In this article, we are going to solve another problem of finding the minimum number of elements to be removed so that pairwise consecutive elements are the same. We can say it is a type of string manipulation, as we are doing operations on strings. Let’s try to understand and learn how this problem is solved logically and programmatically.

In this question, a string is given which consists of different characters. Some of the characters are repeated in pairs. We need to find the minimum number of characters to be removed so that the pairwise consecutive elements are present in the final string. The main task is to count the characters which are not in pairs in the given string. It might be confusing to understand the problem statement, but don’t worry, I am here to help with examples to clarify the question and problem statement in detail.

Let’s try to understand with a few examples.

**Example 1:**

Suppose, the input string is “44522255”.

In this string three pairs are present i.e. “44”, “22”, and “55”. So we have to keep these characters in the string and remove all the remaining characters of the string. So, in this case the total number of characters which are going to be removed are two i.e. “5” and “2”. Then print the total number of characters which are going to be removed as output.

So the output of this example is 2.

**Example 2:**

Try to understand with one more example, it will clear rest for your doubt.

Suppose, the input string is “11344”.

In this string two pairs are present i.e. “11”, and “44”. So we have to keep these characters in the string and remove all the remaining characters of the string. So, in this case the total number of characters which are going to be removed is one i.e. “3”. Then print the total number of characters which are going to be removed as output.

So the output of this example is 1.

Hope, the above example makes the problem statement clear to you. Now, let’s look at the naïve approach to solve the problem.

## **Approach 1: Without Recursion**

In this approach we simply use a function to solve this problem without using recursion.

### **Algorithm**

**Step 1:** Create a function countConsecutive which takes a string as input and returns an integer.

**Step 2:** Create a variable count which stores the total number of removed elements and assign it with zero.

**Step 3:** Using a for loop, iterate up to the length of the string and check for the pair.

**Step 4:** If no pair is available, increase the count and execute the loop.

**Step 5:** At last, return the count as output.

### **Explanation**

Let’s try to understand the approach and see how things work. In this approach we simply define a function countConsecutive which will take input string as parameter and return the total number of elements which are going to be removed to solve this problem. It uses the simple approach by iterating through each character of the string and checking whether they are in pairs or not. If they are not in pairs they should be removed..

### **Program**

|  |
| --- |
| #include <iostream>  using namespace std;  int countConsecutive(string s)  {  int count = 0;  for (int i = 0; i < s.size(); i++) {  if (s[i] == s[i + 1])  i++;  else  count++;  }  return count;  }  int main()  {  string str = "11344";  cout << countConsecutive(str);  return 0;  } |

**Output**

|  |
| --- |
| 1 |

## We see the non-recursive approach to solve the given problem statement now it time to use recursion and understand how we can solve it using recursion.

## **Approach 2: With Recursion**

### **Algorithm**

**Step 1:** Create a recursive function countConsecutive which takes a string and an integer i as input parameters and returns an integer value.

**Step 2:** Set the base condition, return 0 if the value of is equal to length of the string.

**Step 3:** Check if the adjacent characters of the string are equal or not.

**Step 4:** If they are equal, call the countConsecutive function recursively with incrementing the value of i by 2 and return it.

**Step 5:** If they are not equal, call the countConsecutive function recursively with incrementing the value of i by 1 and return it after adding 1 to the value.

### **Explanation**

This is the recursive approach to solve this problem. In this approach we are recursively calling the function countConsecutive to get the total number of characters which should be deleted to solve this question. In this function we are taking the string and the zeroth index as parameters and it will increase the index and check for the pair, again and again until the index is equal to the length of the string.

### **Program**

|  |
| --- |
| #include <iostream>  using namespace std;  int countConsecutive(string s, int i)  {  if (i == s.length())  return 0;  if (s[i] == s[i + 1])  return countConsecutive(s, i + 2);  return 1 + countConsecutive(s, i + 1);  }  int main()  {  string str = "112554";  cout << countConsecutive(str, 0);  return 0;  } |

**Output**

|  |
| --- |
| 2 |

## **Conclusion**

In this article, we learn and solve to find the minimum number of elements to be removed so that pairwise consecutive elements are the same. We understand two approaches to solve this problem with better understanding. Hope you liked this article and understand the concept well.