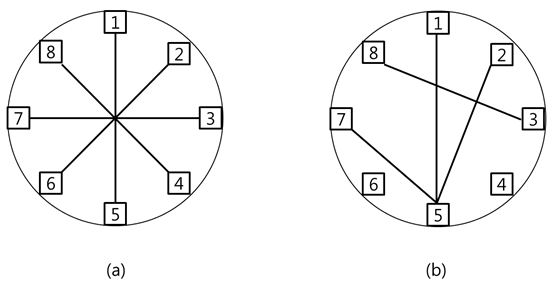
|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 문제: **(중) [사전A-0033] [2019년 12월 PRO 검정 사전 테스트] 스마트 팩토리**  **※ 본 테스트는 Professional 사전테스트로     문제풀이 과정에서 본인의 힘으로 문제를 풀이하지 않고 타인의 조언을 구하거나               본인이 직접 작성하지 않은 자료를 그대로, 혹은 위조, 변조, 참조하여 제출하는 경우 부정행위로     판단될 수 있으며, 이러한 경우 6개월간 모든 SW검정 응시가 제한됨을 알려드립니다.     (이미 Professional 이상을 취득한 자격자 또한 본 규정은 동일하게 적용됩니다.)   ※ 본 문제는 2019년 11월 14일(목) 13시까지(서버시간 기준)만 풀이가능하며,     위 기한 내에 문제의 정답을 제출해야 2019년 12월 Professional SW검정을 신청할 수 있습니다.   ※ 마지막 제출 답안 결과가 최종결과로 반영되니, 답안제출에 유의하여 주시기 바랍니다.   ※ 11월 14일(목) 13시 이후에도 문제는 계속 오픈되며, 일반 연습문제와 같이 계속 풀이할 수 있습니다.       11월 14일(목) 13시 이전에는 정답자이더라도 타 정답자 코드 조회가 불가능하며     이후에는 정답자에 한해 타 정답자 코드 조회가 가능합니다.**   |  |  | | --- | --- | | 시간 제한 | **40** 개의 Test Input 입력시 C/C++ **1** 초, Java **1** 초 | | 메모리 제한 | Stack : **1** Mbytes  /  Total : **256** Mbytes | | 코드길이 제한 | **128** Kbytes | | 채점 | 답안을 제출하면 Test Input에 대한 결과를 판정해서 실시간으로 알려준다.  모든 케이스에 대해 정답일 경우 **Pass**, 그 외의 경우 **Fail** 이다. |   A Company는 전자부품을 납품하는 회사이며 중국에 공장을 가지고 있다.  이 공장에서는 26 가지의 부품을 생산할 수 있으며 각 부품은 기호 A ~ Z 로 나타낸다.  이 공장의 생산라인은 다음과 같이 구성되어 있다.  각 라인에서는 하루에 N 개의 부품을 순차적으로 처리 가능하며, 공장 전체에 M 개의 라인이 있고 모든 라인은 동시에 작동을 시작하고, 동시에 작동을 마감한다.  각 라인의 처리속도는 같으며 모든 부품을 만드는 시간은 동일하다.   그러나 이 공장에는 치명적인 결함이 있어서 엔지니어의 개입이 없을 경우 만들어진 모든 부품이 불량품이 되고 만다.  이러한 상황을 임시적으로 조치하기 위해서 A Company에서는 K 명의 엔지니어를 파견했다.  각 엔지니어들은 한 종류의 부품만을 책임지고 담당하며, 투입된 시점부터 퇴근 전까지 모든 라인에서 생산되는 담당 부품의 불량률을 0%로 만들 수 있다.  엔지니어들은 **한 번 투입되면 공장이 끝날 때까지 일할 수 있지만, 한 번 일한 엔지니어는 그날 다시 일하지 않는다.**  또한 **두 명 이상의 엔지니어가 동시에 일할 수는 없다.** 즉 다른 엔지니어를 투입하고 싶으면 이전에 있던 엔지니어를 퇴근시키고 교체해야 하며 퇴근한 엔지니어는 다음날 출근한다.  이 공장의 일일 생산계획 및 근무 가능한 엔지니어 리스트를 받았을 때 **엔지니어를 효과적으로 투입하여 생산할 수 있는 정상부품의 최대개수**를 구하라.  C:\Users\yongnam.kwon\Pictures\스마트팩토리1.png [그림 1]  [그림 1] 과 같은 일일 생산계획이 주어지고 근무 가능한 엔지니어가 3 명이고 각각 담당하는 부품이 F, G, A인 경우를 생각해보자.  