2-3. 포도주 시음

산들이는 포도주(와인)를 좋아하여 집에 N종류의 포도주를 전시해 놓았다. 집에 있는 N종류의 포도주는 각각 $R[0], R[1], \cdots, R[N-1]$번째로 맛있다. 그러나 산들이는 주량이 적어서 포도주의 제대로 된 맛을 즐기기도 전에 취해버린다. 산들이를 안쓰럽게 생각한 당신은 바텐더를 고용하여 산들이가 포도주의 맛을 음미할 수 있게 하려고 한다.

바텐더는 포도주를 블렌딩하여 포도주 고유의 맛은 잃지 않고 알코올 도수만 조절할 수 있다. 당신은 바텐더에게 적절히 일을 시켜서 각 포도주에 $A[0], A[1], \cdots, A[N-1]$의 알코올이 들어가도록 할 수 있다. 바텐더는 이미 포도주들을 맛있는 순서대로 정렬할 수 있지만 산들이가 직접 맛을 느끼면서 포도주를 정렬하기를 바라는 당신의 귀띔에 따라 산들이에게 각 포도주의 알코올 양만 알려주기로 하였다. 기술적인 문제로 인해 포도주의 알코올 양은 양의 정수여야 한다.

산들이는 바텐더가 블렌딩한 포도주들을 가지고 맛있는 순서대로 정렬하려고 한다. 산들이는 하루에 두 종류의 포도주 P, Q를 마시고 더 맛있는 포도주를 체크한다. 그러나 산들이가 하루에 K보다 많은 알코올을 섭취하면 취하여 쓰러진다. 즉, 산들이가 비교할 두 포도주의 알코올 양의 합 $A[P] + A[Q]$가 K보다 크면 두 포도주의 맛을 비교할 수 없다. 산들이가 포도주를 음미할 수 있어야 하므로 모든 포도주의 알코올 양은 K를 넘으면 안 된다.

산들이는 30일 이내에 포도주를 맛있는 순으로 정렬하려고 한다. 산들이가 30일 이내에 포도주를 정렬할 수 있도록 바텐더가 포도주를 블렌딩하고 산들이가 포도주를 시음하는 전략을 짜보자.

요구 사항

당신은 파일 두 개를 제출해야 한다.

첫 번째 파일은 `bartender.cpp`이다. 이 파일에서 당신은 다음 함수를 구현해야 한다. 이 파일은 `bartender.h`와 같이 컴파일된다.

```
int[] BlendWines(int K, int[] R)
```

- K: 산들이의 주량
- R: 길이 N의 배열. 포도주 i $(0 \le i \le N-1)$는 $R[i]$번째로 맛있는 포도주이다.
- 이 함수는 길이 N의 배열 A를 반환해야 한다. $A[i]$ $(0 \le i \le N-1)$는 i번 포도주의 알코올 양을 의미한다. $1 \le A[i] \le K$를 만족해야 한다.
- 이 함수는 한 번 호출된다.

두 번째 파일은 `taster.cpp`이다. 이 파일에서 당신은 다음 함수를 구현해야 한다. 이 파일은 `taster.h`와 같이 컴파일된다.

```
int[] SortWines(int K, int[] A)
```

- K: 산들이의 주량
- A: 길이 N의 배열. $A[i]$ $(0 \le i \le N-1)$는 i번 포도주의 알코올 양을 의미한다. A는 BlendWines 함수의 반환값과 같다.
- 이 함수는 길이 N의 배열 r을 반환해야 한다. 임의의 $0 \le i \le N-1$에 대하여 $r[i]$는 $R[i]$와 같아야 한다.
- 이 함수는 한 번 호출된다.

SortWines 함수에서 아래 함수를 호출할 수 있다.

```
int Compare(int P, int Q)
```

- P, Q: 산들이가 비교할 두 포도주의 번호. $0 \le P, Q \le N-1$, $P \neq Q$, $A[P] + A[Q] \le K$를 만족해야 한다.
- 이 함수는 P번 포도주가 Q번 포도주보다 맛있다면 -1을, 그렇지 않다면 1을 반환한다.
- 이 함수는 최대 30번 호출할 수 있다.

