

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 // O(nm*8^s + ws) time | O(nm + ws) space
4 function boggleBoard(board, words) {
5   const trie = new Trie();
6   for (const word of words) {
7     trie.add(word);
8   }
9   const finalWords = {};
10  const visited = board.map(row => row.map(letter => false));
11  for (let i = 0; i < board.length; i++) {
12    for (let j = 0; j < board[i].length; j++) {
13      explore(i, j, board, trie.root, visited, finalWords);
14    }
15  }
16  return Object.keys(finalWords);
17 }
18
19 function explore(i, j, board, trieNode, visited, finalWords) {
20   if (visited[i][j]) return;
21   const letter = board[i][j];
22   if (!(letter in trieNode)) return;
23   visited[i][j] = true;
24   trieNode = trieNode[letter];
25   if ('*' in trieNode) finalWords[trieNode['*']] = true;
26   const neighbors = getNeighbors(i, j, board);
27   for (const neighbor of neighbors) {
28     explore(neighbor[0], neighbor[1], board, trieNode, visited, finalWords);
29   }
30   visited[i][j] = false;
31 }
32
33 function getNeighbors(i, j, board) {
```

Solution 1 Solution 2 Solution 3

```
1 function boggleBoard(board, words) {
2   // Write your code here.
3 }
4
5 // Do not edit the line below.
6 exports.boggleBoard = boggleBoard;
7
```

Our Tests

Custom Output

Submit Code

1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.

2

3 // O(nm*8^s + ws) time | O(nm + ws) space

4 function boggleBoard(board, words) {

5 const trie = new Trie();

6 for (const word of words) {

7 trie.add(word);

8 }

9 const finalWords = {};

10 const visited = board.map(row => row.map(letter => false));

11 for (let i = 0; i < board.length; i++) {

12 for (let j = 0; j < board[i].length; j++) {

13 explore(i, j, board, trie.root, visited, finalWords);

14 }

15 }

16 return Object.keys(finalWords);

17 }

18

19 function explore(i, j, board, trieNode, visited, finalWords) {

20 if (visited[i][j]) return;

21 const letter = board[i][j];

22 if (!(letter in trieNode)) return;

23 visited[i][j] = true;

24 trieNode = trieNode[letter];

25 if ('*' in trieNode) finalWords[trieNode['*']] = true;

26 const neighbors = getNeighbors(i, j, board);

27 for (const neighbor of neighbors) {

28 explore(neighbor[0], neighbor[1], board, trieNode, visited, finalWords);

29 }

30 visited[i][j] = false;

31 }

32

33 function getNeighbors(i, j, board) {

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

1	1	1	1	1	1	1
2	1	1	1	1	1	1
3	1	1	1	1	1	1
4	1	1	1	1	1	1
5	1	1	1	1	1	1
6	1	1	1	1	1	1
7	1	1	1	1	1	1
8	1	1	1	1	1	1
9	1	1	1	1	1	1
10	1	1	1	1	1	1
11	1	1	1	1	1	1
12	1	1	1	1	1	1
13	1	1	1	1	1	1
14	1	1	1	1	1	1
15	1	1	1	1	1	1
16	1	1	1	1	1	1
17	1	1	1	1	1	1
18	1	1	1	1	1	1
19	1	1	1	1	1	1
20	1	1	1	1	1	1
21	1	1	1	1	1	1
22	1	1	1	1	1	1
23	1	1	1	1	1	1
24	1	1	1	1	1	1
25	1	1	1	1	1	1
26	1	1	1	1	1	1
27	1	1	1	1	1	1
28	1	1	1	1	1	1
29	1	1	1	1	1	1
30	1	1	1	1	1	1
31	1	1	1	1	1	1
32	1	1	1	1	1	1
33	1	1	1	1	1	1
34	1	1	1	1	1	1
35	1	1	1	1	1	1
36	1	1	1	1	1	1
37	1	1	1	1	1	1
38	1	1	1	1	1	1
39	1	1	1	1	1	1
40	1	1	1	1	1	1
41	1	1	1	1	1	1
42	1	1	1	1	1	1
43	1	1	1	1	1	1
44	1	1	1	1	1	1
45	1	1	1	1	1	1
46	1	1	1	1	1	1
47	1	1	1	1	1	1
48	1	1	1	1	1	1
49	1	1	1	1	1	1
50	1	1	1	1	1	1
51	1	1	1	1	1	1
52	1	1	1	1	1	1
53	1	1	1	1	1	1
54	1	1	1	1	1	1
55	1	1	1	1	1	1
56	1	1	1	1	1	1
57	1	1	1	1	1	1
58	1	1	1	1	1	1
59	1	1	1	1	1	1
60	1	1	1	1	1	1
61	1	1	1	1	1	1
62	1	1	1	1	1	1
63	1	1	1	1	1	1
64	1	1	1	1	1	1
65	1	1	1	1	1	1
66	1	1	1	1	1	1
67	1	1	1	1	1	1
68	1	1	1	1	1	1
69	1	1	1	1	1	1
70	1	1	1	1	1	1
71	1	1	1	1	1	1
72	1	1	1	1	1	1
73	1	1	1	1	1	1
74	1	1	1	1	1	1
75	1	1	1	1	1	1
76	1	1	1	1	1	1
77	1	1	1	1	1	1
78	1	1	1	1	1	1
79	1	1	1	1	1	1
80	1	1	1	1	1	1
81	1	1	1	1	1	1
82	1	1	1	1	1	1
83	1	1	1	1	1	1
84	1	1	1	1	1	1
85	1	1	1	1	1	1
86	1	1	1	1	1	1
87	1	1	1	1	1	1
88	1	1	1	1	1	1
89	1	1	1	1	1	1
90	1	1	1	1	1	1
91	1	1	1	1	1	1
92	1	1	1	1	1	1
93	1	1	1	1	1	1
94	1	1	1	1	1	1
95	1	1	1	1	1	1
96	1	1	1	1	1	1
97	1	1	1	1	1	1
98	1	1	1	1	1	1
99	1	1	1	1	1	1
100	1	1	1	1	1	1

Run or submit code when you're ready.