기억 압축 (Limited Memory)

국제 정보 올림피아드의 일본 대표로 뽑힌 JOI양은 정보 처리 기술을 높이기 위해서, 정보 올림피아드 일본 위원회 K이사장에게서 다음과 같은 과제를 받았다.

K이사장은, JOI양이 모르게, 어떤 문자열 S를 수첩에 썼다. S는 '〈', '〉', '[', ']' 의 네 종류의 문자로 되어있다. K이사장은 JOI양에게 과제의 내용과 S의 길이가 적힌 프린트를 건넸다. JOI양의 과제는, S가 좋은 문자열인지 아닌지를 판단하는 것이다. 좋은 문자열은 다음과 같이 정의된다:

- 빈 문자열(즉, 길이 0의 문자열)은 좋은 문자열이다.
- x가 좋은 문자열 일 때, 〈x〉(즉, x를 홑꺾쇠표 〈〉로 감싼 문자열)은 좋은 문자열이다.
- x가 좋은 문자열 일 때, [x](즉, x를 각괄호 []로 감싼 문자열)은 좋은 문자열이다.
- x, y가 좋은 문자열 일 때, xy(즉, x, y를 순서대로 연결시킨 문자열)은 좋은 문자열이다.
- 이상으로 정의되는 문자열만 좋은 문자열이다.

예를 들어, "〈〉[]"나 "[〈〉]〈〉"는 좋은 문자열이고, "〉〈"나 "[〈]〉"는 좋은 문자열이 아니다.

JOI양은, 매일 정오에 최대 1번 K이사장에게 전화를 할 수 있다. 전화로는, 정수 I를 지정해서, K이사장으로부터 문자 S의 I번째 문자를 알 수 있다. 그리고 JOI양은, "이 과제는 메모를 해선 안 된다."라는 중대한 제약이 붙어 있다. JOI는 매일 밤 22시에 자서, 아침 6시에 일어나지만, 수면 중에 기억하고 있는 것은 22비트의 정보뿐이다. 보다 정확히는, 취침 전에 0 이상 2^{22}-1 이하의 정수 한 개만 기억하고, 다음 날은 기억한 정수를 바탕으로 해서 과제를 해야 한다. 다만 S의 길이는 프린트에 써져 있기 때문에, 언제든지 참고 할 수 있다. JOI양은 취침 전에 정수 1개를 기억하는 대신 S가 좋은 문자열인지 아닌지를 K이사장에게 답변해도 된다. 이 경우에 과제는 종료되고, 정답인지 오답인지를 판정한다. 다만, 과제의 시작부터 15000일 이내에 전자 메일을 보내지 못한 경우는, 오답으로 판정된다.

문제

JOI양의 전략을 구현해서, 위의 과제를 해결할 수 있는, 프로그램을 작성하라.

구현의 상세 사항

당신은 JOI양의 전략을 구현한 1개의 프로그램을 작성해야 한다. 프로그램은 `Memory_lib.h`를 `include` 해야 한다.

- `int Memory(int N, int M)`

  이 함수는, S의 길이와 기억하고 있는 정수를 받아서, 그 날 JOI양의 행동에 대응하는 함수이다.
  - 인자 `N`은, 문자열 S의 길이 N이다.
  - 인자 `M`은, 그 전날 자기 전에 기억하고 있는 내용을 나타내는 정수 M이다. 단, 과제가 시작할 때는 M=0을 기억하고 있는 것으로 한다.
  - 이 함수는, 아래에 설명될 함수 Get을 최대 1번 호출 할 수 있다.
  - 이 함수는, 0 이상 2^{22}-1 이하의 정수, 또는 -1 또는 -2를 반환해야 한다. 이 범위 외의 값을 반환할 경우에는, 오답 [1]이 된다.
    * 반환 값이 0 이상 2^{22}-1 이하의 정수일 경우에는, 자기 전에 그 수를 기억하는 것을 의미한다.
    * 반환 값이 -1일 때는, "S는 좋은 문자열이다." 라는 메일을 답장하는 것을 의미한다.
    * 반환 값이 -2일 때는, "S는 좋은 문자열이 아니다." 라는 메일을 답장하는 것을 의미한다.
  - 이 함수는, 인자 `N`, `M`의 값 및 호출한 Get의 값에만 의존하는 것으로 기대된다. 또, 실제 채점프로그램에서 이 함수는 2^{22}×4회 호출된다. 자세한 것은, "채점 절차"와 "중요한 주의"를 참조한다.

