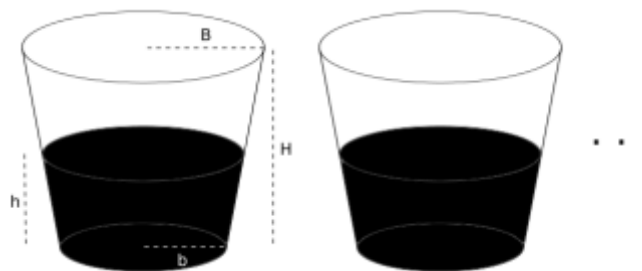


## A. Dividindo a Coca

Time limit: 1s

Um grupo de  $N$  amigos foi a um restaurante almoçar. Embora cada pessoa escolheu um prato diferente para comer, todos decidiram que iriam beber Coca-Cola. O grupo pediu então uma garrafa de  $L$  ml de Coca-Cola, e deve agora dividir o conteúdo da garrafa entre as  $N$  pessoas de tal forma que todas as pessoas recebam partes iguais da bebida.

Há um copo vazio para cada pessoa, que deve ser preenchido com o refrigerante que aquela pessoa irá beber. Todos os  $N$  copos são iguais, e podem ser descritos como um tronco de um cone cuja altura é  $H$  cm, cuja base menor é uma circunferência de raio  $b$  cm e cuja base maior é uma circunferência de raio  $B$  cm, como indicado na imagem.

Indicação de copos preenchidos com Coca-Cola até a altura  $h$  cm

Sua tarefa é ajudar o grupo a dividir a garrafa de Coca-Cola. Para tal, você deve encontrar a altura  $h$  de cada copo de tal forma que, se cada copo for preenchido com refrigerante até a altura  $h$  cm, então todas as pessoas irão receber a mesma quantia da bebida. Considere que nenhum copo será preenchido com uma quantia de refrigerante maior que sua capacidade.

**Entrada**

A primeira linha de entrada contém um inteiro  $C$  que determina a quantidade de casos de teste. Cada caso de teste inicia com uma linha contendo dois inteiros,  $N$  e  $L$  ( $1 \leq N \leq 100$ ,  $1 \leq L \leq 10^8$ ), indicando o número de pessoas no grupo e a quantia de Coca-Cola que deve ser dividida, em mililitros. A segunda linha contém três inteiros  $b$ ,  $B$  e  $H$  ( $1 \leq b \leq B \leq 100$ ,  $1 \leq H \leq 100$ ), indicando o raio da base menor e maior do copo, assim como sua altura. Todas as medidas são dadas em centímetros.

**Saída**

Para cada caso de teste, imprima o valor de  $h$  tal que cada copo deve ser preenchido até a altura  $h$  cm. Arredonde e imprima o resultado com exatamente 2 casas decimais.

Exemplo de Entrada	Exemplo de Saída
2	2.40
1 200	6.19
5 6 8	
2 350	
3 3 16	



## B. Area of Two Circles' Intersection

Time limit: 2s

Memory limit: 256 MB

You are given two circles. Find the area of their intersection.

**Input**

The first line contains three integers  $x_1, y_1, r_1$  ( $-10^9 \leq x_1, y_1 \leq 10^9, 1 \leq r_1 \leq 10^9$ ) — the position of the center and the radius of the first circle.

The second line contains three integers  $x_2, y_2, r_2$  ( $-10^9 \leq x_2, y_2 \leq 10^9, 1 \leq r_2 \leq 10^9$ ) — the position of the center and the radius of the second circle.

**Output**

Print the area of the intersection of the circles. The answer will be considered correct if the absolute or relative error doesn't exceed  $10^{-6}$ .

**Examples****input**

```
0 0 4
6 0 4
```

**output**

```
7.25298806364175601379
```

**input**

```
0 0 5
11 0 5
```

**output**

```
0.00000000000000000000
```

## C. Palavras

Time limit: 1s

Dados dois conjuntos de palavras formadas por zeros e uns, você deve escrever um programa para determinar se existem concatenações de palavras de cada um dos conjuntos que gerem uma mesma palavra. Por exemplo, se um conjunto A contém as palavras 010 e 11 e outro conjunto B contém as palavras 0 e 101, então a palavra 01011010 pode ser formada tanto por concatenações de palavras de A como por concatenações de palavras de B:

$$010 \cdot 11 \cdot 010 = 01011010 = 0 \cdot 101 \cdot 101 \cdot 0$$

**Entrada**

A primeira linha de um caso de teste contém dois inteiros,  $N_1$  ( $1 \leq N_1$ ), e  $N_2$  ( $N_2 \leq 20$ ), que indicam respectivamente o número de palavras do primeiro e do segundo conjunto de palavras. Cada uma das  $N_1$  linhas seguintes contém uma palavra do primeiro conjunto. Cada uma das  $N_2$  linhas seguintes contém uma palavra do segundo conjunto.


Obs: cada palavra tem no mínimo um caractere e no máximo 40 caracteres, todos zeros e uns.

