저울

아미나는 6개의 동전을 가지고 있다. 동전들은 1번 부터 6번까지 번호가 붙어 있다. 그녀는 모든 동전의 무게가 다르다는 것을 알고 있다. 그녀는 모든 동전을 무게 순으로 정렬하려고 한다. 그러기 위해 그녀는 새로운 종류의 저울을 만들었다.

보통의 양팔저울은 두개의 접시가 있고, 그 두 접시에 동전을 놓으면 어느 쪽이 무거운 지를 알려준다.

아미나의 새 저울은 좀더 복잡하다. 저울에는 4개의 팔에 접시가 하나씩 있고, 접시들은 A, B, C, D라고 이름이 붙어 있다. 저울에는 4개의 서로 다른 세팅이 있어서, 세팅에 따라 동전들의 상대적인 무게에 대한 서로 다른 질문들에 대한 대답을 준다. 아미나는 A, B, C의 3개의 접시에 정확히 하나씩의 동전을 올려야 한다. 4번째 세팅에서는, 추가적으로, 접시 D에도 동전을 정확히 하나 올려야 한다.

4개의 세팅에서 각각 대답해 주는 질문은 아래와 같다.

1. A, B, C에 있는 동전 중 어느 것이 가장 무거운가?
2. A, B, C에 있는 동전 중 어느 것이 가장 가벼운가?
3. A, B, C에 있는 동전 중 어느 것의 무게가 중간인가? (무게가 중간이라는 것은 가장 무겁지도, 가장 가볍지도 않다는 것이다.)
4. A, B, C에 있는 동전들 중, D에 있는 것보다 무거운 것들만 고려하라. 그러한 동전들이 있다면, 그 중에 가장 가벼운 것은 무엇인가? D에 있는 것보다 무거운 동전이 없다면, A, B, C에 있는 것들 중 가장 가벼운 것은 무엇인가?

문제

아미나의 6개의 동전을 무게 순으로 정렬하는 프로그램을 작성하라. 프로그램은 아미나의 저울을 이용하여 동전들의 무게를 비교할 수 있다. 프로그램에 여러개의 테스트 케이스들이 주어질 것이며, 각 테스트 케이스는 동전 6개짜리 세트이다.

프로그램은 두개의 함수 `init`과 `orderCoins`를 구현해야 한다. 프로그램이 실행되면, 그레이더는 우선 `init`를 정확히 한번 호출할 것이다. 이 호출에서 테스트 케이스의 개수가 주어지며, 변수들을 초기화할 수 있다. 그 이후, 그레이더는 `orderCoins()`를 테스트 케이스마다 한번씩 호출할 것이다.

- `init(T)`
  - `T`: 프로그램이 실행되는 동안 해결할 테스트 케이스의 수. `T`는 $1, \ldots, 18$ 범위의 자연수이다.
  - 이 함수는 리턴 값이 없다.
- `orderCoins()`

- 이 함수는 테스트 케이스 마다 정확히 한번씩 호출된다.
- 이 함수는 `getHeaviest()`, `getLightest()`, `getMedian()`, `getNextLightest()`의 4개의 함수를 사용하여 아미나의 동전들의 무게 순서를 알아내야 한다.
- 이 함수에서 무게 순서가 파악되면 `answer()` 함수를 호출하여 그 결과를 제출한다.
- `answer()` 함수를 호출한 후, `orderCoins()` 함수는 리턴하여야 한다. 이 함수는 리턴 값이 없다.

당신의 프로그램에서 사용할 수 있는 그레이더 함수는 아래와 같다.

- `answer(W)` — 당신의 프로그램은 이 함수를 이용하여 결과를 제출하여야 한다.
  - `W`: 무게에 따른 동전의 순서를 저장한 크기 6인 배열. `W[0]` 부터 `W[5]` 까지는 가장 가벼운 것 부터 가장 무거운 것 순으로 동전 번호를 저장하고 있어야 한다. 동전 번호는 1 부터 6 까지의 자연수이다.
  - 당신의 프로그램은 이 함수를 테스트 케이스 마다 정확히 한번씩 호출해야 한다. 즉, `orderCoins()` 함수가 한번 실행될 때 정확히 한번의 호출이 있어야한다.
  - 이 함수는 리턴 값이 없다.
- `getHeaviest(A, B, C)`, `getLightest(A, B, C)`, `getMedian(A, B, C)` — 이 함수들은, 순서대로, 아미나의 저울에 있는 3개의 세팅에 해당한다.
  - `A`, `B`, `C`: 각각 A, B, C에 올려진 동전의 번호이다. `A`, `B`, `C`의 세 값은 모두 1 부터 6 까지의 정수들 중 하나라야 하며, 모두 달라야 한다.
  - 각 함수는 `A`, `B`, `C`의 세 값 중 하나를 리턴한다. 즉, 각 함수의 의미에 해당하는 동전의 번호가 리턴된다. 예를 들면, `getHeaviest(A, B, C)`는 주어진 세 개의 동전 번호 중 가장 무거운 것의 번호를 리턴한다.
- `getNextLightest(A, B, C, D)` — 이 함수는 아미나의 저울에 4번째 세팅에 해당한다.
  - `A`, `B`, `C`, `D`: 각각 A, B, C, D에 올려진 동전들의 번호이다. `A`, `B`, `C`, `D`의 네 값은 모두 1 부터 6 까지의 정수들 중 하나라야 하며, 모두 달라야 한다.
  - 이 함수는 `A`, `B`, `C`의 세 값 중 하나를 리턴한다. 리턴되는 값은 저울의 4번째 세팅의 규칙에 따라 결정되는 동전의 번호이다. 즉, 리턴되는 동전의 번호는 A, B, C에 있는 동전들 중, D에 있는 동전보다 무거운 것들 중에서 가장 가벼운 것에 해당한다. 만약, A, B, C에 있는 동전들 중에 D에 있는 것 보다 무거운 것이 없다면, A, B, C에 있는 동전들 중 가장 가벼운 것의 번호가 리턴된다.

