امیرعلی فرازمند

99522329

رئیس سارا از او خواسته متنی را ادیت بزند. طبق دستور رئیس تک تک کلمات(هر رشته متنی که اسپیس در آن نباشد) باید فانتزی باشد. هر رشته را فانتزی می گویند که ابتدا به تعداد صفر یا بیشتر حروف بزرگ و سپس به تعداد صفر یا بیشتر حروف کوچک باشد. برای اینکه کلمه فانتزی شود، سارا می‌تواند در هر مرحله یک حرف را بدون تغییر مکان به حالت دیگر آن بنویسد به این معنا که یک حرف کوچک را به بزرگ یا یک حرف بزرگ را به کوچک تبدیل کند.

سارا به این سوال علاقه مند شد: کمترین تعداد مراحلی که میتوان یک رشته را فانتزی کرد چقدر است؟ به سارا کمک کنید جواب سوال خود را بیابد.

توجه: برای این سوال چرایی درستی راه حل را بیان و پیچیدگی زمانی و حافظه الگوریتم خود را بدست آورید.

Example 1 :

Input :

PRuvetSTAaYA

Output :

5

کد استفاده شده:

```cpp
int main()
{
    string str;
    cin >> str;
    int n = str.length(), min_change = 9999999;
    str = str.insert(n, "%");
    str = str.insert(0, "%");
    unsigned int first_smalls[n + 2] = {}, last_caps[n + 2] = {};
    for (int i = 1; i <= n; i++)
    {
        first_smalls[i] = first_smalls[i - 1];
        if (str[i] >= 'a' && str[i] <= 'z')
            first_smalls[i]++;
    }
    for (int i = n; i >= 1; i--)
    {
        last_caps[i] = last_caps[i + 1];
        if (str[i] >= 'A' && str[i] <= 'Z')
            last_caps[i]++;
    }
    for (int i = 1; i <=n; i++){
        if (first_smalls[i]+last_caps[i]<min_change)
            min_change = first_smalls[i]+last_caps[i];
    }
    cout<<min_change-1<<endl;
    return 0;
```

الگوریتم به این شکل عمل میکند که به سر و ته رشته داده شده یک کرکتر % اضافه میکند. از راست به چپ شروع میکند به افزایشی شمردن حروف بزرگ ،و این عمل را

تکرار میکند و اعداد حاصل را در آرایه ای ذخیره میکند. همین کار را ازآخر به اول رشته و با حرو ف کوچک تکرار میکند. برای مثال برای رشته ی PRuvetSTAaYA داریم:

String: %PRuvetSTAaYA%

First_smalls: 00012344445550

Last_caps: 07655555432210

حال کل رشته را طی میکنیم و جایی که مجموع first_smalls[i], last_caps[i] اش از بقیه کاراکتر ها کمتر بود را در متغیری برای مینیمم تغیرات ذخیره میکنیم. خانه ی انتخاب شده first_smalls[i] تا از خانه های قبلش باید بزرگ شوند و last_caps[i] تا از کرکتر های بعئش باید کوچک شوند. خود کرکتر انتخاب شده چون در مرز هست و هنگامی که سمت راست و چپش درست شده باشند نیازی به تغیرش نیس چه حرف بزرگ باشد چه کوچک ، پس یک واحد کمتر از مینیمم پیدا شده نیاز است برای اینکه رشته را درست کرد.

پیچیدگی زمانی(n را طول رشته در نظر بگیریم):

3 بار رشته را کامل طی کردیم پس:

O(n)

پیچیدگی حافظه ای:

علاوه بر رشته به طول n 2 رشته دیگر به طول n+2 نیاز است پس از O(n) است.