**Saída**

Para cada caso de teste seu programa deve imprimir uma única linha, contendo um único caractere. Se for possível encontrar uma concatenação de uma ou mais palavras do primeiro conjunto que seja igual a uma concatenação de uma ou mais palavras do segundo conjunto então o caractere deve ser S, caso contrário deve ser N.

Exemplo de Entrada	Exemplo de Saída
2 2	S
010	N
11	S
0	
101	
3 1	
1	
00	
000	
01	
1 1	
00	
000	

Maratona de Programação da SBC 2012

Maratona de Programação da SBC  Brasil

## D. Organizador de Vagões

Time limit: 1s

Na estação de trem você ainda pode encontrar o último dos “organizadores de vagões”. Um Organizador de vagões um empregado cujo trabalho é apenas reordenar os vagões do trem, trocando-os de posição. Uma vez que os vagões são organizados em uma ordem considerada ótima, o condutor pode desconectar cada vagão e colocá-los na estação.

O título “organizador de vagões” é dado à pessoa que realiza esta tarefa, cuja estação fica perto de uma ponte. Ao invés da ponte poder subir ou descer, ela roda sobre um pilar que fica no centro do rio. Após rodar 90 graus, os barcos podem passar na esquerda ou direita dela. O Primeiro organizador de vagões descobriu que girando a ponte 180 graus com dois vagões em cima dela, é possível a troca de lugar entre os dois vagões. Obviamente a ponte pode operar no máximo com dois vagões sobre ela.

Agora que quase todos os organizadores de vagões já faleceram, a estação gostaria de automatizar esta operação. Parte do programa a ser desenvolvido é uma rotina que decide para um dado trem com um determinado número de vagões, o número de trocas entre trens adjacentes que são necessárias para que o trem fique ordenado. Sua tarefa é criar tal rotina.

## Entrada

A entrada contém na primeira linha o número de caso de testes (**N**). Cada caso de teste consiste de duas linhas de entrada. A primeira linha de um caso de teste contém um inteiro **L**, determinando o tamanho do trem ( $0 \leq L \leq 50$ ). A segunda linha de um caso de teste contém uma permutação dos números 1 até **L**, indicando a ordem corrente dos vagões. Os vagões devem ser ordenados de forma que o vagão 1 venha por primeiro, depois o 2, etc, com o vagão **L** vindo por último.

## Saída

Para cada caso de teste imprima a sentença: 'Optimal train swapping takes **S** swaps.' onde **S** é um inteiro.

Exemplo de Entrada	Exemplo de Saída
3	Optimal train swapping takes 1
3	swaps.
1 3 2	Optimal train swapping takes 6
4	swaps.
4 3 2 1	Optimal train swapping takes 1
2	swaps.
2 1	

Tradução, entrada e saída por Neilor.

Autor Desconhecido

## E. FACE 2015 FREE GIFT

Time limit: 1s

A FACE em 2015 está apoiando a terceira edição da Maratona de Programação, mas desta vez a organização solicitou sua ajuda para criar um sistema de sorteio utilizando as letras da palavra FACE. Como a feira utiliza uma proposta diferenciada e alegre, cada participante que entra na feira ganha 4 letras, uma de cada cor e em formato de bloco de madeira, conforme Figura 1, e deve inserí-las num painel. Se, no momento da inserção, as 4 letras formarem o contrário das 4 últimas letras, o visitante ganhará um brinde.



Figura 1 - Entrada de FACE no painel seguido de ACEF.

Por exemplo: suponha que já tiveram 3 participantes que entraram na feira e o painel ficou da seguinte forma: F A C E E C F A A C F E A C E F. Note que sempre que o painel fica vazio, assim como no início do evento, as letras F A C E são inseridas pela organização do evento. Agora, na entrada do quarto participante, ele inseriu as letras F E C A e, com isso, receberá um brinde por fechar o contrário de A C E F. Após essa situação, o painel deve ficar F A C E E C F A A C F E.

Escreva um algoritmo que, dadas as letras recebidas e inseridas pelos participantes, diga quantos participantes ganharam brindes. Lembre-se que sempre que o painel fica vazio as letras F A C E são inseridas pela organização do evento.

## Entrada

A primeira linha de cada caso de teste contém um inteiro  $N$  ( $1 \leq N \leq 100$ ), representando o número de visitantes que vão receber as letras. Em cada uma das  $N$  linhas seguintes deve ser informada a combinação das 4 letras que o visitante deseja inserir no painel, separadas por espaço.


## Saída

Para cada grupo de visitantes, deve ser informado quantos destes receberão brindes.

Exemplos de Entrada	Exemplos de Saída
4 E C F A A C E F F E C A A F C E	2
3 E A C F A F C E E F C A	0

6	6
E C A F	
E C A F	
E C A F	
E C A F	
E C A F	
E C A F	

III Maratona de Programação FACE - 2015

Por Tiago Zonta, Unoesc Campus Chapecó  Brazil

## F. Airport

Time limit: 1s

A crescente utilização do transporte aéreo preocupa os especialistas, que prevêem que o congestionamento em aeroportos poderá se tornar um grande problema no futuro. Os números atuais já são alarmantes: relatórios oficiais demonstram que na Europa, em junho de 2001, houve uma média de 7.000 atrasos de vôos por dia. Preocupada com a previsão dos seus especialistas em tráfego aéreo, a Associação de Transporte Aéreo Internacional (ATAI) está começando um estudo para descobrir quais são os aeroportos onde o tráfego aéreo pode vir a ser mais problemático no futuro.

