2020 SCPC Round 1 Problems

Scrapped by git@j-min5u

1.

당신은 내일부터 KK일 동안 다이어트를 할 계획이다. 한 식당 AA에서 점심을, 다른 식당 BB에서 저녁을 먹을 것이다.  
다이어트 중이라 아침은 건너뛰기로 한다. AA 식당에는 NN 가지의 메뉴가 있고, BB 식당에도 NN 가지의 메뉴가 있다.  
당신은 KK일 동안 AA와 BB 식당에서 모두 서로 다른 메뉴를 먹을 것이다.  
즉, AA 식당에서 서로 다른 KK개의 메뉴를 고르고 BB 식당에서도 서로 다른 KK개의 메뉴를 고른 후 이들 메뉴를 적절한 순서로 먹을 것이다.   
당신이 원하는 것은 KK일 중에서 최대 칼로리를 섭취한 날의 칼로리 양을 최소로 하는 것이다.  
AA 식당 메뉴 각각의 칼로리 값과 BB 식당 메뉴 각각의 칼로리 값을 입력으로 받아 최대 칼로리를 섭취한 날의 칼로리 양을 최소화하는 프로그램을 작성하라.  
  
- 제한시간: 전체 테스트 케이스는 100개 이하이며, 전체 수행 시간은 2초 이내. (Java 4초 이내)   
    제한 시간을 초과하면 제출한 소스코드의 프로그램이 즉시 종료되며,  
    그때까지 수행한 결과에서 테스트 케이스를 1개 그룹 이상 통과하였더라도 점수는 0점이 됩니다.  
    그러나, 제한 시간을 초과하더라도 테스트 케이스를 1개 그룹 이상 통과하였다면 '부분 점수(0< 점수< 만점)'를 받을 수 있으며,  
    이를 위해서는, C / C++ 에서 "printf 함수" 사용할 경우, 프로그램 시작부분에서 "setbuf(stdout, NULL);"를 한번만 사용하십시오.  
    C++에서는 "setbuf(stdout, NULL);"와 "printf 함수" 대신 "cout"를 사용하고, Java에서는 "System.out.printIn"을 사용하시면,  
    제한 시간을 초과하더라도 '부분 점수'를 받을 수 있습니다.                                     ※ 언어별 기본 제공 소스코드 내용 참고  
    만약, 제한 시간을 초과하지 않았는데도 '부분 점수'를 받았다면, 일부 테스트 케이스를 통과하지 못한 경우 입니다.  
  
- 메모리 사용 제한 : heap, global, static 총계 256MB, stack 100MB  
  
- 제출 제한 : 최대 10회 (제출 횟수를 반영하여 순위 결정 → 동점자의 경우 제출 횟수가 적은 사람에게 높은 순위 부여)

**메모리 사용 제한**

heap, global, static (총계) : 256MB  
stack : 100MB

**입력**

입력 파일에는 여러 테스트 케이스가 포함될 수 있다.  
파일의 첫째 줄에 테스트 케이스의 개수를 나타내는 자연수 TT 가 주어지고,  
이후 차례로  TT 개의 테스트 케이스가 주어진다. (1≤T≤100)(1≤T≤100)  
각 테스트 케이스의 두 식당의 메뉴 수를 나타내는 정수 NN과 날짜 수를 나타내는 KK가 주어진다 (1≤N≤200,000, 1≤K≤N)(1≤N≤200,000, 1≤K≤N).   
둘째 줄에는 AA 식당 메뉴들의 칼로리 값이 양의 정수로 주어진다.  
셋째 줄에는 BB 식당 메뉴들의 칼로리 값이 양의 정수로 주어진다.   
모든 칼로리 값은 109109이하이다.  
  
- 점수 : 각 제출에서 취득한 점수 중에서 최대 점수 (만점 100점)  
   주어지는 테스트 케이스 데이터들의 그룹은 아래와 같으며,  
 각 그룹의 테스트 케이스를 모두 맞추었을 때 해당되는 부분 점수를 받을 수 있다.  
  
ㆍ 그룹 1 (   7점) : N≤10N≤10.  
ㆍ 그룹 2 (  29점) : N≤2,000N≤2,000.  
ㆍ 그룹 3 (  64점) : 이 그룹의 테스트 케이스에서는 원래의 조건 외에는 다른 제약조건이 없다.

