3. 회문

읽는 순서를 뒤에서 시작하여도 앞에서부터 읽는 것과 똑같은 문장, 혹은 문자열을 회문이라고 한다. 문자의 집합 {A, B, C, D}를 알파벳으로 삼는 언어가 있다고 가정해보자. ABBCDAA는 이 언어에 속하는 문자열의 한 예이다. 이 때 문자열의 부분순서라 함은 A, B, ABC, ABBA, ACDAA, ACA, BD, BDA 등이 될 수 있다. 이 예시 중에서 앞에서부터 읽거나 뒤에서부터 읽어도 같은 부분순서는 A, B, ABBA와 ACA, 이 넷이다.

이와 같이 {A, B, C, D}로 이루어진 문자열에는 부분순서가 있을 것이다. 이 부분순서들 중에서 회문의 조건을 만족함과 동시에 가장 긴 부분순서의 길이를 구하는 프로그램을 작성하라. 알고리즘은 최대한 효율적으로 작성하라. 10개의 케이스 모두를 수행한 알고리즘의 총 수행 시간이 10초를 넘지 않아야 한다. 채점을 위한 컴파일 시에 최적화 옵션은 쓰지 않는다.

[입력]

입력 파일에는 10 개의 테스트 케이스가 주어진다. 각 케이스는 2 줄로 이루어진다. 첫 줄에는 문자열의 길이가 주어지고, 다음 줄에는 A, B, C, D로 이루어진 문자열(길이는 10 이상 5000 이하)이 주어진다. 입력파일의 이름은 “input3.txt”이다.

[출력]

각 테스트 케이스에 대해서, 케이스의 번호를 "#x" 의 형식으로 출력한 후(여기서 x는 테스트 케이스 번호), 공백을 하나 둔 다음 주어진 케이스에서 입력된 문자열에 대해 회문인 최장 부분순서의 길이를 출력한다.

[예제]

입력 (input3.txt)

```
10                                                    ← 1번 케이스
ABBCBDDBCC
20                                                    ← 2번 케이스
ADDDCBCABDCACABBAAAA
…
```

출력 (output3.txt)

```
#1 6
#2 10
…
```