Como programador recém contratado pela ATAI você foi encarregado de escrever um programa para determinar, a partir de uma listagem de aeroportos e vôos, qual aeroporto possui maior probabilidade de congestionamento no futuro. Como medida da probabilidade de congestionamento será utilizado neste estudo o número total de vôos que chegam ou que partem de cada aeroporto.

**Entrada**

A entrada é composta de vários conjuntos de teste. A primeira linha de um conjunto de teste contém dois números inteiros **A** ( $0 \leq A \leq 100$ ) e **V** ( $0 \leq V \leq 10000$ ), que indicam respectivamente o número de aeroportos e o número de vôos. Os aeroportos são identificados por inteiros de 1 a A. As **V** linhas seguintes contêm cada uma a informação de um vôo, representada por um par de números inteiros positivos **X** e **Y** ( $1 \leq X \neq Y \leq A$ ), indicando que há um vôo do aeroporto **X** para o aeroporto **Y**. O final da entrada é indicado quando **A** = **V** = 0.

**Saída**

Para cada conjunto de teste da entrada seu programa deve produzir três linhas. A primeira linha identifica o conjunto de teste, no formato "Teste n", onde n é numerado a partir de 1. A segunda linha deve conter o identificador do aeroporto que possui maior tráfego aéreo. Caso mais de um aeroporto possua este valor máximo, você deve listar todos estes aeroportos, em ordem crescente de identificação, e separados por pelo menos um espaço em branco. A terceira linha deve ser deixada em branco. A grafia mostrada no Exemplo de Saída, abaixo, deve ser seguida rigorosamente.

---

**Exemplo de Entrada**

---


**Exemplo de Saída**

---



Exemplo de Entrada	Exemplo de Saída
5 7 1 3 2 1 3 2 3 4 4 5 3 5 2 5 3 5 1 3 1 2 3 2 1 2 2 1 0 0	Teste 1 3  Teste 2 1 2

OBI - Olimpíada Brasileira de Informática 2002

Por OBI - Olimpíada Brasileira de Informática 2002  Brazil

## G. Playing with Sets

Time limit: 1s

Dabriel é um menino fissurado por matemática, ele acaba de aprender em sua escola operações sobre conjuntos.

Após passar a tarde toda brincando com alguns conjuntos que ele possui, chega a hora de resolver as lições de casa, porém ele já está muito cansado e com medo de que possa cometer alguns erros, solicitou sua ajuda.

Dabriel deseja um programa de computador que dado **N** conjuntos e os elementos de cada conjunto, ele possa realizar algumas operações, são elas:

**1 X Y:** Retorna a quantidade de elementos distintos da intersecção entre o conjunto **X** com o **Y**.

**2 X Y:** Retorna a quantidade de elementos distintos da união entre o conjunto **X** com o **Y**.

## Entrada


A entrada é composta por diversas instâncias. A primeira linha da entrada contém um inteiro **T** indicando o número de instâncias. Cada instância inicia com um inteiro **N** ( $1 \leq N \leq 10^4$ ), representando a quantidade de conjuntos que Dabriel possui. As próximas **N** linhas começam com um inteiro **M<sub>i</sub>** ( $1 \leq M_i \leq 60$ ), que indica o total de elementos que o conjunto **i** possui, segue então **M<sub>i</sub>** inteiros **X<sub>ij</sub>** ( $1 \leq X_{ij} \leq 60$ ), que representam o valor de cada elemento. Na próxima linha contém um inteiro **Q** ( $1 \leq Q \leq 10^6$ ), representando quantas operações Dabriel deseja realizar. Nas próximas **Q** linhas terá a descrição de uma operação.

## Saída

Para cada operação seu programa deverá imprimir a quantidade de elementos, conforme explicado na descrição.

Exemplo de Entrada	Exemplo de Saída
1	1
4	1
1 1	4
2 1 5	7
3 2 4 6	2
4 1 3 5 7	
5	
1 1 2	
1 1 4	
2 1 4	
2 3 4	
1 2 4	

II Maratona de programação do IFCE-Aracati

Por Gabriel Duarte, UNIFESO  Brazil

## H. Lucky Numbers

Time limit: 0.5s

Memory limit: 64 MB

The numbers of all offices in the new building of the Tax Office of IT City will have lucky numbers.

Lucky number is a number that consists of digits 7 and 8 only. Find the maximum number of offices in the new building of the Tax Office given that a door-plate can hold a number not longer than  $n$  digits.

**Input**

The only line of input contains one integer  $n$  ( $1 \leq n \leq 55$ ) — the maximum length of a number that a door-plate can hold.

**Output**

Output one integer — the maximum number of offices, than can have unique lucky numbers not longer than  $n$  digits.

**Examples**

<b>input</b>
2
<b>output</b>
6