C:\Users\yongnam.kwon\Pictures\스마트팩토리2.png [그림 2]  A를 담당하는 엔지니어와 F를 담당하는 두 명의 엔지니어를 [그림 2]와 같이 투입하여 작업을 진행한다면 총 8개의 정상부품을 생산할 수 있으며, 이 경우 생산할 수 있는 정상부품의 수가 최대 값이 된다.    **[제한사항]**  1. N 은 1 이상 5000 이하의 자연수, M 은 1 이상 100 이하의 자연수이다.  2. K 는 1 이상 5 이하의 자연수이다.  3. K 명의 엔지니어를 모두 투입 시킬 필요는 없다.  4. K 명의 엔지니어 중 서로 다른 두 담당자의 담당 부품이 동일 할 수도 있다.    **[입력]**  첫 줄에 테스트 케이스의 개수 T 가 주어지며, 그 다음 T 개의 테스트 케이스가 주어진다.  각 테스트 케이스는 여러 줄로 구성되어 있으며, 첫 줄에는 각 라인에서 하루에 처리하는 부품의 수 N, 전체 라인의 수 M, 투입되는 엔지니어의 수 K 가 공백으로 구분되어 주어진다.  그 다음 줄에는 각 엔지니어가 담당하는 부품 알파벳이 K 개 주어진다.  그 다음부터 M 개의 줄에 걸쳐 각 생산 라인의 일일 생산계획이 주어진다. 각 생산 라인의 일일 생산계획은 N 개의 알파벳으로 주어진다.    **[출력]**  각 테스트 케이스에 대해 #x (x는 테스트 케이스 번호, 1부터 시작)을 출력하고 공백을 하나 둔 다음, 최대 생산 가능한 정상부품의 수를 출력한다.    **[입출력 예]**  (입력)                     4      // 테스트케이스 개수  5 5 1  // 1번째 케이스  A  AAEFM  BDEKN  ADFJN  ACFIM  ABGHA  5 5 1  // 2번째 케이스  F  AAEFM  BDEKN  ADFJN  ACFIM  ABGHA  5 5 3  // 3번째 케이스  GFA  AAEFM  BDEKN  ADFJN  ACFIM  ABGHA  5 5 4  // 4번째 케이스  BAZA  AACBA  ABCHU  ACFIA  ABKPA  AAZMA   (출력)                                           #1 6  #2 3  #3 8  #4 13   (sample\_input.txt 의 출력)  #1 6  #2 3  #3 8  #4 13  #5 1  #6 0  #7 10  #8 7  #9 7  #10 3  #11 3  #12 7  #13 8  #14 9  #15 15  #16 27  #17 33  #18 30  #19 68  #20 169  문제: **(중상) [기출P-0018] 종이컵 전화기** | | |
| 시간 제한 | | **25**개의 Test Input 입력시 C/C++ **1**초 / Java **1.5**초 |
| 메모리 제한 | | Stack : **1** Mbytes  /  Total : **256** Mbytes |
| 제출횟수 제한 | | **제한없음** |
| 채점 | | 답안을 제출하면 제공된 Sample input 에 대한 결과를 판정해서 실시간으로 알려주며  그 의미는 다음과 같다.  **Submit**: Sample input에 대하여 1개 이상 정답  **Fail** : Sample input에 대하여 오답 또는 런타임 에러, Time out 등  (※ sample input 파일 다운로드는 문제 지문 바로 아래에 있습니다.)  **최종결과는 별도의 채점용 Test input으로 진행되므로 시험 중 판정 결과와 다를 수 있다.** |
| 평가기준 | | 정답 처리된 Test Input의 개수, 실행 속도, 메모리 사용량, 구현 코드 |

