2000 Canadian Computing Competition, Stage 1

Problem J5/S3: Surfin'

For this problem, we are concerned only with the markup that identifies links to other pages within a given page. A link within the page is denoted <A HREF="URL"> where URL is the URL of some other page. A user viewing a page containing a link may click on the link to view the other page.

You are to write a program that reads a number of web pages and their associated URLs. For each link in each page, you must print the URL of the page containing the link, and the URL of the page referred to by the link. Following the last page, you are then given several pairs of URLs. For each pair you are to assume that you are viewing the page identified by the first URL, and determine whether it is possible to click a sequence of links so as to view the page identified by the second URL. If so, you should print "Can surf from *here* to *there*." where *here* and *there* are the two URLs. If not you should print "Can't surf from *here* to *there*.

The first line of input contains an integer n ≤ 100, the number of Web Pages. For each Web Page, there will be a line containing its URL, followed by several lines containing the page. The URL will consist of up to 80 non-blank printable characters and will not contain any quotation marks. The first line of the page will be <HTML> and the last line will be </HTML>. Each page will contain up to 100 links in the format described above. Each link will be contained within a single line of input. URLs in the link will be those of pages given in the input. The markup keywords A, HREF, and HTML will appear only in upper case.

Following the n pages will be several pairs of lines giving URLs required by the problem as specified above. The last line of input will be "The End". For each pair, print the appropriate message given above.

Sample Input

```
3
http://ccc.uwaterloo.ca
<HTML> <TITLE>This is the CCC page</TITLE>
Hello there boys
and girls. <B>Let's</B> try <A HREF="http://abc.def/ghi"> a little problem </A>
</HTML>
http://abc.def/ghi
<HTML> Now is the <TITLE>time</TITLE> for all good people to program.
<A HREF="http://www.www.www.com">hello</A><A HREF="http://xxx">bye</A>
</HTML>
http://www.www.www.com
<HTML>
<TITLE>Weird and Wonderful World</TITLE>
</HTML>
http://ccc.uwaterloo.ca
http://www.www.www.com
http://www.www.www.com
http://ccc.uwaterloo.ca
The End
```

Sample Output

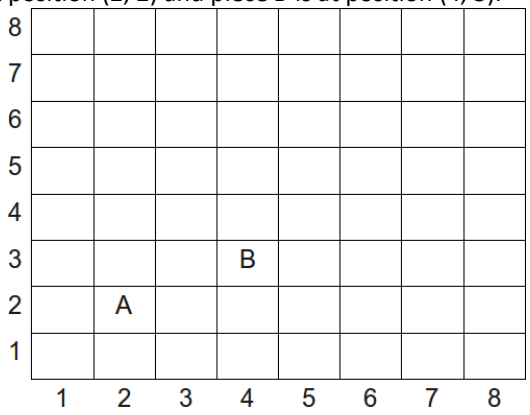
```
Link from http://ccc.uwaterloo.ca to http://abc.def/ghi
Link from http://abc.def/ghi to http://www.www.www.com
Link from http://abc.def/ghi to http://xxx

Can surf from http://ccc.uwaterloo.ca to http://www.www.www.com
Can't surf from http://www.www.www.com to http://ccc.uwaterloo.ca
```

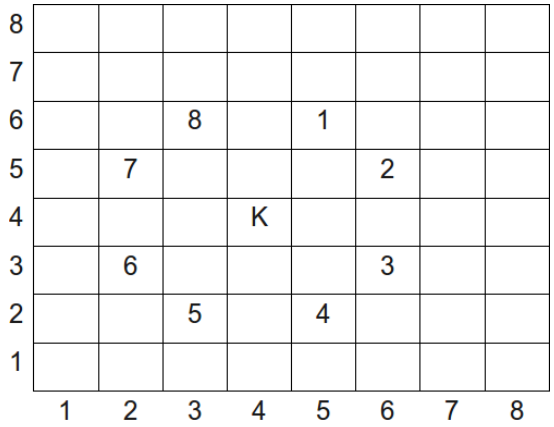
2010 Canadian Computing Competition, Stage 1

Problem J5: Knight Hop

Below is an 8 × 8 chessboard on which we will designate square locations using the ordered pairs as indicated. For example, notice that piece *A* is at position (2, 2) and piece *B* is at position (4, 3).



A knight is a special game piece that can leap over other pieces, moving in the "L" pattern. Specifically, in the diagram below, *K* represents the knight's starting position and the numbers 1 through 8 represent possible places the knight may move to.



Your program will read the starting location of the knight and output the smallest number of jumps or moves needed to arrive at a location specified in the second input.

Input Format

Your program will read four integers, where each integer is in the range 1…8. The first two integers represent the starting position of the knight. The second two integers represent the final position of the knight.

Output Format

Your program should output the minimum (non-negative integer) number of moves required to move the knight from the starting position to the final position. Note that the knight is not allowed to move off the board during the sequence of moves.

Sample Cases

Input	**Input**
2 1	4 2
3 3	7 5

Output	**Output**
1	2

2007 Canadian Computing Competition, Stage 1

Problem J5: Keep on Truckin'

A truck driver is planning to drive along the Trans-Canada highway from Vancouver to St. John's, a distance of 7000 km, stopping each night at a motel. The driver has been provided with a list of locations of eligible motels, with the respective distance of each motel, measured in km, from the starting point in Vancouver. Some of the motel locations are:

0, 990, 1010, 1970, 2030, 2940, 3060, 3930, 4060, 4970, 5030, 5990, 6010, 7000

but more motel locations may be added just before the trip begins.

Determine if it is possible to complete the journey if:

1. the trucking company insists that the driver travels a minimum distance of *A* km per day,
2. the law sets a maximum distance of *B* km per day, and
3. each night, the driver must stay at an eligible motel (from the above list or the additional locations described below).

The driver is interested in different options when making the trip, and you are to write the program to calculate how many different options there are.

For example, if no new motel locations are added, $A = 1$ and $B = 500$, then it is impossible to make the trip, i.e., the number of options is 0. If $A = 970$ and $B = 1030$ then there is one way to make the trip, but if $A = 970$ and $B = 1040$ then there are four ways to make the trip. There are two ways to make the trip if $A = 970$, $B = 1030$, and we add one stop at 4960.

Input

The first two lines of the input are the minimum distance *A* and the maximum distance *B* ($1 \le A \le B \le 7000$), both of which are integers. The third line of the input is an integer *N* ($0 \le N \le 20$), followed by *N* lines, each giving the location *m* of an additional eligible motel ($0 < m < 7000$).

You should note that no two motels are located at the same distance from Vancouver.

Output

Output the number of different ways the driver can choose the motels to make the trip, under the given constraints.

Examples

Sample Input	**Sample Input**	**Sample Input**	**Sample Input**
1	970	970	970
500	1030	1040	1030
0	0	0	1
			4960
Sample Output	**Sample Output**	**Sample Output**	**Sample Output**
0	1	4	2