AlgoExpert Quad Layout Go 12px Sublime Monokai 00:00:00

Prompt Scratchpad Our Solution(s) Video Explanation Run Code

```
Solution 1
 1\ \ //\ \mbox{Copyright @ 2020 AlgoExpert, LLC.} All rights reserved.
    package main
    import "strings"
    type counts struct {
      \times int
 9
      y int
10
    }
11
12
    // O(n^2 + m) time | O(n + m) space
13
     func PatternMatcher(pattern string, str string) []string {
14
      if len(pattern) > len(str) {
15
        return []string{}
16
17
      pattern, switched := getNewPattern(pattern)
18
      count, firstY := getCountsAndFirstYPos(pattern)
19
      if count.y != 0 {
20
         for lenx := 1; lenx < len(str); lenx++ \{
21
           totalLeny := len(str) - lenx*count.x
            \textbf{if len}(\texttt{str}) \ <= \ \texttt{lenx*count.x} \ | | \ \texttt{totalLeny%count.y} \ != \ \textbf{0} \ \{ \\
22
23
24
25
           leny := totalLeny / count.y
26
          yindex := firstY * lenx
27
           x, y := str[:lenx], str[yindex:yindex+leny]
28
           potentialMatch := doReplace(pattern, x, y, count)
29
           if str == potentialMatch {
30
            if !switched {
31
               return []string{x, y}
32
             return []string{y, x}
33
34
35
36
      } else {
37
        if len(str)%count.x == 0 {
          lenx := len(str) / count.x
38
39
           potentialMatch := strings.Repeat(x, len(pattern))
40
41
           if str == potentialMatch {
            if !switched {
43
              return []string{x, ""}
44
45
             return []string{"", x}
46
47
48
49
50
      return []string{}
51
52
53
     func doReplace(pattern, x, y string, count counts) string {
      result := make([]byte, 0)
54
55
       for _, r := range pattern {
56
        if r == 'x' {
57
          result = append(result, []byte(x)...)
59
          result = append(result, []byte(y)...)
60
61
62
      return string(result)
63
64
65
     \begin{tabular}{ll} func & getNewPattern(pattern string) & (string, bool) & ( \end{tabular} \label{table}
66
      if pattern[0] == 'x' {
67
        return pattern, false
68
69
      runes := make([]rune, len(pattern))
70
       for i := range pattern {
71
        if pattern[i] == 'x' {
72
          runes[i] = 'y'
73
         } else {
74
           runes[i] = 'x'
75
76
77
      return string(runes), true
78
80 func getCountsAndFirstYPos(pattern string) (counts, int) {
81
      firstY := strings.Index(pattern, "y")
82
      count := counts{}
83
      for _, r := range pattern {
       if r == 'x' {
84
         count.x += 1
85
```