ALPS 7월 내부대회 E번 풀이

sponsored by NAVER D2

민폐 문자

> ALPS 2020년 7월 내부대회 E번
> 출제자: 최용욱

풀이

많은 분들이 deque를 사용하거나, stack 또는 queue를 변형하여 사용하셨는데,
사실 제가 원했던 정해는 수학이였습니다 ㅋㅋㅋ

만약 A@B@C@D@E 라고 해봅시다.
여기서의 A, B, C, D, E는 민폐 문자들로 split된 부분 문자열을 뜻합니다.

그리고 A'는 A라는 부분 문자열이 뒤집어진 형태라고 하겠습니다.

그럼 우리 함께 민폐 문자를 없애 봅시다.

1. A'B@C@D@E
2. B'AC@D@E
3. C'A'BD@E
4. D'B'ACE

이렇게 됩니다.

이번에는 A@B@C@D 로 똑같이 해보겠습니다.

1. A'B@C@D
2. B'AC@D
3. C'A'BD

보면 두 번째 결과로 나온 문자열이, 첫 번째 문자열을 처리하다가 3번에 나온 앞 문자열과 같다는 것을 알 수 있습니다.

어느정도 해보면, 민폐 문자의 개수가 홀수인지 짝수인지에 따라서 홀/짝이 뒤집어질지 안뒤집어질지 결정되고,
순서는 항상 홀/짝의 뒤에서부터 + 양끝에서부터 시작하기 때문에 규칙을 유추하는 것이 가능합니다.

간단하네요!

딱히 복잡한 문제도 아니기 때문에, 제가 보여드릴 풀이는 간단하게 비트를 이용해서 제 문제를 푸는 코드를 올려드리고자 합니다.

혹시 비트를 잘 다루지 못해보신 분들은 한 번 보시고 아 이렇게 사용할 수 있구나라고만 생각하시면 될 것 같습니다.

정해 코드

```
#include <algorithm>
#include <cstdio>
#include <iostream>
#include <string>
```

```cpp
using namespace std;

const int MAX = 1e6 + 10;
string ss[MAX];

int main() {
    ios_base::sync_with_stdio(0);
    cin.tie(0);

    int n, cnt = 0;
    cin >> n;

    char tmp;
    for (int i = 0; i < n; i++) {
        cin >> tmp;
        if (tmp == '@') {
            cnt++;
        } else {
            ss[cnt].push_back(tmp);
        }
    }

    int idx = cnt - 1;
    while (idx > -1) {
        reverse(ss[idx].begin(), ss[idx].end());
        cout << ss[idx];
        idx -= 2;
    }

    idx = (cnt + 1) % 2 ? 0 : 1;
    while (idx <= cnt) {
        cout << ss[idx];
        idx += 2;
    }

    return 0;
}
```