Problem D – Matchmaker

We need maximum numbers of matches between letters, and for a match to be successful, we will need the same letter twice. Hence, if a letter has even count, all of its letters can be used as a match. Hence, the main goal is to try to perform operations such that the count of all letters become even.

So for all letters ‘a’ to ‘z’, we will maintain a prefix sum to count the number of occurences of It between l and r. Now we need to make all of the odd counts as even. So for this there will be exactly odd / 2 operations. Why ? Lets two counts as 7 and 9. Just two random odds. Now I will replace one letter with the other. Hence first count decreases and second count increases. So they become 8 and 8 and now both are even.

If the total number of letters between l and r are itself odd, then exactly 1 count will always be odd u cant change it. So the final answer is just count / 2

Implementation

```
void solve() {
        int n, q;
        cin >> n >> q;
        string s;
        cin >> s;
        vvi count(26, vi(n, 0));
        for(int i=0;i<n;i++) count[s[i] - 97][i] = 1;
        for(int i=0;i<26;i++) for(int j=1;j<n;j++) count[i][j] += count[i][j - 1];
        for(int i=0;i<q;i++) {
                int l, r;
                cin >> l >> r;
                l--;
                r--;
                int odds = 0;
                for(int j=0;j<26;j++) {
                        int cnt = count[j][r];
                        if(l - 1 >= 0) cnt -= count[j][l - 1];
                        if(cnt % 2 != 0) odds++;
                }
                cout << odds / 2 << "\n";
        }
}
```