2004/2005 SOUTHERN CALIFORNIA REGIONAL
ACM INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 7
Escape from Swamp County Park

(requires an interactive server)

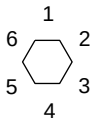
A chill wind descended upon Swamp County. It swept out of the north hills like a flood, then settled in the heart of the swamp. There it festered among the warm moist air of the cursed bog, and it stewed a thick, toxic fog. Your team wandered into the park (before the wind and fog) for a relaxing picnic. Now, however, you are lost, choking for breath, and must escape before you suffocate.

Swamp County Park is carved out of the swamp according to a hexagonal grid, with pathways cut through barbed vines. You must grope through the blinding fog to find the exit. You have a limited number of movement attemps before the toxic fumes overtake you; you must move with efficiency to escape the park. Fortunately, the sheer size of the park dilutes the toxicity of the fog, so the bigger the park, the more moves you have to escape. Sadly, you don't know the exact size of the park, other than it cannot exceed the size of the swamp, which is 20-by-20 hexagonal grid locations.

Input and Output

Your program must converse with a server, issuing movement attempts to standard output and receiving responses through standard input. Your program starts the conversation. You will never start on the exit location. You may attempt to move in any of six directions:

1 attempt to move one hex location up
2 attempt to move one hex location up and to the right
3 attempt to move one hex location down and to the right
4 attempt to move one hex location down
5 attempt to move one hex location down and to the left
6 attempt to move one hex location up and to the left



Each movement attempt sent to standard output must be a single decimal digit '1'..'6' followed by end-of-line. *It is very important that your program flush standard output after issuing a movement attempt.* e.g.

C	`fputs("1\n",stdout); fflush(stdout);`
C++	`cout << "1\n" << flush;`
Java	`System.out.println("1"); System.out.flush();`

If your program encounters the exit, it should terminate normally. Successful escape from the park generates a Correct condition. Any characters sent to the server that do not follow the protocol are deemed a Runtime Error. If the server has to wait more than one second (real time) between your movement requests, your program will trigger a Time Limit Exceeded condition. If your program continues to move around after reaching the exit, it will receive a Presentation Error. If the number of move attempts exceeds the limit ($2 \times rows \times columns$), your program will elicit a Wrong Answer condition (you suffocated).

In response to your attempted moves, the server answers with either

- **`n`** (you ran into a hex location with barbed vines, and did not move)
- **`y`** (you successfully moved in your intended direction)
- **`E`** (you successfully moved in your intended direction and found the exit)

Each response from the server consists of one of the characters shown above, followed by end-of-line. You must read these responses from standard input.

Problem 7
Escape from Swamp County Park (continued)

Hints for Testing

You may construct a park layout by hand on the supplied hexagonal graph paper (the judges recommend using pencil rather than pen). Choose a starting point and an exit location, then simply execute your program and let standard input come from your command line environment. You, the programmer, must act as the server. Keep track of your program's location within the park and count the moves. Type the appropriate response to each movement request.

Additionally, you have access to the server program that the judges use to assess your solution. For this, you must type in a representation of the park and feed this to the server. The sample representation below corresponds to the park layout of Figure 7.1. The first line is two space-separated integers specifying the number of rows and columns, respectively, in the park. The remaining lines describe "rows" of the park: An asterisk '*' represents a vine-filled location, a space ' ' indicates a pathway, upper case 'E' designates the exit, and a hyphen '-' represents the starting location in the park. The starting location is a pathway. Except for the exit location, the outer edge of the park must be marked by asterisks.

Execute the server using the test7 command:

`test7` *sourceFile mazeFile traceFile*

where *sourceFile* is the name of your source code file, *mazeFile* is the name of your maze representation, and *traceFile* is the name of a file to receive the transcript of your conversation with the server.

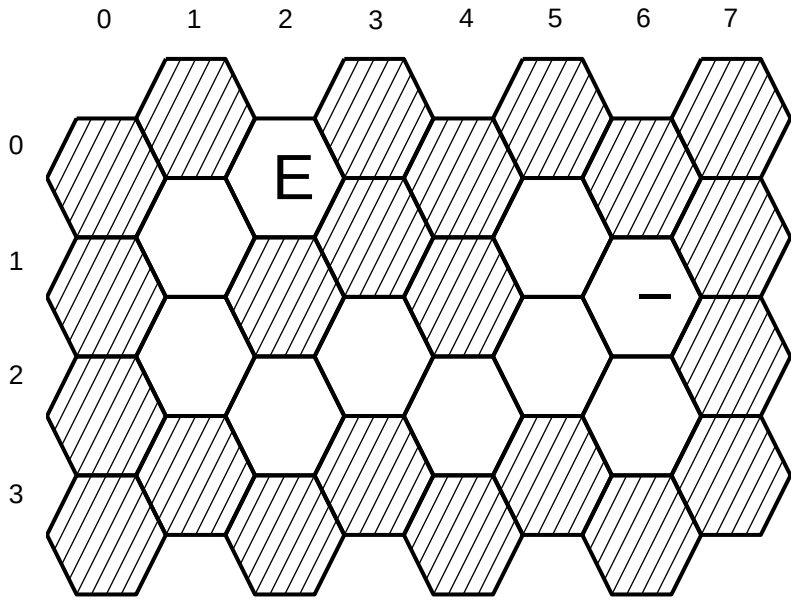


Figure 7.1. A 4-row by 6-column park corresponding to maze.txt.
The starting point is at (row,column) location (1,6).
The exit is at (row,column) location (0,2).

Sample Maze Representation, **`maze.txt`**

```
4 8
**E*****
* *** -*
*      *
********
```