

PromptScratchpadOur Solution(s)Video Explanation

Run Code

Solution 1Solution 2

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 #include <set>
4 #include <unordered_map>
5 #include <algorithm>
6 #include <climits>
7 #include <vector>
8 using namespace std;
9
10 int getMinSpaces(string pi, set<string> numbersTable,
11                 unordered_map<int, int> *cache, int idx);
12
13 // O(n^3 + m) time | O(n + m) space - where n is the number of digits in Pi and
14 // m is the number of favorite numbers
15 int numbersInPi(string pi, vector<string> numbers) {
16     set<string> numbersTable;
17     for (string number : numbers) {
18         numbersTable.insert(number);
19     }
20     unordered_map<int, int> cache;
21     int minSpaces = getMinSpaces(pi, numbersTable, &cache, 0);
22     return minSpaces == INT_MAX ? -1 : minSpaces;
23 }
24
25 int getMinSpaces(string pi, set<string> numbersTable,
26                 unordered_map<int, int> *cache, int idx) {
27     if (idx == pi.length())
28         return -1;
29     if (cache->find(idx) != cache->end())
30         return cache->at(idx);
31     int minSpaces = INT_MAX;
32     for (int i = idx; i < pi.length(); i++) {
33         string prefix = pi.substr(idx, i + 1 - idx);
34         if (numbersTable.find(prefix) != numbersTable.end()) {
35             int minSpacesInSuffix = getMinSpaces(pi, numbersTable, cache, i + 1);
36             // Handle int overflow.
37             if (minSpacesInSuffix == INT_MAX) {
38                 minSpaces = min(minSpaces, minSpacesInSuffix);
39             } else {
40                 minSpaces = min(minSpaces, minSpacesInSuffix + 1);
41             }
42         }
43     }
44     cache->insert({idx, minSpaces});
45     return cache->at(idx);
46 }
```