**출력**

각 테스트 케이스의 답을 순서대로 표준출력으로 출력하여야 하며,  
각 테스트 케이스마다 첫 줄에는 “Case #CC”를 출력하여야 한다. 이때 CC는 테스트 케이스의 번호이다.  
그 다음 줄에, 하루 칼로리 섭취량 최대의 가능한 최소값을 정수로 출력한다.

**입출력예**

| 입력 |
| --- |
| 2  2 2  1 2  4 2  3 2  6 3 1  1 4 3 |
| 출력 |
| Case #1  5  Case #2  4 |
|  |

2.

두 사람 AA와 BB가 숫자가 적힌 카드들을 가지고 게임을 할 것이다. AA와 BB사이에는 각각 nn장이 쌓여 있는 카드 더미가 2개 있다.  
AA부터 게임을 진행할 것이고 둘이 번갈아 가며 한 턴 씩 진행한다.  
AA와 BB는 자신의 턴이 되면 두 더미 중 하나를 골라서 위에서부터 한 장 이상의 카드를 가져가야 한다.  
단, 한 턴에 가져가는 카드들에 적힌 숫자의 합이 kk 이하라야 한다. 카드들에 적힌 숫자는 모두 kk 이하임이 보장된다.   
  
마지막으로 카드를 가지고 가는 사람이 지는 것이 게임의 규칙이다.  
두 더미를 XX와 YY라고 부르고 두 더미에 주어진 카드들에 적힌 숫자를 아래쪽 카드부터 기록한 것이 각각 X1,X2,...,XnX1,X2,...,Xn와 Y1,Y2,...,YnY1,Y2,...,Yn이라고 하자.  
우리가 관심이 있는 것은 AA와 BB가 받을 수 있는 모든 가능한 카드 더미의 상태들을 구별해 보는 것이다.  
구별한다의 의미를 설명하기 위해 우선 상태가 무엇인지부터 명확히 하도록 하자.  
카드 더미의 한 상태란 XX에 ii장이 있고 YY에 jj장이 있는 (0≤i,j≤n)(0≤i,j≤n) 모든 경우들을 각각 말한다.  
즉 모든 가능한 상태는 (n+1)2(n+1)2가지이다. 참고로, XX에 ii장이 있고 YY에 j장이 있는 상태에서 XX에 있는 카드들에 적힌 숫자는  
아래쪽 카드부터 X1,X2,...,Xi, YX1,X2,...,Xi, Y에 있는 카드들에 적힌 숫자는 아래쪽 카드부터 Y1,Y2,...,YjY1,Y2,...,Yj일 것이다.  
이 상태를 {(X1,X2,...,Xi),(Y1,Y2,...,Yj)}{(X1,X2,...,Xi),(Y1,Y2,...,Yj)}로 표기하기로 한다.  
  
내 턴이 되었는데 XX에 ii장이 있고 YY에 jj장이 있는 상태라면,   
    1. 내가 최선의 작전을 편다면 상대방이 어떤 작전을 쓰더라도 반드시 이길 수 있는 상황,   
    2. 내가 지금부터 어떤 작전을 쓰더라도 상대방이 최선의 작전을 쓰면 내가 절대로 이길 수 없는 상황,  
두가지의 상황을 생각할 수 있다.  
우리는 위의 두가지 종류의 상황에만 관심이 있다.  
특별히 i=0, j=0i=0, j=0인 상태는 이미 이긴 것이므로 반드시 이기는 상황으로 간주한다.  
  
예를 들어, n=2, k=3n=2, k=3 이라고 하자. 두 더미의 카드에 있는 숫자들이 처음에 {(3,3),(3,3)}{(3,3),(3,3)}인 경우를 생각해 보자.  
k=3k=3이므로 AA든 BB든 매번 단 한장의 카드만 가져가는 것이 가능하다.  
따라서, 상태 {(3,3),(3,3)}{(3,3),(3,3)}를 받으면 반드시 이기고, 상태 {(3,3),(3)}{(3,3),(3)}나 {(3),(3,3)}{(3),(3,3)}를 받으면 반드시 지고,  
상태 {(3,3),()},{(),(3,3)}{(3,3),()},{(),(3,3)} 혹은 {(3),(3)}{(3),(3)}를 받으면 반드시 이기며,  
상태 {(),(3)}{(),(3)}혹은 {(3),()}{(3),()}를 받으면 반드시 진다는 것을 알 수 있다.  
  
