Longwood University (Fall 2011)

Problem 1 : Flooded Maze (By Robert Marmorstein, Longwood University)

Introduction

While mining for precious gems, the dwarves accidentally tapped into an underground spring. It is flooding their mine slowly, but will soon fill the entire cavern. In order to make an evacuation plan, the dwarves need to know when the flood will reach each of their lookout stations. The flood spreads one cave at a time: every hour any unflooded cave next to a flooded cave or the spring becomes flooded. The flood only spreads due east, due west, due north, or due south.

Input

The input consists of a rectangular ASCII maze representing a two-dimensional projection of the cavern. The walls of the maze are depicted by a '#' character while the caves are represented by spaces. The initial source of the spring is represented by a capital 'S' character. Each lookout station is depicted by a single decimal digit (i.e. a character in the range '0' to '9'). Like empty caves, the lookout stations can flood and subsequently cause neighboring caves to flood.

The height and width of the maze will both be at most 100 characters in size. The boundary of the maze will always be intact (that is, the entire boundary will be composed entirely of '#' characters). The map is oriented with north at the top and west on the left.

Output

Print a list of lookout stations in the order the flood will reach them. The list should be space separated with an end-of-line character terminating the list. There should be no white space between the last station number and the end-of-line character.

If the flood reaches two stations in the same hour, the lowest numbered station should be printed first. If a station cannot be flooded, it should be omitted from the list. It is possible for none of the lookout stations to be flooded, in which case the program should simply print a blank line consisting of a single end-of-line character.

Examples

Input:

```
#####
#S#1#
# # #
#  2#
#####
```

Output:

```
2 1
```

(More examples on the back)

Input:

```
########
#S   1 #
# #  # #
#3 4  2#
########
```

Output:

```
3 1 4 2
```

Input:

```
######
#S1#2#
######
```

Output:

```
1
```