

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1

Solution 2

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 package main
4
5 import "math"
6
7 // O(n^3 + m) time | O(n + m) space - where n is the number of digits
8 // in Pi and m is the number of favorite numbers.
9 func NumbersInPi(pi string, numbers []string) int {
10     numbersTable := map[string]bool{}
11     for _, number := range numbers {
12         numbersTable[number] = true
13     }
14     minSpaces := getMinSpaces(pi, numbersTable, map[int]int{}, 0)
15     if minSpaces == math.MaxInt32 {
16         return -1
17     }
18     return minSpaces
19 }
20
21 func getMinSpaces(pi string, numbersTable map[string]bool,
22     cache map[int]int, idx int) int {
23     if idx == len(pi) {
24         return -1
25     } else if val, found := cache[idx]; found {
26         return val
27     }
28     minSpaces := math.MaxInt32
29     for i := idx; i < len(pi); i++ {
30         prefix := pi[idx : i+1]
31         if _, found := numbersTable[prefix]; found {
32             minSpacesInSuffix := getMinSpaces(pi, numbersTable, cache, i+1)
33             minSpaces = min(minSpaces, minSpacesInSuffix+1)
34         }
35     }
36     cache[idx] = minSpaces
37     return minSpaces
38 }
```

Solution 1

Solution 2

Solution 3

```
1 package main
2
3 func NumbersInPi(pi string, numbers []string) int {
4     // Write your code here.
5     return -1
6 }
7
```

Our Tests

Custom Output

Submit Code

1

2

3

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

1

2

3

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

```
10 return 0
11
12 def isPrime(n):
13     return isPrimeRec(n, 2)
14
15 def isPrimeRec(n, p):
16     if p >= n:
17         return True
18     if n % p == 0:
19         return False
20     return isPrimeRec(n, p + 1)
21
22 def isPrime(n):
23     return isPrimeRec(n, 2)
24
25 def isPrime(n):
26     return isPrimeRec(n, 2)
27
28 def isPrime(n):
29     return isPrimeRec(n, 2)
```

Run or submit code when you're ready.