Solution 1 Solution 2

Our Solution(s)

Run Code

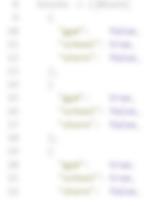
```
Your Solutions Run Code
```

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
   package main
   import "math"
   type Block map[string]bool
   // O(b^2*r) time | O(b) space - where b is the number of blocks
10 // and r is the number of requirements.
11 func ApartmentHunting(blocks []Block, reqs []string) int {
12
     maxDistancesAtBlocks := make([]int, len(blocks))
13
     for i := range blocks {
14
       maxDistancesAtBlocks[i] = -1
15
       for _, req := range reqs {
16
         closestReqDistance := math.MaxInt32
17
          for j := range blocks {
18
           if blocks[j][req] {
19
             closestReqDistance = min(closestReqDistance, distanceBetween
20
21
22
          maxDistancesAtBlocks[i] = max(maxDistancesAtBlocks[i], closestRe
23
24
25
26
     var optimalBlockIdx int
27
     smallestMaxDistance := math.MaxInt32
     for i, currentDistance := range maxDistancesAtBlocks {
28
29
       if currentDistance < smallestMaxDistance {</pre>
30
         smallestMaxDistance = currentDistance
31
          optimalBlockIdx = i
32
33
```

```
package main

type Block map[string]bool

func ApartmentHunting(blocks []Block, reqs []string) int {
    // Write your code here.
    return -1
}
```



Run or submit code when you're ready.