수진이네 동네는 원 모양이며, 원 위에 N개의 집이 동일한 간격으로 위치해 있다. 집들은 한 집에서 시작하여 시계방향으로 1번부터 N번까지 번호가 붙어 있다. 각 집에는 종이컵이 하나씩 있으며, 서로 다른 2개의 종이컵을 연결하는 실들이 존재한다. 즉, 하나의 실 양 끝에 연결된 종이컵 끼리 통신이 가능하며, 이런 방법으로 집들간에 대화를 나눈다.

아래 그림의 (a)와 (b)는 4 개의 실이 4 쌍의 집을 연결하고 있는 경우의 예시이다. 실은 언제나 팽팽하게 연결되어 있으므로 항상 직선으로 표현됨에 주의하라. 그림 (a)의 경우 4개의 실이 한 점에서 교차하고 있다. 그림 (b)도 4 개의 실로 집을 연결하는 예시이지만 교차하는 방식이 다르다.



한 집에는 단 하나의 종이컵만 있어서 최대 다른 한 집하고만 대화를 할 수 있다. 또, 두 개 이상의 실이 교차하는 경우에는 그들 중 단 하나만을 사용할 수 있다. 이러한 상황이 주어졌을 때 최대로 동시에 대화가 가능한 집의 쌍의 수를 알고자 한다. 위의 (a)의 경우는 단 한 쌍만 대화가 가능하며 ( (1,5), (2,6), (3,7), (4,8) 중 한 쌍 ), (b)의 경우는 두 쌍만 대화가 가능하다 ( (3,8) 그리고 (5,7) )

집들의 개수와 실들이 연결된 쌍들을 입력으로 받아 대화가 동시에 가능한 집 **쌍**의 최대 수를 출력하는 프로그램을 작성하라.

**[입력]**첫 줄에 케이스의 개수 T가 주어지고, 이후 차례로 T개의 테스트 케이스가 주어진다. 각 케이스의 첫번째 줄에는 집의 개수 N 과 실의 개수 M 이 차례로 주어진다. (**2 ≤ N ≤ 200, 1 ≤ M ≤ 500**, N, M 은 정수) 이후 M개의 줄에는 두 개의 서로 다른 집 번호가 주어진다. 한 쌍의 집을 연결하는 실의 개수는 최대 한 개이다.

**[출력]**

각 케이스마다 줄의 시작에 “#x”를 출력하고 (x는 케이스의 번호이며 1부터 시작함.) 공백을 하나 둔 뒤 동시에 대화가 가능한 집의 쌍의 최대 수를 출력한다.

**[입출력 예]**(입력)  
2  
8 4  
1 5  
6 2  
4 8  
7 3  
8 4  
1 5  
2 5  
7 5  
3 8

(출력)  
#1 1  
#2 2

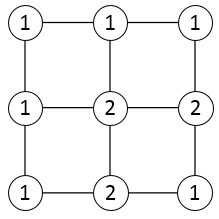
(sample\_input.txt에 대한 출력)

#1 4  
#2 3  
#3 4  
#4 4  
#5 3  
#6 15  
#7 15  
#8 14  
#9 15  
#10 16

문제: **(중상) [기출P-0046] 직사각형 체인**

|  |  |
| --- | --- |
| 시간 제한 | **30** 개의 Test Input 입력시 C/C++ **2.5**초, Java **3**초 |
| 메모리 제한 | Stack : **5** Mbytes  /  Total : **256** Mbytes |
| 코드길이 제한 | **128** Kbytes |
| 채점 | 답안을 제출하면 Test Input에 대한 결과를 판정해서 실시간으로 알려준다.  모든 케이스에 대해 정답일 경우 **Pass**, 그 외의 경우 **Fail** 이다. |

2차원 정사각형 모양의 격자가 있다. 격자의 크기는 N x N 이다 (즉, 격자의 선분이 교차하는 부분을 교점이라 하면 한 줄에 교점이 N개씩 있다는 뜻이다). 아래 **[그림 1]** 은 3 x 3 격자의 예시이다. 문제의 해설을 위해 편의상 격자의 교점을 원으로 그렸다.

