PROBLEM:

The famous treasure hunter Bob, has to hunt treasures in the old ruins of a long lost city. Apart from facing obstacles, he has to avoid deadly pits along the way. To do so Bob has a map of the area. In this map, the treasures are marked by ‘*’ and the pits are marked by ‘#’. Bob’s job is to find the shortest route, through which he would not visit any of the pits, and he could find the maximum number of the treasures (if not all). Write a program for his assistance.

INPUT:

Each input consists of a two- dimensional matrix of size n=5. For each character, a space ‘ ‘ stands for an open space; a hash mark ‘#’ stands for an obstructing wall; the capital letter ‘**S**’ stands for the position of Bob’s tent, which is where his journey is to start and end; the capital letter ‘**X**’ stands for a dangerous place(the pits); and an asterisk ‘*****’ stands for a place he has to visit to find treasure. The perimeter of the map is always closed, i.e., there is no way to get out from the coordinate of the ‘**S**’. The number of places that Bob has to visit is at most **10**.

OUTPUT:

For each input, if Bob is unable to visit any treasure place at all, just print “DEAD END”. Otherwise, your program should output the lexicographically smallest shortest path so that the number of target places that Bob visits is maximized. Use the letters '**N**', '**S**', '**E**' and '**W**' to denote north, south, east and west respectively. Note that by 'north' we mean facing upwards. You can be sure that the length of a correct output path will never exceed 200.

SAMPLE:

- Input:
  ```
  #####
  #  S#
  # XX#
  #  *#
  #####
  ```
  Output:
  WWSSEEWWNNEE

- Input:
  ```
  #####
  #* X#
  ###X#
  #S *#
  #####
  ```
  Output:
  EEWW

- Input:

```
#####
#S X#
#  X#
# #*#
#####
```

Output:
DEAD END