Marathon Match

Problem Statement

Contest: Marathon Match 68

Normal view

Problem: BeautifulCrossword

Problem Statement

You are given a square board of **N**x**N** cells and a list of words. Your task is to create a crossword on this board using only these words so that it scores as high as possible on the given criteria of "beauty".

You are given the size of the board **N**, the list of allowed **words** and the list of **weights** used to balance the scores on individual criteria. You have to return a vector <string> which contains your crossword. Each element corresponds to one row of the board. '.' mark empty cells, and uppercase letters mark the cells in which letters are written (letter cells). Rows must be listed from top to bottom, and cells within each row must be listed from left to right. Each maximal vertical sequence (when read from top to bottom) and horizontal sequence (when read from left to right) of two and more letters must be a valid word (i.e., it must be present in **words**). Each letter must be a part of at least one word (vertical or horizontal). Each word can be used at most once.

A valid crossword will be scored by calculating the scores on 4 criteria:

- *Board filling* score is the total number of letter cells, divided by the total number of cells on the board (**N*****N**).
- *Rows/columns filling* score is the number of columns which contain at least one letter cell, multiplied by the number of rows which contain at least one letter cell, divided by **N*****N**.
- *Symmetry score* is calculated as follows: for each cell of 1/8 of the grid the number of empty cells among itself and 7 cells symmetrical to it (horizontally, vertically, diagonally and combinations) is calculated. More exactly, let (i, j) be the cell in row i, column j of the board (both rows and columns are numbered from 0 to **N**-1). Then for each cell (i, j), such that 0 <= i <= N-i-1, 0 <= j <= i, the number of empty cells is calculated among the following eight cells: (i, j), (j, i), (N-i-1, j), (j, N-i-1), (i, N-j-1), (N-j-1, i), (N-i-1, N-j-1) and (N-j-1, N-i-1). If some cell occures several times among these 8 cell and it is empty, then each occurrence contributes independently towards the number of empty cells. Next, values 0 or 8 add 1.0 to the score, values 1 or 7 add 0.5 to the score, values 2 or 6 add 0.1 to the score. Finally, the score is divided by the number of cells in 1/8 of the grid, i.e., the number of cells (i, j), such that 0 <= i <= N-i-1, 0 <= j <= i.
- *Crossings score* is the number of crossing cells, divided by the total number of letter cells. A letter cell is a crossing cell if it is a part of both vertical and horizontal word. If there are no letter cells in a crossword, then its crossings score is 0.

Your score for the test case will be the weighted average of these values with weights given in **weights**. In other words, if your board filling, rows/columns filling, symmetry and crossings scores are A, B, C and D, then your score for this test case is (A***weights**[0] + B***weights**[1] + C***weights**[2] + D***weights**[3]) / (**weights**[0] + **weights**[1] + **weights**[2] + **weights**[3]).

Your overall score will be the sum of individual scores over all test cases. Invalid returns are given zero score.

A tester is provided for offline testing. See its source code for details of test case generation and scoring.

Definition

Class:	BeautifulCrossword
Method:	generateCrossword
Parameters:	int, vector <string>, vector <int>
Returns:	vector <string>

Method signature:vector <string> generateCrossword(int N, vector <string> words, vector <int> weights)
(be sure your method is public)

Notes

- **words** is generated from this list (which is effectively TWL06 list with words of length 2 removed) using the following algorithm. Value **prob** is chosen between 0.001 and 0.01, and each word from the list is included in **words** with this probability. **words** is always sorted in alphabetical order. All elements of **words** will be uppercase.
- **N** and elements of **weights** will be chosen uniformly at random.
- The memory limit is 1024 MB and the time limit is 10 seconds per test case (which includes only time spent in your code). There is no explicit code size limit. The implicit source code size limit is around 1 MB (it is not advisable to submit codes of size close to that or larger).
- There are 10 example test cases and 100 full submission test cases.

Constraints

- **N** will be between 20 and 100, inclusive (except example 0).
- **weights** will contain exactly 4 elements, each element being between 1 and 10, inclusive.
- Your return must contain **N** elements, each element must contain exactly **N** characters.
- Each character of your return must be 'A'..'Z' or '.'.

Examples

0)

```
N = 11
Number of words = 179
Weights are  6  8  7  10
```

One of the possible crosswords for this test case is

```
BORER.BOTAS
R......V..I
A....C.E..N
S.F..O.R..E
H.U..C..T.W
..STACKER..
..E..O..A..
S.L..I..Y.U
I..INDEX..M
M.........P
AHED..SINUS
```

1)

```
N = 74
Number of words = 1258
Weights are  9  7  7  3
```

2)

```
N = 78
Number of words = 645
Weights are  5  10  1  8
```

3)

```
N = 41
Number of words = 903
Weights are  7  4  2  10
```

4)

```
N = 43
 Number of words = 198
 Weights are  2  1  10  2
```

5)

```
N = 71
 Number of words = 489
 Weights are  7  9  4  9
```

6)

```
N = 80
 Number of words = 1578
 Weights are  2  2  9  1
```

7)

```
N = 24
 Number of words = 1120
 Weights are  9  4  10  9
```

8)

```
N = 84
 Number of words = 398
 Weights are  7  8  4  7
```

9)

```
N = 87
 Number of words = 289
 Weights are  4  10  2  2
```