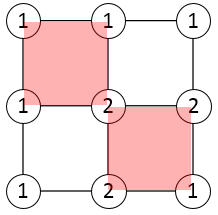


**[그림 1]**

이 격자에 제일 왼쪽 위의 점에서 시작해서 제일 오른쪽 아래 점에서 끝나는 직사각형 체인을 만들려고 한다. 직사각형 체인은 격자 내에 있는 직사각형들을 순서대로 배치한 것을 말하는데, 체인이 만들어지는 규칙은 다음과 같다.

1. 직사각형 체인을 구성하는 직사각형은 격자의 교점을 꼭짓점으로 가지며 변이 수평이나 수직인 것들이다.   
2. 첫번째 직사각형은 격자의 제일 왼쪽 위 교점을 자신의 왼쪽 위 꼭짓점으로 가져야 한다.   
3. 마지막 직사각형은 격자의 제일 오른쪽 아래 교점을 자신의 오른쪽 아래 꼭짓점으로 가져야 한다.   
4. 직사각형 체인의 한 직사각형 A 다음에 직사각형 B가 온다면, A의 오른쪽 아래 꼭짓점과 B의 왼쪽 위 꼭짓점이 같아야 한다.

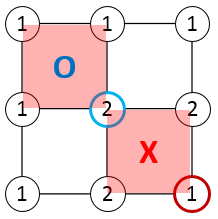
아래 **[그림 2]** 는 위의 [그림 1]에 직사각형 체인을 이루는 두개의 직사각형을 덮어씌워 그린 예이다.



**[그림 2]**

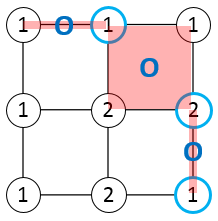
**유의할 사항은, 본 문제에서는 가로 길이가 0 이거나 세로 길이가 0 인 것도 특별히 직사각형으로 허용하지만, 가로, 세로 길이가 모두 0 인 것은 직사각형으로 허용하지 않는다는 것이다.**

교점에 적힌 수는 다음을 의미한다. 한 교점에 적힌 수가 K 라면, 그 교점을 **오른쪽 아래 꼭짓점**으로 두는 직사각형은 **가로 길이과 세로 길이의 합이 정확히 K 이어야 한다**는 것이다. 위 그림을 보면 직사각형 체인이기는 하지만, 제일 오른쪽 아래에 적힌 K=1인 상황과 맞지 않아 이 문제에서 허용되는 체인은 아님을 알 수 있다 (**[그림 3]** 참고).



**[그림 3]**

아래의 **[그림 4]** 는 옳은 예시 중 하나이다.



**[그림 4]**

격자의 정보를 입력 받아 모든 가능한 직사각형 체인의 개수를 출력하는 프로그램을 작성하라.

**[입력]**   
입력 파일의 제일 첫째 줄에는 파일에 포함된 케이스의 수 T 가 주어진다. 케이스의 첫 줄에 격자의 크기 N 이 주어진다 **(2 ≤ N ≤ 1000)**. 다음 N 개의 줄에 격자 교점에 적힌 숫자들이 정확히 위의 그림과 같은 형태로 주어진다. 격자 교점에 적힌 수는 **1 이상 2N-2 이하의 정수**이다.

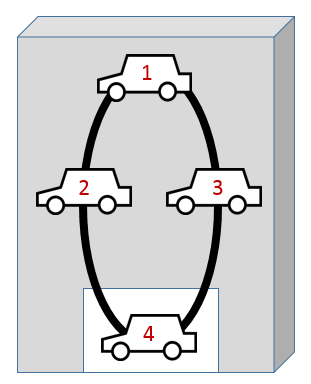
**[출력]**   
각 테스트 케이스의 답을 순서대로 표준출력으로 출력하며, 각 케이스마다 줄의 시작에 “#x”를 출력하여야 한다. 이때 x는 케이스의 번호이다. 같은 줄에 가능한 직사각형 체인의 개수를 1,000,000,007로 나눈 나머지를 출력한다.

