

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 #include <unordered_map>
4 #include <vector>
5 using namespace std;
6
7 class TrieNode {
8 public:
9     unordered_map<char, TrieNode*> children;
10    string word = "";
11 };
12
13 class Trie {
14 public:
15     TrieNode* root;
16     char endSymbol;
17
18     Trie();
19     void add(string str);
20 };
21
22 void explore(int i, int j, vector<vector<char>>> board, TrieNode* trieNode,
23             vector<vector<bool>>> *visited,
24             unordered_map<string, bool> *finalWords);
25 vector<vector<int>>> getNeighbors(int i, int j, vector<vector<char>>> board);
26
27 // O(nm*8^s + ws) time | O(nm + ws) space
28 vector<string> boggleBoard(vector<vector<char>>> board, vector<string> words) {
29     Trie trie;
30     for (string word : words) {
31         trie.add(word);
32     }
33     unordered_map<string, bool> finalWords;
```

Our Tests

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 #include <vector>
4 #include <string>
5 #include <unordered_map>
6 #include <iostream>
7 using namespace std;
8
9 vector<string> boggleBoard(vector<vector<char>>> board, vector<string> words) {
10     Trie trie;
11     for (string word : words) {
12         trie.add(word);
13     }
14     unordered_map<string, bool> finalWords;
15
16     vector<vector<int>>> neighbors = getNeighbors(0, 0, board);
17
18     explore(0, 0, board, trie.root, &neighbors, &finalWords);
19
20     return finalWords;
21 }
```

Solution 1 Solution 2 Solution 3

```
1 #include <vector>
2 using namespace std;
3
4 vector<string> boggleBoard(vector<vector<char>>> board, vector<string> words) {
5     // Write your code here.
6     return {};
7 }
8
```

Custom Output

Submit Code

```

14
15
16 class Program {
17     static void Main() {
18         // Read input
19         int n = int.Parse(Console.ReadLine());
20         int k = int.Parse(Console.ReadLine());
21
22         // Read the array
23         int[] arr = new int[n];
24         for (int i = 0; i < n; i++) {
25             arr[i] = int.Parse(Console.ReadLine());
26         }
27
28         // Sort the array
29         Array.Sort(arr);
30
31         // Find the k-th smallest element
32         int result = arr[k - 1];
33
34         // Print the result
35         Console.WriteLine(result);
36     }
37 }

```

Run or submit code when you're ready.