위에 언급된 조건을 만족하지 않거나 SortWines의 반환값이 올바르지 않다면, 당신의 프로그램은 Wrong Answer으로 채점된다. 나머지 경우에는 당신의 프로그램은 Accepted으로 채점된다.

채점 과정

채점은 아래 절차대로 진행한다. 당신의 프로그램이 Wrong Answer임이 확실하다면 프로그램이 중간에 끝날 수 있다.

1. BlendWine 함수가 한 번 호출된다.
2. SortWine 함수가 한 번 호출된다.
3. 프로그램이 문제 없이 동작했다면 당신의 프로그램은 Accepted으로 채점된다.

중요 공지사항

- 실제 채점 시에는 제출한 두 프로그램이 독립적으로 컴파일 및 실행된다.
- 첫 번째 프로그램이 비정상적으로 끝났으면 (Runtime Error, Time/Memory Limit Exceeded), 두 번째 프로그램을 실행하지 않고 바로 Wrong Answer로 채점된다.
- 시간 및 메모리 사용량은 각 프로그램의 사용량 합으로 계산된다.
- 작성한 프로그램에서 표준 입력과 표준 출력을 사용하면 안 되고, 어떠한 파일에도 접근하면 안 된다. 이를 어길 시 당신의 프로그램이 Wrong Answer로 채점될 수 있다.

제한

- $1 \le N \le 30$
- $7 \le K \le 30$
- $1 \le R[i] \le N$ $(0 \le i \le N-1)$
- $R[i] \ne R[j]$ $(0 \le i < j \le N-1)$

부분문제

1. (100점) 추가 제약조건은 없다.

당신의 프로그램이 $K \ge C$를 만족하는 (그런 C가 여러 개 있다면 가장 작은 값 선택) 모든 K에 대하여 `Accepted`으로 채점되었다면, 당신의 점수 P는 C의 값에 따라 결정된다. 점수를 계산하는 규칙은 다음과 같다.

- $7 < C \le 30$이면, $P = \left\lfloor 100 \times 0.87^{C-7} \right\rfloor$. $C = 30$이면, $P = 4$이다.
- $C = 7$이면, $P = 100$.

만약 그런 $C \le 30$가 없다면, 당신의 점수는 0이다.

예제

`bartender.cpp`에서 발생한 다음 호출을 고려해보자.

```
BlendWines(7, [1, 3, 2])
```

`BlendWines` 함수가 $[3, 4, 5]$를 반환했다고 가정하자.

`taster.cpp`에서 다음 호출이 발생한다.

```
SortWines(7, [3, 4, 5])
```

`SortWines`에서 아래와 같이 함수 호출을 할 수 있다.

```
Compare(0, 1)
```

반환값은 -1이다. `Compare(0, 2)` 호출의 경우 산들이가 취하므로 올바르지 않은 호출이다.

당신의 프로그램이 `Accepted`으로 채점되려면, `SortWines`의 반환값은 $[1, 3, 2]$여야 한다.

샘플 인터페이스

문제 페이지에서 샘플 코드를 다운로드받을 수 있다. 만약 Visual Studio나 Eclipse, Code::Blocks와 같은 IDE 툴을 사용한다면 `bartender.cpp`, `bartender.h`, `taster.cpp`, `taster.h`, `grader.cpp`를 한 프로젝트에 넣어서 컴파일하면 된다.

터미널에서 코드를 컴파일한다면 아래 컴파일 명령어를 이용하면 된다.

```
g++-7 -Wall -lm -static -DEVAL -o wine -O2 grader.cpp bartender.cpp t
aster.cpp -std=c++17
```

답안을 제출할 때에는 `bartender.cpp`와 `taster.cpp`를 제출하면 된다.

Input format

- line 1: N K
- line 2: $R[0]$ $R[1]$ $\cdots$ $R[N-1]$

Output format

당신의 프로그램이 `Accepted`으로 채점되었다면, 샘플 그레이더는 첫 번째 줄에 `Correct`를 출력한다.

당신의 프로그램이 `Wrong Answer`으로 채점되었다면, 샘플 그레이더는 첫 번째 줄에 에러 메시지를 출력한다.