Our Solution(s) Run Code

Solution 1 Solution 2

```go
// 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
// and r is the number of requirements.
func ApartmentHunting(blocks []Block, reqs []string) int {
  maxDistancesAtBlocks := make([]int, len(blocks))
  for i := range blocks {
    maxDistancesAtBlocks[i] = -1
    for _, req := range reqs {
      closestReqDistance := math.MaxInt32
      for j := range blocks {
        if blocks[j][req] {
          closestReqDistance = min(closestReqDistance, distanceBetween
        }
      }
      maxDistancesAtBlocks[i] = max(maxDistancesAtBlocks[i], closestRe
    }
  }

  var optimalBlockIdx int
  smallestMaxDistance := math.MaxInt32
  for i, currentDistance := range maxDistancesAtBlocks {
    if currentDistance < smallestMaxDistance {
      smallestMaxDistance = currentDistance
      optimalBlockIdx = i
    }
  }
```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```go
package main

type Block map[string]bool

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

```

Our Tests

Custom Output Submit Code

Run or submit code when you're ready.