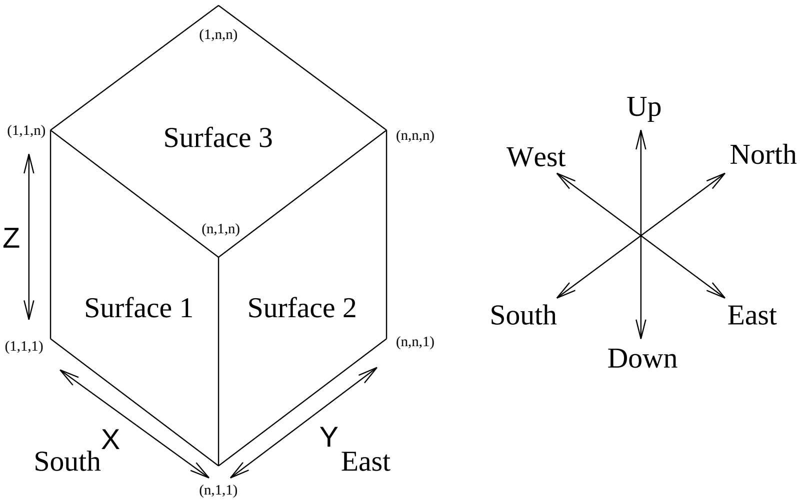
British Informatics Olympiad Final

11–13 April 1997
Sponsored by Data Connection

3-Dimensional Maze

Given a 3-dimensional maze, a start point and an end point, we are trying to find the shortest path between the two points. The mazes are constructed within cubes, and a cube of size $n \times n \times n$ can be thought of as containing n^3 locations. These locations are either 'good' or 'bad', and a valid move is between two adjacent (but not diagonally adjacent) good locations. Moves can never go outside the boundary of the cube.

Rather than describe the maze by indicating which of the n^3 locations are good and bad, three surfaces of the cube will be given instead. As shown in the diagram, surface 1 is the southern (x-z) face of the cube, surface 2 the eastern (y-z) face, and surface 3 the top (x-y) face. The corner at the bottom-west of surface 1 is the point $(1, 1, 1)$.



A location in the maze is 'good' if and only if :

- The point on surface 1, directly to the south of the location, is 'good'.
- The point on surface 2, directly to the east of the location is 'good'.
- The point on surface 3, directly above the location is 'good'.

Note that locations which touch the given surfaces, eg. $(2, 1, 2)$ or $(n, 1, n)$, are still subject to these three requirements. Ie. Just because a location touches part of surface that is 'good' does not automatically imply the location itself is 'good'.

Your program should first read in a number indicating the size of the cube, followed by the data for the three surfaces (which will be given in order). For surface 1 the corner (1,1,n) will be given first, for surface 2 the corner (n,1,n), and for surface 3 the corner (1,n,n). For each surface, treating the given corner as 'top-left', the data will go from left to right then top to bottom. A '`*`' will indicate a 'bad' square, and a '`.`' will indicate a 'good' square.

The next input will be three numbers representing the co-ordinates of the start point. This will be followed on the next line by the co-ordinates of the end point. You should calculate the shortest path between these two points, then output it along with the number of moves involved. The directions up, down, north, south, east and west should be abbreviated to `U`, `D`, `N`, `S`, `E`, `W`. [For clarity, please output your directions in groups of 5.]

If you do not believe there is a solution you should print out '`Impossible`'. The given start and end points will always be 'good'.

Sample Input

```
4
..*.
*...
..*.
..*.
*...
....
..*.
...*
....
.*..
.*..
....
1 1 1
4 4 4
```

Sample Output

```
9
UEUEN EUNN
```