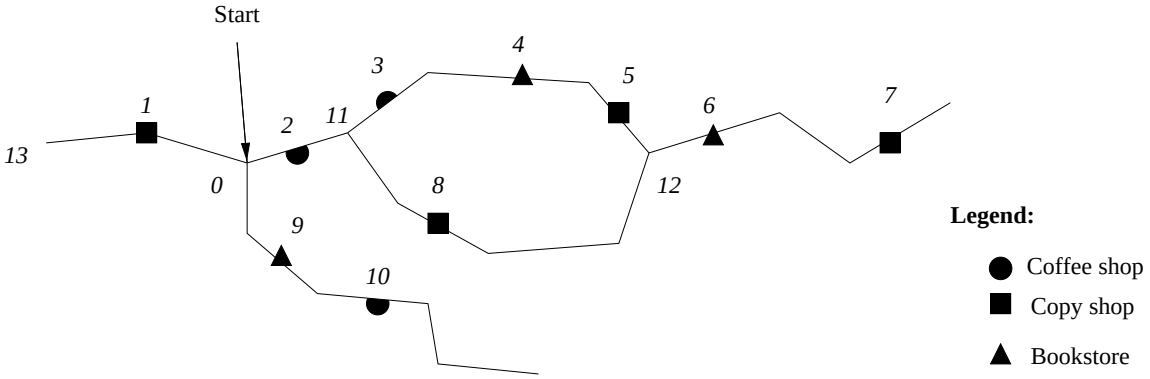
6. Consider a branching network of roads, sometimes doubling back on itself, along which one finds a variety of shops. The network is connected; if one starts walking from any fixed point in this network, one will encounter some sequence of shops, and one can reach any shop in the network. I'd like to know, however, what is the *first* shop of some particular kind (produce, meat, stereo equipment, etc.) that I might encounter on any particular walk. For example, on the following map, the first coffee shop I encounter, depending on the route I take, will be #2 or #10. The first bookstore will be #4, #6, or #9. The first copy shop will be #1, #8, or #5. You are to write a program to determine the general answer to the question, "Which shop of type X might I first encounter on a walk?"



The input (in free format) will consist of a non-negative integer called the *starting address*, another non-negative integer called the *target type*, a sequence of numbered *places* terminated by a pair of negative numbers, finally followed by a sequence of *paths* joining pairs of places. Each place takes the form of a pair of non-negative integers, the first of which (the *address*) is unique to each place, and the second of which (the *type*) corresponds to a type of shop; any number of addresses may have the same type. Each path consists of a pair of previously defined addresses, indicating that a direct path exists between the places with those addresses. Assume that all addresses are less than 1000.

Having read this input, your program must print out the list of places of the specified target type that one first encounters when traversing paths starting at the starting address (the shop at the starting address counts as an encounter). List the addresses of the places in increasing numerical order, without repetitions, using the format illustrated in the example.

Example:

Input:

```
0 2
0 0
1 2 8 2 5 2 7 2
2 1 3 1 10 1
9 3 4 3 6 3
11 0 12 0 13 0
-1 -1
13 1 1 0 0 9 9 10
0 2 2 11
11 3 11 8 3 4 4 5
5 12 8 12
12 6 6 7
```

Output:

```
Starting at 0, you might first encounter type 2 shop #1, 5, 8
```