Our Solution(s) Run Code

Solution 1 Solution 2

```cpp
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

#include <set>
#include <unordered_map>
#include <algorithm>
#include <climits>
#include <vector>
using namespace std;

int getMinSpaces(string pi, set<string> numbersTable,
                 unordered_map<int, int> *cache, int idx);

// O(n^3 + m) time | O(n + m) space - where n is the number of digits
// m is the number of favorite numbers
int numbersInPi(string pi, vector<string> numbers) {
  set<string> numbersTable;
  for (string number : numbers) {
    numbersTable.insert(number);
  }
  unordered_map<int, int> cache;
  int minSpaces = getMinSpaces(pi, numbersTable, &cache, 0);
  return minSpaces == INT_MAX ? -1 : minSpaces;
}

int getMinSpaces(string pi, set<string> numbersTable,
                 unordered_map<int, int> *cache, int idx) {
  if (idx == pi.length())
    return -1;
  if (cache->find(idx) != cache->end())
    return cache->at(idx);
  int minSpaces = INT_MAX;
  for (int i = idx; i < pi.length(); i++) {
    string prefix = pi.substr(idx, i + 1 - idx);
```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```cpp
#include <vector>
using namespace std;

int numbersInPi(string pi, vector<string> numbers) {
  // Write your code here.
  return -1;
}

```

Our Tests

Custom Output Submit Code

Run or submit code when you're ready.