

PromptScratchpadOur Solution(s)Video Explanation

Run Code

Solution 1

```
21     foreach (string key in finalWords) {
22         finalWordsArray.Add(key);
23     }
24     return finalWordsArray;
25 }
26
27 public static void explore(int i, int j, char[,] board, TrieNode trieNode, bool[,] visited,
28     HashSet<string> finalWords) {
29     if (visited[i,j]) {
30         return;
31     }
32     char letter = board[i,j];
33     if (!trieNode.children.ContainsKey(letter)) {
34         return;
35     }
36     visited[i,j] = true;
37     trieNode = trieNode.children[letter];
38     if (trieNode.children.ContainsKey('*')) {
39         finalWords.Add(trieNode.word);
40     }
41     List<int[]> neighbors = getNeighbors(i, j, board);
42     foreach (int[] neighbor in neighbors) {
43         explore(neighbor[0], neighbor[1], board, trieNode, visited, finalWords);
44     }
45     visited[i,j] = false;
46 }
47
48 public static List<int[]> getNeighbors(int i, int j, char[,] board) {
49     List<int[]> neighbors = new List<int[]>();
50     if (i > 0 && j > 0) {
51         neighbors.Add(new int[] {i - 1, j - 1});
52     }
53     if (i > 0 && j < board.GetLength(1) - 1) {
54         neighbors.Add(new int[] {i - 1, j + 1});
55     }
56     if (i < board.GetLength(0) - 1 && j < board.GetLength(1) - 1) {
57         neighbors.Add(new int[] {i + 1, j + 1});
58     }
59     if (i < board.GetLength(0) - 1 && j > 0) {
60         neighbors.Add(new int[] {i + 1, j - 1});
61     }
62     if (i > 0) {
63         neighbors.Add(new int[] {i - 1, j});
64     }
65     if (i < board.GetLength(0) - 1) {
66         neighbors.Add(new int[] {i + 1, j});
67     }
68     if (j > 0) {
69         neighbors.Add(new int[] {i, j - 1});
70     }
71     if (j < board.GetLength(1) - 1) {
72         neighbors.Add(new int[] {i, j + 1});
73     }
74     return neighbors;
75 }
76
77 public class TrieNode {
78     public Dictionary<char, TrieNode> children = new Dictionary<char, TrieNode>();
79     public string word = "";
80 }
81
82 public class Trie {
83     public TrieNode root;
84     public char endSymbol;
85
86     public Trie() {
87         this.root = new TrieNode();
88         this.endSymbol = '*';
89     }
90
91     public void Add(string str) {
92         TrieNode node = this.root;
93         for (int i = 0; i < str.Length; i++) {
94             char letter = str[i];
95             if (!node.children.ContainsKey(letter)) {
96                 TrieNode newNode = new TrieNode();
97                 node.children.Add(letter, newNode);
98             }
99             node = node.children[letter];
100         }
101         node.children[this.endSymbol] = null;
102         node.word = str;
103     }
104 }
105 }
106
```