

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1Solution 2

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 #include <vector>
4 #include <climits>
5 using namespace std;
6
7 // O(n^2) time | O(n^2) space
8 int palindromePartitioningMinCuts(string s) {
9     vector<vector<bool>> palindromes(s.length(), vector<bool>(s.length()
10     for (int i = 0; i < s.length(); i++) {
11         palindromes[i][i] = true;
12     }
13     for (int length = 2; length < s.length() + 1; length++) {
14         for (int i = 0; i < s.length() - length + 1; i++) {
15             int j = i + length - 1;
16             if (length == 2) {
17                 palindromes[i][j] = (s[i] == s[j]);
18             } else {
19                 palindromes[i][j] = (s[i] == s[j] && palindromes[i + 1][j - 1]
20             }
21         }
22     }
23     vector<int> cuts(s.length(), INT_MAX);
24     for (int i = 0; i < s.length(); i++) {
25         if (palindromes[0][i]) {
26             cuts[i] = 0;
27         } else {
28             cuts[i] = cuts[i - 1] + 1;
29             for (int j = 1; j < i; j++) {
30                 if (palindromes[j][i] && cuts[j - 1] + 1 < cuts[i]) {
31                     cuts[i] = cuts[j - 1] + 1;
32                 }
33             }
34         }
35     }
36     return cuts[s.length() - 1];
37 }
```

Solution 1Solution 2Solution 3

```
1 #include <vector>
2 using namespace std;
3
4 int palindromePartitioningMinCuts(string string) {
5     // Write your code here.
6     return -1;
7 }
8
```

Our Tests

Custom Output

Submit Code

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 #include <vector>
4 #include <climits>
5 using namespace std;
6
7 // O(n^2) time | O(n^2) space
8 int palindromePartitioningMinCuts(string s) {
9     vector<vector<bool>> palindromes(s.length(), vector<bool>(s.length()
10     for (int i = 0; i < s.length(); i++) {
11         palindromes[i][i] = true;
12     }
13     for (int length = 2; length < s.length() + 1; length++) {
14         for (int i = 0; i < s.length() - length + 1; i++) {
15             int j = i + length - 1;
16             if (length == 2) {
17                 palindromes[i][j] = (s[i] == s[j]);
18             } else {
19                 palindromes[i][j] = (s[i] == s[j] && palindromes[i + 1][j - 1]
20             }
21         }
22     }
23     vector<int> cuts(s.length(), INT_MAX);
24     for (int i = 0; i < s.length(); i++) {
25         if (palindromes[0][i]) {
26             cuts[i] = 0;
27         } else {
28             cuts[i] = cuts[i - 1] + 1;
29             for (int j = 1; j < i; j++) {
30                 if (palindromes[j][i] && cuts[j - 1] + 1 < cuts[i]) {
31                     cuts[i] = cuts[j - 1] + 1;
32                 }
33             }
34         }
35     }
36     return cuts[s.length() - 1];
37 }
```

```
1 #include <vector>
2 using namespace std;
3
4 int palindromePartitioningMinCuts(string string) {
5     // Write your code here.
6     return -1;
7 }
8
```

```
1 def test_test_case_01():
2     """ 0. test function with missing function """
3
4     assert test_test_case_01() == 0
5
6 def test_test_case_02():
7     """ 1. test function with missing function """
8
9     assert test_test_case_02() == 0
10
11 def test_test_case_03():
12     """ 2. test function with missing function """
13
14     assert test_test_case_03() == 0
15
16 def test_test_case_04():
17     """ 3. test function with missing function """
18
19     assert test_test_case_04() == 0
20
21 def test_test_case_05():
22     """ 4. test function with missing function """
```

Run or submit code when you're ready.