프로그램 중에는 다음의 함수를 호출 할 수 있다.

- `char Get(int I)`

이 함수는, `Memory`가 한 번 호출 될 때 마다, 최대 한번 밖에 호출 할 수 없다. 두 번 이상 호출 될 경우에는, 오답 [2]이 된다.

인자 `I`는, $1 \leq I \leq N$을 만족하는 I이어야 한다. 이것을 만족하지 않을 경우 오답 [3]이 된다.

이 함수의 반환 값은, 문자열 S의 I번째 문자를 의미한다.

채점의 방법

채점의 각 입력 데이터는 여러 개의 테스트 케이스이며, 그것들의 문자열 S의 길이는 모두 같은 값 N이다.
채점은 다음과 같이 진행된다. 오답으로 판정된 경우에는 채점 프로그램이 종료된다.

(1) 인자 N, M의 값 및 Get의 반환 값에 따른 함수의 동작을 모두 조사한다. 즉 $0 \leq M \leq 2^{22}-1$을 만족하는 M의 각각에 대해, 다음을 진행한다:

(i) 문자 c를 '〈', '〉', '[', ']'의 각각에 대해 다음을 진행한다.

인자 `N`을 N, 인자 `M`을 M이라 하고 `Memory`를 호출한다. 그 때, `Get`이 호출되면, 문자 c가 반환된다. `Memory`의 반환값을 $m(M,c)$라 한다.

(ii) (i)에서 4번의 `Memory` 호출에 대해서, `Get`을 호출하는지의 여부는 같아야 한다. 그리고 `Get`을 호출한 경우에는, 4번 모두 정수 I를 같은 인자 `I`로 해서 호출해야 하고, `Get`을 호출하지 않은 경우에는, 4번 모두 같은 값을 반환해야 한다. 이것을 만족하지 않는 경우에는 오답 [4]이 된다. `Get`을 호출 했을 때 인자 `I`를 $i(M)$이라 한다. (단, `Get`을 호출하지 않았을 경우 $i(M)$=1 이라 한다.)

(2) 각 테스트 케이스의 문자열 S에 대해서, 문제에서 설명된 과제의 시뮬레이션을 한다. 즉, 다음을 한다.

(i) M=0으로 한다.

(ii) 다음을 반복한다.

(a) c를 S의 $i(M)$번째 문자라고 한다.

(b) M을 $m(M,c)$로 대체한다.

(c) M=-1 이거나 M=-2 인 경우에는, (iii)로 간다.

(d) 이 항목을 15000번 이상 도달한 경우, 오답 [5]이 된다.

(iii) 이하의 경우에는 오답 [6]이 된다.

- S가 좋은 문자열이나, M=-2인 경우.
- S가 좋은 문자열이 아니나, M=-1 인 경우.

(3) 정답이다.

중요한 주의

- 실행 시간 측정 및 메모리 측정의 대상은 "채점의 방법"의 순서 (1)이다. 순서 (1)에서 Memory는 $2^{22} \times 4$번 호출됨에 주의한다.
- 순서 (1)에서, 모든 $2^{22} \times 4$번의 호출에 대해, 오답 [1], [2], [3]이 되거나, 실행시 에러가 발생해선 안됨에 주의한다.

컴파일·실행의 방법

작성한 프로그램을 테스트 할 때는, 채점 프로그램의 샘플이 컨테스트 사이트에서 다운로드 할 수 있는 아카이브에 있다. 이 아카이브에는, 제출해야 하는 파일의 샘플이 들어있다.

채점 프로그램의 샘플은 grader-simple 과 grader-strict 2개가 제공된다. grader-simple 의 파일은 `grader-simple.c` 또는 `grader-simple.cpp` 이고, grader-strict의 파일은 `grader-strict.c` 또는 `grader-strict.cpp` 이다. 예로, 작성한 프로그램이 `Memory.c` 또는 `Memory.cpp` 인 경우, 작성한 프로그램을 grader-simple 이나 grader-strict 로 테스트 할 때에는, 다음의 커맨드를 실행한다.