비슷한 예로, n=2,k=3n=2,k=3이라고 하자.  
두 더미의 카드에 있는 숫자들이 처음에 {(1,2),(1,2)}{(1,2),(1,2)}인 경우를 생각해 보자.  
AA는 한 더미에서 한 장을 가져가거나 두 장을 가져갈 수 있다.  
BB가 받은 상태가 {(1,2),(1)}{(1,2),(1)}라면 BB는 두 장을 가져가서 {(),(1)}{(),(1)}인 상태를 AA에게 주어 반드시 이길 수 있다.  
또, BB가 받은 상태가 {(1,2),()}{(1,2),()}라면 BB는 한 장을 가져가서 {(1),()}{(1),()}인 상태를 AA에게 주어 반드시 이길 수 있다.  
즉, AA가 받은 {(1,2),(1,2)}{(1,2),(1,2)}인 상태는 BB가 최선을 다한다면 AA가 어떤 작전을 쓰더라도 이길 수 없는 상황이다.   
두 카드 더미에 있는 카드들에 적힌 숫자들을 입력으로 받아서, 만들어질 수 있는 모든 가능한 상태들에 대해서 위의 1번과 2번으로 구분되는 상태들의 개수를 계산하는 프로그램을 작성하시오.  
  
- 제한시간: 전체 테스트 케이스는 50개 이하이며, 전체 수행 시간은 3초 이내. (Java 6초 이내)   
    제한 시간을 초과하면 제출한 소스코드의 프로그램이 즉시 종료되며,  
    그때까지 수행한 결과에서 테스트 케이스를 1개 그룹 이상 통과하였더라도 점수는 0점이 됩니다.  
    그러나, 제한 시간을 초과하더라도 테스트 케이스를 1개 그룹 이상 통과하였다면 '부분 점수(0< 점수< 만점)'를 받을 수 있으며,  
    이를 위해서는, C / C++ 에서 "printf 함수" 사용할 경우, 프로그램 시작부분에서 "setbuf(stdout, NULL);"를 한번만 사용하십시오.  
    C++에서는 "setbuf(stdout, NULL);"와 "printf 함수" 대신 "cout"를 사용하고, Java에서는 "System.out.printIn"을 사용하시면,  
    제한 시간을 초과하더라도 '부분 점수'를 받을 수 있습니다.                                     ※ 언어별 기본 제공 소스코드 내용 참고  
    만약, 제한 시간을 초과하지 않았는데도 '부분 점수'를 받았다면, 일부 테스트 케이스를 통과하지 못한 경우 입니다.  
  
- 메모리 사용 제한 : heap, global, static 총계 256MB, stack 100MB  
  
- 제출 제한 : 최대 10회 (제출 횟수를 반영하여 순위 결정 → 동점자의 경우 제출 횟수가 적은 사람에게 높은 순위 부여)

**메모리 사용 제한**

heap, global, static (총계) : 256MB  
stack : 100MB

**입력**

입력 파일에는 여러 테스트 케이스가 포함될 수 있다.  
파일의 첫째 줄에 테스트 케이스의 개수를 나타내는 자연수 TT가 주어지고,  
이후 차례로  TT개의 테스트 케이스가 주어진다. (1≤T≤50)(1≤T≤50)  
각 테스트 케이스의 첫 줄에는 한 더미의 카드 수를 나타내는 정수 nn과 한꺼번에 가져갈 수 있는 카드들의 숫자의 합 제한 kk가 주어진다 (1≤n,k≤3,000)(1≤n,k≤3,000).   
둘째 줄에는 XX더미에 있는 카드들에 적힌 숫자를 아래에서부터 기록한 값들이 양의 정수로 주어진다.  
셋째 줄에는 YY더미에 있는 카드들에 적힌 숫자를 아래에서부터 기록한 값들이 양의 정수로 주어진다.  
카드에 적힌 숫자들은 kk이하임이 보장된다.  
   
