{{ self.title() }}

题目简述

给定 $2n$ 个格子，每个格子为白色或黑色。通过恰好 n 次区间反色使每个格子均为白色，其中每个格子必须恰好为某个区间的左端点或右端点。求方案数。

子任务设计

前面的 40 分是给爆搜及优化的。不确定会不会有选手学傻，所以加了一档 $n \leq 2000$ 的。

算法1：搜索

直接搜索每次选择哪个区间进行翻转，可以拿到 20 分。

一个小优化是，由于最终每个格子的颜色与操作进行的顺序无关，所以只要搜索每次用哪个格子与当前编号最小的格子进行配对即可。最后将方案数乘上 $n!$ 即可。注意求最终状态时要用差分（即区间修改用单点修改打标记，最后求一遍前缀和，这里是用异或来进行修改及求和）的思想来减小复杂度。

算法2：正解

从差分入手，我们可以发现最终状态只与每个格子是某次操作区间的左端点还是右端点有关；跟这个格子跟另外哪个格子配对，在什么时候进行操作无关。

显然第 1 个格子一定是左端点。由此开始逐个递推，如果前缀和的颜色与当前颜色相同，则当前格子为右端点（不影响当前格子的颜色，但影响之后格子的颜色），否则为左端点（影响当前格子及之后格子的颜色）。我们在右端点的时候统计方案数：每个右端点对答案的贡献为乘上当前未配对的左端点的个数。最后输出时同样乘上 $n!$ 即可。

注意判各种无解情况，如不能凭空出现右端点、出现没有匹配上的左端点等。

总复杂度为 $O\left(\sum n\right)$，可以通过本题。

```cpp
#include <cstdio>
using namespace std;
#define MOD 1000000007
char s[200010];
int main(){
    register int T, n, n2, i, ans, cnt, pre, now;
    scanf("%d", &T);
    while (T--) {
        scanf("%d", &n);
        n2 = n * 2;
        scanf("%s", s + 1);
        if (s[1] == 'W' || s[n2] == 'W') {
            puts("0");
            continue;
        }
        ans = cnt = 1, pre = 0;
        for (i = 2; i < n2; ++i) {
            if (s[i - 1] != s[i]) now = pre;
```

```
            else now = pre ^ 1;
            //now: 0 = [, 1 = ]
            if (now) {
                ans = 1ll * ans * cnt % MOD;
                --cnt;
            }
            else ++cnt;
            pre = now;
        }
        if (cnt != 1) {
            puts("0");
            continue;
        }
        for (i = 2; i <= n; ++i) ans = 1ll * ans * i % MOD;
        printf("%d\n", ans);
    }
    return 0;
}
```