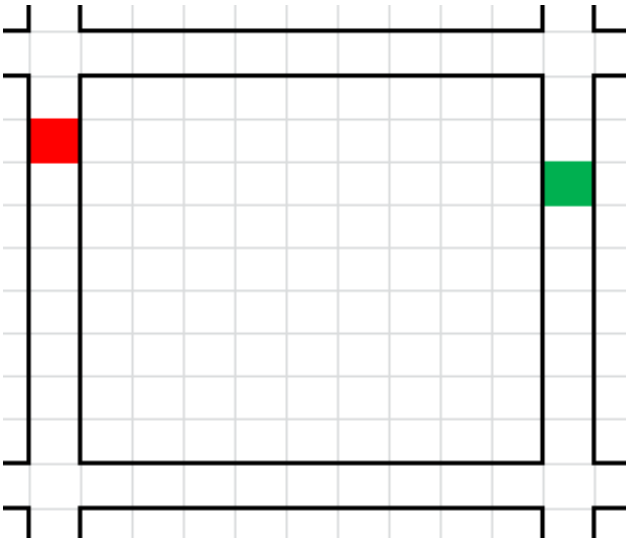
Zombie Ambulance (600 points)

Introduction

You're an ambulance driver in a city hit with a zombie plague. Patients are scattered across the city, and you're tasked with bringing back as many as you can before time runs out and they join the living dead!

Your ambulance starts at a specified **hospital location** on a coordinate grid. To save a particular patient, you have to drive to them and then return to the hospital. The ambulance travels at a constant 1 unit/sec, so **the time to rescue a particular patient is twice the distance of the optimal path from the hospital to that patient.**

However, there's a constraint: **your ambulance must drive around city blocks, not through them.** The city is made up of square 10x10 blocks on a coordinate grid, with streets along `x = 0, y = 0, x = 10, y = 10,` etc. This means the travel distance between two points is sometimes not just the Manhattan distance. In this example, the distance between the red and green tiles isn't the Manhattan distance of 11, but instead 15, to drive around the block:



Your task is to determine the maximum number of patients it's possible to save in the allotted time, and identify which patients are saved.

- Both the hospital and all the patients are guaranteed to be on a street (at least one of x and y is 0 mod 10).
- Loading, unloading, and curing patients all happen instantly.
- Driving through the hospital is fine.
- You can only carry one patient at a time - no carpools!
- For certain inputs, multiple solutions are possible, but **you will only be given test cases with a unique solution**.

Input Specifications

Your program will take one space-separated line with problem parameters, followed by one line for each patient.

The first line will contain, separated by spaces:

- **X**, the integer x-coordinate of the hospital **(0 ≤ X ≤ 100)**

- **Y**, the integer y-coordinate of the hospital **(0 ≤ Y ≤ 100)**
- **T**, the amount of time you have to rescue patients, in seconds **(0 ≤ T ≤ 2000)**
- **N**, the number of patients present **(0 ≤ N ≤ 20)**

The first line will be followed by **N** additional space-separated lines, containing:

- a single unique capital letter labeling the patient
- **X**, the integer x-coordinate of the patient **(0 ≤ X ≤ 100)**
- **Y**, the integer y-coordinate of the patient **(0 ≤ Y ≤ 100)**

Output Specifications

Based on the input, print to stdout a single line containing:

- the maximum number of patients it's possible to rescue in the **T** seconds provided
- if at least one patient can be saved, the labels of the patients that will be saved, sorted in ascending alphabetical order, separated by spaces

Sample Input/Output

Input

```
10 10 40 1
A 20 20
```

Output

```
1 A
```

Explanation

Patient A, the only patient, can be rescued in exactly 40 seconds:

1. (10,10) to (20,10) (ten seconds)
2. (20,10) to (20,20), pick up patient (ten seconds)
3. (20,20) to (20,10) (ten seconds)
4. (20,10) to (10,10), drop off patient (ten seconds)

Input

```
17 10 60 4
A 0 1
B 12 10
C 30 6
D 38 20
```

Output

```
2 B C
```

Explanation

Patient B can be rescued in 10 seconds, and Patient C in 34 seconds. Patient A could be rescued in 54 seconds, but it would then be impossible to rescue a second patient. There is no other pair of patients that can be rescued with the 60 seconds provided.