- 점수 : 각 제출에서 취득한 점수 중에서 최대 점수 (만점 150점)  
   주어지는 테스트 케이스 데이터들의 그룹은 아래와 같으며,  
 각 그룹의 테스트 케이스를 모두 맞추었을 때 해당되는 부분 점수를 받을 수 있다.  
  
ㆍ 그룹 1 ( 53점) : 1≤n,k≤2001≤n,k≤200  
ㆍ 그룹 2 ( 97점) : 이 그룹의 테스트 케이스에서는 원래의 조건 외에는 다른 제약조건이 없다.

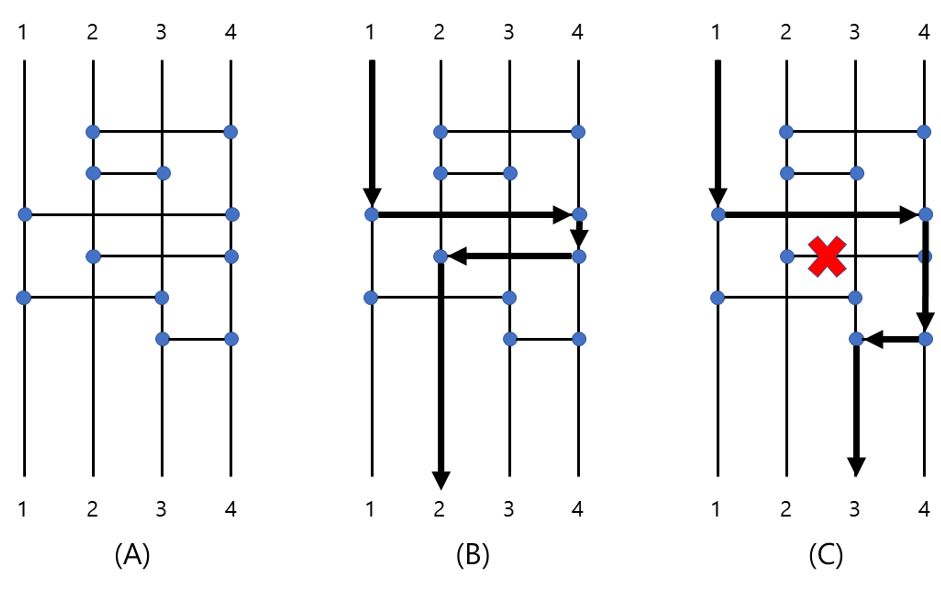
**출력**

각 테스트 케이스의 답을 순서대로 표준출력으로 출력하여야 하며,  
각 테스트 케이스마다 첫 줄에는 “Case #CC”를 출력하여야 한다. 이때 CC는 테스트 케이스의 번호이다.  
1번으로 구분되는 상태의 개수와, 2번으로 구분되는 상태의 개수를 정수로 출력한다.

**입출력예**

| 입력 |
| --- |
| 2  2 3  3 3  3 3  2 3  1 2  1 2 |
| 출력 |
| Case #1  5 4  Case #2  6 3 |

3.

아래 그림 (A)에서 보인 것 같은 사다리가 주어질 때,  
모든 가로방향의 이음선이 정상적인 게임에선 그림 (B)에서 보인 것처럼 상단 1번을 출발한 선은 하단 2번에 도착하게 된다.  
즉, 1→2로 연결된다.  
그러나, 그림 (C)에서 보인 것처럼 일부 가로이음선이 고장(X로 표시된 것)이라면 그것을 통해 이동할 수 없어 1→3으로 연결된다.  
  
  
  
사다리에는 NN개의 세로선이 있는데, 그림에서 보인 것처럼 각 세로선은 11부터 NN사이의 정수로 구분된다.  
kk개의 가로이음선 각각은 양끝점을 연결하는 세로선의 번호 쌍 (i,j)(i,j)  (1≤i<j≤N)(1≤i<j≤N)으로 표현된다.  
가로이음선의 높이(위치)는 모두 다르며, 위에서부터 아래로 순서대로 그 정보가 주어진다.   
  