채점 방법

이 문제는 부분문제가 없다. 대신, 당신의 점수는 당신의 프로그램이 무게 비교 함수(즉, `getLightest()`, `getHeaviest()`, `getMedian()`, `getNextLightest()`)를 몇번이나 호출하느냐에 따라 결정될 것이다.

당신의 프로그램은 몇차례 실행될 것이며, 각각의 실행에서 여러개의 테스트 케이스가 주어질 것이다. 당신의 프로그램이 실행되는 횟수가 r이라고 하자. 이 수는 테스트 데이터에서 결정된다. 만약 당신의 프로그램이 모든 실행에서 주어진 테스트 케이스들 중 하나에 대해서라도 잘못된 답을 제출한다면 전체에 대해 0점을 받는다. 그렇지 않다면, 다음의 규칙에 따라 각 실행 별로 점수가

주어질 것이다.

아미나의 저울을 이용하여 어떤 동전들이 주어진 경우에도 그 무게에 따른 정렬을 얻을 수 있는 최소의 무게 비교 횟수를 Q라고 하자. 문제를 어렵게 유지하기 위해 Q의 값을 여기에 제시하지는 않는다.

프로그램의 모든 실행에서 주어진 모든 테스트 케이스들 중 가장 많은 무게 비교를 한 경우의 비교 횟수가 어떤 정수 y에 대해 $Q + y$라고 하자. 이제 당신의 프로그램의 실행들 중 하나를 고려해 보자. 그 실행에서 주어진 T개의 테스트 케이스들 중 가장 많은 무게 비교를 한 횟수가 어떤 음아닌 정수 x에 대해 $Q + x$라고 하자. (만약 당신의 프로그램이 모든 테스트 케이스에 대해서 Q보다 작은 횟수의 무게 비교를 했다면 $x = 0$으로 간주한다.) 이때, 주어지는 점수는 $\frac{100}{r((x+y)/5+1)}$의 값을 소수점 이하 세째자리 이하를 버림으로 계산한 값이다.

특히, 당신의 프로그램이 모든 테스트 케이스에 대해서 Q번 이하의 무게 비교를 한다면 100점을 받게 된다.

예제

동전들의 무게 순서가 가벼운 것에서 무거운 것으로 **3 4 6 2 1 5**라고 하자.

호출	리턴	설명
getMedian(4, 5, 6)	6	**6**번 동전이 **4**, **5**, **6** 중에서 중간 무게이다.
getHeaviest(3, 1, 2)	1	**1**번 동전이 **1**, **2**, **3** 중에서 가장 무겁다.
getNextLightest(2, 3, 4, 5)	3	**2**, **3**, **4**번 동전이 모두 **5**번 동전보다 가볍다. 따라서, 그 중 가장 가벼운 동전의 번호(**3**)가 리턴된다.
getNextLightest(1, 6, 3, 4)	6	**1**번과 **6**번 동전이 **4**번 동전보다 무겁다. **1**번과 **6**번 중, **6**번이 더 가볍다.
getHeaviest(3, 5, 6)	5	**5**번 동전이 **3**, **5**, **6** 중 가장 무겁다.
getMedian(1, 5, 6)	1	**1**번 동전이 **1**, **5**, **6** 중 중간 무게이다.
getMedian(2, 4, 6)	6	**6**번 동전이**2**, **4**, **6** 중 중간 무게이다.
answer([3, 4, 6, 2, 1, 5])		프로그램이 이 테스트 케이스의 정답을 찾았다.

Sample grader

Sample grader는 다음과 같은 형식의 입력을 받는다.

- line **1**: T — 테스트 케이스의 수
- lines **2**…$T + 1$: **1** 부터 **6** 까지의 자연수의 순열이 주어짐. 주어지는 순서는 가벼운 것 부터 무거운 것 순으로 동전들의 번호를 적은 것이다.

예를 들어, 각각 **1 2 3 4 5 6**과 **3 4 6 2 1 5**이 가벼운 것부터의 순서인, 두개의 케스트 케이스를 가진 입력은 다음과 같다.

```
2
1 2 3 4 5 6
```

```
3 4 6 2 1 5
```

Sample grader는 `answer()` 함수에 주어진 배열을 출력한다.