- C의 경우

```
gcc -O2 -o grader-simple grader-simple.c Memory.c -lm
gcc -O2 -o grader-strict grader-strict.c Memory.c -lm
```

- C++의 경우

```
g++ -std=c++11 -O2 -o grader-simple grader-simple.cpp Memory.cpp
gcc -std=c++11 -O2 -o grader-strict grader-strict.cpp Memory.cpp
```

컴파일이 성공 할 경우에는, `grader-simple` 이나 `grader-strict` 라는 파일이 생성된다.

실제의 채점 프로그램은, grader-simple 이나 grader-strict 와는 다름에 주의해야한다. grader-simple 이나 grader-strict 는 단일 프로세스로 실행된다. 이 프로그램들은 표준 입력으로 입력하고, 표준 출력으로 출력한다.

채점프로그램의 샘플의 개요

grader-simple 은, "채점의 방법" (1)과 같이 처음에 `Memory`를 호출 하지 않고, 당신의 프로그램에 교대로 교환을 함으로써 문제에서 설명된 과제의 시뮬레이션을 한다. "교환의 예"를 참조하라.

grader-strict 는, "채점의 방법"과 마찬가지로 `Memory`를 호출한다.

다음의 사항이 실제 채점 프로그램과 동작이 다름을 주의하라:

- grader-simple 은, "채점의 방법"과 `Memory`를 호출하는 방식이 달라 오답 [4]를 판정하지 않는다.
- **grader-simple 이나 grader-strict 는, 오답 [6]을 판정하지 않고, 대신에 M의 값을 출력한다.**

채점프로그램의 샘플의 입력

grader-simple 이나 grader-strict는 표준 입력으로 다음을 읽어 들인다.

- 첫째 줄에, 정수 N, Q를 공백으로 구분해 입력한다. N은 문자열 S의 길이이고, Q($0 \leq Q \leq 2^{31}-1$)은 테스트 케이스의 개수를 의미한다.
- 다음 Q개의 줄에, 각 테스트 케이스를 나타내는 문자열 S(길이 N)를 입력한다.

채점프로그램의 샘플의 출력

grader-simple 이나 grader-strict는 표준 출력으로 다음을 출력한다.

- "채점의 방법" (2)(iii)에 대한 M의 값(각 테스트 케이스에 대해)를 한 줄로 출력한다.
- 오답 [1], [2], [3], [4], [5]중의 하나로 판정된 경우, 오답의 종류가 "Wrong Answer [1]"과 같이 출력된다. 그 시점에서 프로그램은 종료되고, 그 이후의 출력은 이루어지지 않는다.

제한

모든 입력 데이터는 다음의 조건을 만족한다.

- 1 ≤ (S의 길이) ≤ 100
- S의 각 문자는 '〈', '〉', '[', ']'로 이루어져 있다.

Subtask

Subtask1 [10점]

다음의 조건을 만족한다.

- (S의 길이) ≤ 8

Subtask2 [10점]

다음의 조건을 만족한다.

- (S의 길이) ≤ 14

Subtask3 [5점]

다음의 조건을 만족한다.

- (S의 길이) ≤ 24

Subtask4 [5점]

다음의 조건을 만족한다.

- (S의 길이) ≤ 30

Subtask5 [10점]

다음의 조건을 만족한다.

- S의 각 문자는 '〈', '〉'으로 이루어져 있다.

Subtask6 [60점]

추가 제한 조건이 없다.

함수 호출의 예

채점 프로그램의 샘플이 읽는 입력의 예와, 그에 대응하는 함수 호출 예는 다음과 같다.

입력 예제	함수 호출의 예			
	함수 호출	반환 값	함수 호출	반환 값
4 1 <>[]	Memory(4, 0)			
			Get(1)	
				<
		2015		
	Memory(4, 2015)			
			Get(3)	
				[
		3		
	Memory(4, 3)			
			Get(2)	
				>
		23		
	Memory(4, 23)			
			Get(4)	
				]
		4194303		
	Memory(4, 4194303)			
			Get(3)	
				[
		-1		

다음의 교환이 이루어 질 때, grader-simple은 1을 출력한다.