출발지점을 나타내는 정수 i(1≤i≤N)i(1≤i≤N)와 도착지점을 나타내는 정수 j(1≤j≤N)j(1≤j≤N)가 주어질 때, ii 에서 jj 로 가는 경로가 형성되기 위해 가로이음선 중 최소 몇 개가 고장인지를 밝히는 프로그램을 작성하시오.  
  
- 제한시간: 전체 테스트 케이스는 70개 이하이며, 전체 수행 시간은 3초 이내. (Java 6초 이내)   
    제한 시간을 초과하면 제출한 소스코드의 프로그램이 즉시 종료되며,  
    그때까지 수행한 결과에서 테스트 케이스를 1개 그룹 이상 통과하였더라도 점수는 0점이 됩니다.  
    그러나, 제한 시간을 초과하더라도 테스트 케이스를 1개 그룹 이상 통과하였다면 '부분 점수(0< 점수< 만점)'를 받을 수 있으며,  
    이를 위해서는, C / C++ 에서 "printf 함수" 사용할 경우, 프로그램 시작부분에서 "setbuf(stdout, NULL);"를 한번만 사용하십시오.  
    C++에서는 "setbuf(stdout, NULL);"와 "printf 함수" 대신 "cout"를 사용하고, Java에서는 "System.out.printIn"을 사용하시면,  
    제한 시간을 초과하더라도 '부분 점수'를 받을 수 있습니다.                                     ※ 언어별 기본 제공 소스코드 내용 참고  
    만약, 제한 시간을 초과하지 않았는데도 '부분 점수'를 받았다면, 일부 테스트 케이스를 통과하지 못한 경우 입니다.  
  
- 메모리 사용 제한 : heap, global, static 총계 256MB, stack 100MB  
  
- 제출 제한 : 최대 10회 (제출 횟수를 반영하여 순위 결정 → 동점자의 경우 제출 횟수가 적은 사람에게 높은 순위 부여)

**메모리 사용 제한**

heap, global, static (총계) : 256MB  
stack : 100MB

**입력**

입력 파일에는 여러 테스트 케이스가 포함될 수 있다.  
파일의 첫째 줄에 테스트 케이스의 개수를 나타내는 자연수 TT 가 주어지고,  
이후 차례로  TT 개의 테스트 케이스가 주어진다. (1≤T≤70)(1≤T≤70)  
각 테스트 케이스의 첫 줄에는 세 정수 N,N, k,k, mm 이 주어진다.  
여기서 NN (2≤N≤1500)(2≤N≤1500)은 사다리의 세로선의 개수, kk (1≤k≤2000)(1≤k≤2000)는 가로이음선의 개수,  
mm (1≤m≤105)(1≤m≤105)은 출발점과 도착점 쌍의 개수를 나타낸다.  
다음 줄부터 kk줄에 걸쳐서 가로이음선의 양끝점을 연결하는 세로선의 번호 쌍 (i,j)(i,j) (1≤i<j≤N)(1≤i<j≤N)이 kk 개 차례로 주어진다.  
가로이음선의 높이(위치)는 모두 다르며, 위에서부터 아래로 순서대로 정보가 주어진다.  
다음 줄부터 mm개의 줄에 걸쳐서 출발점과 도착점의 위치를 나타내는 정수 쌍 (i,j)(i,j) (1≤i,j≤N)(1≤i,j≤N)이 한 줄에 한 쌍씩 주어진다.  
  
- 점수 : 각 제출에서 취득한 점수 중에서 최대 점수 (만점 150점)  
   주어지는 테스트 케이스 데이터들의 그룹은 아래와 같으며,  
 각 그룹의 테스트 케이스를 모두 맞추었을 때 해당되는 부분 점수를 받을 수 있다.  
  
ㆍ 그룹 1 (26 점) : 이 그룹의 테스트 케이스에서는 N≤20,N≤20, k≤15k≤15 이다.  
ㆍ 그룹 2 (53 점) : 이 그룹의 테스트 케이스에서는 N≤300,N≤300, k≤300k≤300 이다.  
ㆍ 그룹 3 (71 점) : 이 그룹의 테스트 케이스에서는 원래의 조건 외에는 다른 제약조건이 없다.