**[입출력 예]**   
(입력)   
2                                              ← 2 test cases in total   
3                                              ← 1st case   
1 1 1   
1 1 1   
1 1 1   
3                                              ← 2nd case   
1 1 1   
1 2 2   
1 2 1

(출력)   
#1 6   
#2 4   
  
#1 2   
#2 0   
#3 3   
#4 0   
#5 2899389   
#6 1   
#7 1   
#8 379014945   
#9 47579842   
#10 600073979

|  |  |  |  |
| --- | --- | --- | --- |
|  | | |  | | --- | | 문제: **(중상) [기출P-0035] 주차 타워** | |
| 시간 제한 | | **20** 개의 Test Input 입력시 C/C++ **1**초, Java **1**초 |
| 메모리 제한 | | Stack : **1** Mbytes  /  Total : **256** Mbytes |
| 코드길이 제한 | | **128** Kbytes |
| 채점 | | 답안을 제출하면 Test Input에 대한 결과를 판정해서 실시간으로 알려준다.  모든 케이스에 대해 정답일 경우 **Pass**, 그 외의 경우 **Fail** 이다. |

주차 타워에 차가 N 대 주차되어 있다. 주차 타워는 1층에서 차를 넣거나 꺼낼 수 있는 구조이다. 주차 타워의 내부는 하나의 원형 체인으로 생각할 수 있다. 차들은 1부터 N까지 번호가 붙어 있다. 아래 그림은 N=4 일 때의 예시 중 하나를 도식화한 것이다.



위 그림에서 주차 타워에는 4대의 차를 넣을 수 있는 공간이 있으며 1번부터 4번까지의 차가 주차되어 있다. 현재 바로 꺼낼 수 있는 차는 4번 차이다. 만일, 1번 차를 꺼내려 한다면 체인을 왼쪽이나 오른쪽으로 2칸 회전하면 꺼낼 수 있다. 체인을 왼쪽이나 오른쪽으로 회전하는 칸 수를 비용으로 생각하자.

여기 있는 차들을 정확히 번호 순서대로 꺼내야 한다고 하면 (즉, k번 차는 k번째로 꺼내야 함), 위의 예의 경우 최소 비용은 6이 됨을 알 수 있다.

하지만 꺼내는 번호 순서에 최대 1의 오차를 허용한다고 하면 (예를 들어 3번 차는 2번째 ~ 4번째 사이에 꺼내면 됨) 위의 예에서 4의 비용만 들일 수 있다.

주차 타워의 상태를 입력으로 받아 번호 순서에 최대 1의 오차를 허용하는 조건으로 차를 꺼내는 최소 비용을 계산하는 프로그램을 작성하라.

**[입력]**입력 파일의 제일 첫째 줄에는 파일에 포함된 케이스의 수 T가 주어진다. 케이스의 첫 줄에 주차할 수 있는 칸의 개수 N이 주어진다. (1≤N≤1,000,000) 다음 줄에 N개의 차 번호의 순서가 주어진다. 그 순서대로 원형 체인에 위치하고 있다는 의미이다. 순서에서 첫 번째로 주어진 번호의 차가 꺼낼 수 있는 위치(1층)에 존재한다.

**[출력]**각 테스트 케이스의 답을 순서대로 표준출력으로 출력하며, 각 케이스마다 줄의 시작에 “#x”를 출력하여야 한다. 이때 x는 케이스의 번호이며 1부터 시작한다. 같은 줄에 공백을 하나 두고 가능한 최소비용을 출력한다.

**[입출력 예]**(입력)  
3         <- 3 test cases in total  
4         <- 1st case  
4 2 1 3  
4         <- 2nd case  
3 4 1 2  
5         <- 3rd case  
3 5 4 1 2

(출력)  
#1 4  
#2 4  
#3 6

※ [입출력 예]의 1번째 케이스가 문제 내용에서 언급한 예시와 동일함.

