Problem: UVA Live 6823

Count No. of substrings of a given string that is dividable by 3.

Input	Output
130	3
303	6
2014	2
2012	4

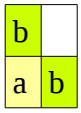
Explanation:
Case 1:
3
0
30
Case 2:
3
0
3
30
03
303
Case 3:
0
201
Case 4:
0
12
201
012

Solution Approach:

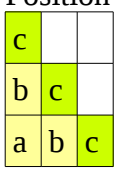
At first let's look how to calculate substrings. Suppose we have a string '**abc**' and we have to find all substrings of '**abc**'. So at:

Position 0

Position 1



Position 2



So we can easily see that at every position/character this character can be concatenated with previous substrings and this character itself is a substring. So **no. of substrings** that ends at a particular position is equal to **no. of substrings ends in previous position+ 1**. Now we cumulatively add this value at every position to get total no. of substring for the given string. Now in our case, we can say a substring ends at a particular position is divisible by 3 if it holds residue of '0' (modulo 3) at this position. So we will go to every position and calculate **no. of substrings = x with a residue of 0 (modulo 3)** then we will add this value(x) cumulatively to calculate total no. of substrings that ends with residue of 0 (modulo 3) that is divisible by 3.

Let's take input as a string '**str**', at each character we have residue of '**r**', that is **r = (str[i] - '0')%3** that is value of **r is in the range 0, 1 and 2**. Suppose we have two array namely **residue** and **previous_residue**. **residue[i]** holds number of substrings that ends in current character with a residue i, and **previous_residue[i]** holds number of substrings that ends in previous character with residue i. So if we add number of substrings that ends at a character/position with a residue 0, we will have our solution by cumulatively summing them all.

Now let's we have residue of '0' for the current character that is (str[i] - '0') % 3 = 0, then we have now an extra substring (the current character) with residue 0 and no. of substrings ends in position i with residue 1 and residue 2 will be same. So we can write:

```
residue[0] = previous_residue[0] + 1.
residue[1] = previous_residue[1].
residue[2] = previous_residue[2].
```

If we have residue '1' for the current character then the number of substrings ends at previous character with residue 0 will now have residue 1, number of substrings with residue 1 at previous character will have residue 2, and number of substrings with residue 2 will have now residue 0. And we have now a an extra substring (the current character) with residue 1. So we can write:

```
residue[0] = previous_residue[2] .
residue[1] = previous_residue[0] + 1.
residue[2] = previous_residue[1].
```

If we have residue '2' for the current character then the number of substrings ends at previous character with residue 0 will have residue 2, number of substrings with residue 1 at previous character will have residue 0, and number of substrings with residue 2 will have now residue 1. And we have now a an extra substring (the current character) with residue 2. So we can write:

```
residue[0] = previous_residue[1] .
residue[1] = previous_residue[2] .
residue[2] = previous_residue[0] + 1.
```

Now merging this we can write:

```
residue[0] = previous_residue[(3 – r + 0) % 3].
residue[1] = previous_residue[(3 – r + 1) % 3].
residue[2] = previous_residue[(3 – r + 2) % 3].
```

residue[r] = residue[r] + 1.

Now at every position we just have to count how many substrings ends at this position with **residue 0**, and we take that value included in our answer. And we will just copy our '**residue**' array to our '**previous_residue**' array for using in next step. So the pseudo code can be look like this:

```
total = 0;
residue[3] = {0, 0, 0};
previous_residue[3] = {0, 0, 0};

for ( i = 0 ; i<str.length; i++)
{
        r = (str[i] - '0') % 3;

        for( j = 0; j<3; j++)
                residue[j] = previous_residue[ (3 – r + j) % 3 ];

        residue[r] = residue[r] + 1;

        total += residue[0];

        for( j = 0; j<3; j++)
                previous_residue[j] = residue[j];

}
print( total );
```