**출력**

각 테스트 케이스의 답을 순서대로 표준출력으로 출력하여야 하며,  
각 테스트 케이스마다 첫 줄에는 “Case #CC”를 출력하여야 한다. 이때 CC는 테스트 케이스의 번호이다.  
다음 줄엔 입력에서 주어진 (출발점, 도착점) 쌍 각각에 대해, 주어진 출발점에서 주어진 도착점으로 경로가 형성되기 위해 필요한 고장난 가로이음선의 최소 개수의 합을 출력한다.  
만약 출발점에서 도착점으로 연결되는 방법이 없을 경우 -1을 더하면 된다.

**입출력예**

| 입력 |
| --- |
| 3  3 4 4  1 2  2 3  1 2  1 3  1 1  1 2  2 3  3 2  4 6 5  1 2  2 3  3 4  1 3  2 4  1 4  1 3  1 4  3 3  3 4  4 4  4 2 4  1 2  2 3  1 1  1 2  1 3  1 4 |
| 출력 |
| Case #1  3  Case #2  6  Case #3  1 |

4.

길이가 nn인 숫자로 된 문자열 tt가 있다. 정수 kk가 주어졌을 때, tt에서 연속한 kk개의 숫자로 이루어진 숫자들을 모두 만들어보자.    
예를 들어, t=3141592t=3141592이고 k=3k=3이라면, 수 314, 141, 415, 159, 592314, 141, 415, 159, 592를 만들 수 있다.   
동일한 숫자가 두 번 이상 나온다면, 모두 따로따로 취급한다.   
어떤 수가 0으로 시작한다면 해당하는 자릿수는 없다. 예를 들면, 03120312는 312312이다.   
이제 정수 mm이 주어졌을 때, 길이 mm인 구간 [a, a+m][a, a+m]의 시작 위치 aa를 조절하여 이 수들을 가장 많이 포함하는 구간을 찾고, 이 구간에 포함된 수의 개수를 세려고 한다.      
예를 들어,  m=200m=200이라고 하면, 구간 [140, 340][140, 340]에 141, 159, 314141, 159, 314가 들어갈 수 있다.    
구간의 시작 위치에 따라 포함되는 수가 달라질 수 있지만, 최대 3개의 수를 포함할 수 있다는 것은 달라지지 않는다.   
  
여기까지만 풀면 너무 쉬우니까, tt의 정확하게 한 위치 ii를 골라서 여기에 놓인 숫자를 1로 바꿀 수 있다.  
바꾼 문자열에 대해서 역시 연속한 kk개의 숫자로 이루어진 숫자들을 모두 만들어보고, 이 수들을 가장 많이 포함하는 길이 mm인 구간을 찾고 이 구간에 포함된 수의 개수를 세자.  
총 nn곳의 위치를 고를 수 있고, 원래 문자열까지 고려하면 우리는 n+1n+1가지 경우와 각각의 답을 알고 있다.  
이 중 최대값을 구하고, 이를 구간 안의 수의 최대값이라고 부르자.   
  
n, t, k, mn, t, k, m 이 주어질 때, 구간 안의 수의 최대값을 구해서 출력하는 프로그램을 작성하시오.  
  
- 제한시간: 전체 테스트 케이스는 40개 이하이며, 전체 수행 시간은 10초 이내. (Java 15초 이내)   
    제한 시간을 초과하면 제출한 소스코드의 프로그램이 즉시 종료되며,  
    그때까지 수행한 결과에서 테스트 케이스를 1개 그룹 이상 통과하였더라도 점수는 0점이 됩니다.  
    그러나, 제한 시간을 초과하더라도 테스트 케이스를 1개 그룹 이상 통과하였다면 '부분 점수(0< 점수< 만점)'를 받을 수 있으며,  
    이를 위해서는, C / C++ 에서 "printf 함수" 사용할 경우, 프로그램 시작부분에서 "setbuf(stdout, NULL);"를 한번만 사용하십시오.  
    C++에서는 "setbuf(stdout, NULL);"와 "printf 함수" 대신 "cout"를 사용하고, Java에서는 "System.out.printIn"을 사용하시면,  
    제한 시간을 초과하더라도 '부분 점수'를 받을 수 있습니다.                                     ※ 언어별 기본 제공 소스코드 내용 참고  
    만약, 제한 시간을 초과하지 않았는데도 '부분 점수'를 받았다면, 일부 테스트 케이스를 통과하지 못한 경우 입니다.  
  