(sample\_input.txt에 대한 정답)  
#1 4  
#2 49  
#3 96  
#4 216  
#5 1767  
#6 54342  
#7 180884  
#8 18111073  
#9 72349290  
#10 1816156699

문제: **(중상) [기출A-0014] 타잔과 치타**

|  |  |
| --- | --- |
| 시간 제한 | **40** 개의 Test Input 입력시 C/C++ **2**초 / Java **2.5**초 |
| 메모리 제한 | Stack : **1** Mbytes  /  Total : **256** Mbytes |
| 제출횟수 제한 | **10** 회 |
| 채점 | 답안을 제출하면 Test Input에 대한 결과를 판정해서 실시간으로 알려주며  그 의미는 다음과 같다.  **Pass : Test Input에 대하여 모두 정답 처리된 경우**  **Fail : Test Input에 대하여 일부 혹은 모두 오답 처리된 경우**  ※ Test Input : 검정 시스템에서 제출된 코드 실행 時 자동으로 입력되는 Input Data |
| 평가기준 | 정답 처리된 Test Input의 개수 |

**[문제]**

일직선 상에 나무들이 N개 서 있고 인접한 나무들 사이의 간격은 1미터이다. 나무들은 왼쪽부터 오른쪽으로 1번부터 번호가 붙어 있다. 나무들의 높이는 충분이 높은 것으로 가정한다. 나무들의 동일한 높이에 서로 길이가 다른 로프들이 나무마다 하나씩 달려 있다. 타잔은 제일 왼쪽 나무에서 시작해서 로프를 타고 다른 나무로 이동한다. 한 나무에 달려 있는 로프 길이가 L이라면, 타잔은 현재 나무에서 거리가 L이하인 어떤 나무로든 이동할 수 있다.

타잔은 치타의 도움을 받을 수 있는데, 치타는 어떤 나무로든 바로 이동할 수 있고, 그 나무의 로프를 타고 타잔이 위치한 나무로 (타잔과 동일한 방식으로) 이동하여 타잔이 그 로프를 잡도록 할 수 있다. 타잔은 그 로프를 받아서 이동하는 방법으로 더 멀리 이동할 수 있다. 타잔이 로프를 타는 횟수는 K번 이하이며, 그 중 치타의 도움을 받을 수 있는 횟수는 C번 이하이다. (치타의 도움을 받은 경우도 K번의 횟수에 포함된다.)

간단한 예로 아래의 수열이 나무들의 로프 길이를 순서대로 표현한 것이라고 하자.

**2, 1, 1, 3, 1, 1, 2**

K=1, C=0인 경우라면 타잔은 3번 나무까지만 이동이 가능하다. 하지만, K=1, C=1인 경우라면 4번 나무에서 치타가 로프를 타고 와서 전달한 로프를 받아 7번 나무까지 한번에 갈 수가 있다.  
N, L들, K, C의 값을 입력 받아 타잔이 오른쪽으로 가장 멀리 갈 수 있는 나무의 번호를 계산하는 프로그램을 작성하라.

**[입력]**  
입력 파일에는 여러 테스트 케이스가 포함될 수 있다. 파일의 첫째 줄에 케이스의 개수 T가 주어지고, 이후 차례로 T개 테스트 케이스가 주어진다.

각 테스트 케이스의 첫 줄에 자연수 세 개가 주어지는데, 첫 수는 나무의 수를 나타내는 N(1≤N≤5,000), 타잔이 로프를 탈 수 있는 횟수인 K (1≤K≤N), 셋째 수는 치타가 도움을 줄 수 있는 횟수 C(0≤C≤2, C≤K)이다. 이후 N개의 줄에 왼쪽부터 각 나무에 걸린 로프의 길이를 나타내는 자연수가 하나씩 주어진다. 로프의 길이는 최대 N이다.

**[출력]**  
각 테스트의 답을 순서대로 표준출력으로 출력하며, 케이스 마다 줄의 시작에 “#x”를 출력하여야 한다. 이때 x는 케이스의 번호이다. 바로 한 칸을 떼고 타잔이 가장 멀리 갈 수 있는 나무의 번호를 출력한다.

**[입출력 예]**  
입력   
2                          ← 총 2개의 테스트 케이스가 있다는 뜻    
7 1 1                      ← N, K, C의 값, Case 1의 시작 (예제의 경우)  
2                          ← 로프의 길이 7개의 시작  
1  
1  
3  
1  
1  
2  
5 2 0                        ← Case 2의 시작  
2  
3  
1  
1  
2

출력  
#1 7  
#2 5

(sample\_input.txt에 대한 정답)  
#1 7  
#2 12  
#3 1234  
#4 1664  
#5 875  
#6 1318  
#7 1889  
#8 1009  
#9 143  
#10 1542