

AlgoExpert

Quad Layout

Go

12px

Sublime

Monokai

00:00:00

PromptScratchpadOur Solution(s)Video Explanation

Run Code

Solution 1Solution 2

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

2

3package main

4

5import "math"

6

7type Block map[string]bool

8

9// O(br) time | O(br) space - where b is the number of blocks

10// and r is the number of requirements.

11func ApartmentHunting(blocks []Block, reqs []string) int {

12 minDistancesFromBlocks := [][]int{

13 for _, req := range reqs {

14 minDistancesFromBlocks = append(minDistancesFromBlocks,

15 getMinDistances(blocks, req))

16 }

17 maxDistancesAtBlocks := getMaxDistancesAtBlocks(blocks, minDistancesFromBlocks)

18

19 var optimalBlockIdx int

20 smallestMaxDistance := math.MaxInt32

21 for i, currentDistance := range maxDistancesAtBlocks {

22 if currentDistance < smallestMaxDistance {

23 smallestMaxDistance = currentDistance

24 optimalBlockIdx = i

25 }

26 }

27 return optimalBlockIdx

28 }

29

30func getMinDistances(blocks []Block, req string) []int {

31 minDistances := make([]int, len(blocks))

32 closestReq := math.MaxInt32

33 for i := range blocks {

34 if val, found := blocks[i][req]; found && val {

35 closestReq = i

36 }

37 minDistances[i] = distanceBetween(i, closestReq)

38 }

39

40 for i := len(blocks) - 1; i >= 0; i-- {

41 if val, found := blocks[i][req]; found && val {

42 closestReq = i

43 }

44 minDistances[i] = min(minDistances[i], distanceBetween(i, closestReq))

45 }

46 return minDistances

47 }

48

49func getMaxDistancesAtBlocks(blocks []Block, minDistancesFromBlocks [][]int) []int {

50 maxDistancesAtBlocks := make([]int, len(blocks))

51 for i := range blocks {

52 minDistancesAtBlock := []int{

53 for _, distances := range minDistancesFromBlocks {

54 minDistancesAtBlock = append(minDistancesAtBlock, distances[i])

55 }

56 maxDistancesAtBlocks[i] = max(minDistancesAtBlock)

57 }

58 return maxDistancesAtBlocks

59 }

60

61func distanceBetween(a, b int) int {

62 if a > b {

63 return a - b

64 }

65 return b - a

66 }

67

68func min(a, b int) int {

69 if a < b {

70 return a

71 }

72 return b

73 }

74

75func max(array []int) int {

76 if len(array) == 0 {

77 return 0

78 }

79

80 max := array[0]

81 for i := 1; i < len(array); i++ {

82 if array[i] > max {

83 max = array[i]

84 }

85 }

86 return max

87 }

88