- 메모리 사용 제한 : heap, global, static 총계 256MB, stack 100MB  
  
- 제출 제한 : 최대 10회 (제출 횟수를 반영하여 순위 결정 → 동점자의 경우 제출 횟수가 적은 사람에게 높은 순위 부여)

**메모리 사용 제한**

heap, global, static (총계) : 256MB  
stack : 100MB

**입력**

입력 파일에는 여러 테스트 케이스가 포함될 수 있다.  
파일의 첫째 줄에 테스트 케이스의 개수를 나타내는 자연수 TT 가 주어지고,  
이후 차례로  TT 개의 테스트 케이스가 주어진다. (1≤T≤40)(1≤T≤40)  
각 테스트 케이스의 첫 줄에는 문자열의 길이를 나타내는 정수 n (1≤n≤50,000),n (1≤n≤50,000), 부분문자열의 길이를 나타내는 정수 k (1≤k≤min(9,n)),k (1≤k≤min⁡(9,n)), 구간의 길이를 나타내는 정수 m (0≤m≤1,000,000,000)m (0≤m≤1,000,000,000)이 주어진다.   
둘째 줄에는 0~9 숫자로 이루어진 길이 nn 인 문자열 tt가 주어진다.   
   
- 점수 : 각 제출에서 취득한 점수 중에서 최대 점수 (만점 200점)  
   주어지는 테스트 케이스 데이터들의 그룹은 아래와 같으며,  
 각 그룹의 테스트 케이스를 모두 맞추었을 때 해당되는 부분 점수를 받을 수 있다.  
  
ㆍ 그룹 1 ( 39점) : 이 그룹의 테스트 케이스에서는 n≤1,000n≤1,000이다.  
ㆍ 그룹 2 ( 161점) : 이 그룹의 테스트 케이스에서는 원래의 조건 외에는 다른 제약조건이 없다.

**출력**

각 테스트 케이스의 답을 순서대로 표준출력으로 출력하여야 하며,  
각 테스트 케이스마다 첫 줄에는 “Case #CC”를 출력하여야 한다. 이때 CC는 테스트 케이스의 번호이다.  
그 다음 줄에, 구간 안의 수의 최대값을 출력한다.

**입출력예**

| 입력 |
| --- |
| 1  10 1 0  2211122112 |
| 출력 |
| Case #1  6 |

5.

유명 관광지로 여행을 떠나기 전에, 지도를 보며 관광 계획을 세우고 있다.   
지도(2차원 평면)에서 nn개의 위치 a1,a2,...,ana1,a2,...,an이 주어지고, 각 위치 aiai에는 사건(또는 사고)이 일어날 확률 pipi가 주어진다.   
모든 위치들의 좌표는 서로 다르다.  
지도에서 우범지역이 정의되고, 우리가 방문할 위치 qq가 주어질 때, 우리는 qq가 우범지역에 포함될 확률을 알고 싶다.  
단, 임의의 서로 다른 두 위치 aiai와 ajaj에 대해서, 위치 qq와 ​​ai, aj​​ai, aj는 일직선 상에 놓여있지 않다.  
우범지역이란 지도에서 실제로 사건이 일어난 위치들(3개이상의 위치들)의 convex hull로 정의한다.   
Convex hull은 주어진 위치들을 모두 포함하는 가장 작은 convex 집합이다.   
여기서, convex 집합 CC란, CC에 속한 임의의 두 점에 대해서, 두 점을 연결하는 선분이 모두 CC에 포함되는 경우를 말한다.  
nn개 위치들의 좌표와 그 위치에서 사건이 일어날 확률이 주어지고, 방문할 위치 qq가 주어질 때, qq가 우범지역에 포함될   
확률을 계산해서 출력하는 프로그램을 작성하시오.  
  
