

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1

Solution 2

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 #include <vector>
4 #include <unordered_map>
5 #include <climits>
6 #include <algorithm>
7 #include <cmath>
8
9 using namespace std;
10
11 int getIdxAtMinValue(vector<int> array);
12 int distanceBetween(int a, int b);
13
14 // O(b^2*r) time | O(b) space - where b is the number of blocks and r
15 // number of requirements
16 int apartmentHunting(vector<unordered_map<string, bool>> blocks,
17                      vector<string> reqs) {
18     vector<int> maxDistancesAtBlocks(blocks.size(), INT_MIN);
19     for (int i = 0; i < blocks.size(); i++) {
20         for (string req : reqs) {
21             int closestReqDistance = INT_MAX;
22             for (int j = 0; j < blocks.size(); j++) {
23                 if (blocks[j][req]) {
24                     closestReqDistance = min(closestReqDistance, distanceBetween
25                 )
26             }
27             maxDistancesAtBlocks[i] =
28                 max(maxDistancesAtBlocks[i], closestReqDistance);
29         }
30     }
31     return getIdxAtMinValue(maxDistancesAtBlocks);
32 }
33
```

Solution 1

Solution 2

Solution 3

```
1 #include <vector>
2 #include <unordered_map>
3
4 using namespace std;
5
6 int apartmentHunting(vector<unordered_map<string, bool>> blocks,
7                      vector<string> reqs) {
8     // Write your code here;
9     return -1;
10 }
11
```

Our Tests

Custom Output

Submit Code

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

2

3 #include <vector>

4 #include <unordered_map>

5 #include <climits>

6 #include <algorithm>

7 #include <cmath>

8

9 using namespace std;

10

11 int getIdxAtMinValue(vector<int> array);

12 int distanceBetween(int a, int b);

13

14 // O(b^2*r) time | O(b) space - where b is the number of blocks and r

15 // number of requirements

16 int apartmentHunting(vector<unordered_map<string, bool>> blocks,

17 vector<string> reqs) {

18 vector<int> maxDistancesAtBlocks(blocks.size(), INT_MIN);

19 for (int i = 0; i < blocks.size(); i++) {

20 for (string req : reqs) {

21 int closestReqDistance = INT_MAX;

22 for (int j = 0; j < blocks.size(); j++) {

23 if (blocks[j][req]) {

24 closestReqDistance = min(closestReqDistance, distanceBetween

25)

26 }

27 maxDistancesAtBlocks[i] =

28 max(maxDistancesAtBlocks[i], closestReqDistance);

29 }

30 }

31 return getIdxAtMinValue(maxDistancesAtBlocks);

32 }

33

1 #include <vector>

2 #include <unordered_map>

3

4 using namespace std;

5

6 int apartmentHunting(vector<unordered_map<string, bool>> blocks,

7 vector<string> reqs) {

8 // Write your code here;

9 return -1;

10 }

11

1	def main():	pass
2		
3	if __name__ == '__main__':	main()
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		
34		
35		
36		
37		
38		
39		
40		
41		
42		
43		
44		
45		
46		
47		
48		
49		
50		
51		
52		
53		
54		
55		
56		
57		
58		
59		
60		
61		
62		
63		
64		
65		
66		
67		
68		
69		
70		
71		
72		
73		
74		
75		
76		
77		
78		
79		
80		
81		
82		
83		
84		
85		
86		
87		
88		
89		
90		
91		
92		
93		
94		
95		
96		
97		
98		
99		
100		

Run or submit code when you're ready.