- 제한시간: 전체 테스트 케이스는 70개 이하이며, 전체 수행 시간은 2초 이내. (Java 4초 이내)   
    제한 시간을 초과하면 제출한 소스코드의 프로그램이 즉시 종료되며,  
    그때까지 수행한 결과에서 테스트 케이스를 1개 그룹 이상 통과하였더라도 점수는 0점이 됩니다.  
    그러나, 제한 시간을 초과하더라도 테스트 케이스를 1개 그룹 이상 통과하였다면 '부분 점수(0< 점수< 만점)'를 받을 수 있으며,  
    이를 위해서는, C / C++ 에서 "printf 함수" 사용할 경우, 프로그램 시작부분에서 "setbuf(stdout, NULL);"를 한번만 사용하십시오.  
    C++에서는 "setbuf(stdout, NULL);"와 "printf 함수" 대신 "cout"를 사용하고, Java에서는 "System.out.printIn"을 사용하시면,  
    제한 시간을 초과하더라도 '부분 점수'를 받을 수 있습니다.                                     ※ 언어별 기본 제공 소스코드 내용 참고  
    만약, 제한 시간을 초과하지 않았는데도 '부분 점수'를 받았다면, 일부 테스트 케이스를 통과하지 못한 경우 입니다.  
  
- 메모리 사용 제한 : heap, global, static 총계 256MB, stack 100MB  
  
- 제출 제한 : 최대 10회 (제출 횟수를 반영하여 순위 결정 → 동점자의 경우 제출 횟수가 적은 사람에게 높은 순위 부여)

**메모리 사용 제한**

heap, global, static (총계) : 256MB  
stack : 100MB

**입력**

입력 파일에는 여러 테스트 케이스가 포함될 수 있다.  
파일의 첫째 줄에 테스트 케이스의 개수를 나타내는 자연수 TT 가 주어지고,  
이후 차례로  TT 개의 테스트 케이스가 주어진다 (1≤T≤70)(1≤T≤70).  
각 테스트 케이스의 첫 줄에는 지도에서 위치들의 개수를 나타내는 정수 nn이 주어진다 (3≤n≤100,000)(3≤n≤100,000).   
둘째 줄에는 nn개의 정수 x1,x2,...,xnx1,x2,...,xn이 주어지고, xixi는 ii번째 위치 aiai의 xx 좌표이다 (1≤xi≤109)(1≤xi≤109).   
셋째 줄에는 nn개의 정수 y1,y2,...,yny1,y2,...,yn이 주어지고, yiyi는 ii번째 위치 aiai의 yy 좌표이다 (1≤yi≤109)(1≤yi≤109).   
넷째 줄에는 nn개의 실수 p1,p2,...,pnp1,p2,...,pn이 주어지고, pipi는 ii번째 위치 aiai에서 사건이 일어날 확률이다 (0<pi≤1)(0<pi≤1).   
여기서, 각 확률 pipi는 소수점 2자리 실수이다.  
다섯째 줄에는 방문할 위치 qq의 xx 좌표와 yy 좌표를 나타내는 두 정수 aa와 bb가 주어진다 (1≤a, b≤109)(1≤a, b≤109).  
  
- 점수 : 각 제출에서 취득한 점수 중에서 최대 점수 (만점 200점)  
   주어지는 테스트 케이스 데이터들의 그룹은 아래와 같으며,  
 각 그룹의 테스트 케이스를 모두 맞추었을 때 해당되는 부분 점수를 받을 수 있다.  
  
ㆍ 그룹 1 (81 점) : 이 그룹의 테스트 케이스에서는 n≤5,000이다.  
ㆍ 그룹 2 (119 점) : 이 그룹의 테스트 케이스에서는 원래의 조건 외에는 다른 제약조건이 없다.

**출력**

각 테스트 케이스의 답을 순서대로 표준출력으로 출력하여야 하며,  
각 테스트 케이스마다 첫 줄에는 “Case #CC”를 출력하여야 한다. 이때 CC는 테스트 케이스의 번호이다.  
그 다음 줄에, qq가 우범지역에 포함될 확률을 출력한다. 출력 값은 정답과 절대오차가 10−410−4이하를 허용합니다.

**입출력예**

| 입력 |
| --- |
| 2  3  1 2 3  1 3 1  0.01 0.02 0.01  2 2  3  1 2 3  1 3 1  0.01 0.02 0.01  1 2 |
| 출력 |
| Case #1  0.000002  Case #2  0.000000 |