Maratona de Programação de 2009

Universidade de São Paulo

Caderno de Problemas

Departamento de Ciência da Computação IME-USP

Domingo, 16 de agosto de 2009.

Problema A: Cinema de Xing Tzen Zu

Arquivo: `cinema.[c|cpp|java]`

Harbin tem um dos maiores cinemas do mundo. O Cinema "Xing Tzen Zu" é muito largo, tendo poucas filas com muitas cadeiras. O governo chinês tem regras específicas para as pessoas irem ao cinema: apenas casais podem ir ao cinema; e cada casal deve se sentar sempre na mesma fileira e em cadeiras adjacentes (a primeira fileira é ocupada por fazendeiros, motoristas, mecânicos, a segunda por professores, comerciantes, bombeiros, e assim por dias). Mas, ao mesmo tempo, é proibido que todas as pessoas sentem exatamente na mesma posição em duas noites, ou seja, uma mesma configuração não pode ocorrer duas vezes. Isso preocupou o prefeito da cidade, que procurou então descobrir quantas noites o cinema poderia abrir sem que fosse necessário repetir uma configuração que já tinha acontecido anteriormente. Uma restrição importante é que os casais devem sempre ocupar poltronas adjacentes na fileira.

Sua tarefa neste problema é determinar, dado o número de poltronas N de uma fileira e o número de casais M que podem sentar nesta fileira, determine quantos jeitos diferentes todos casais poderiam ocupar as poltronas de forma que não fiquem separados.

Entrada

A entrada é composta por diversas instâncias. A primeira linha da entrada contém um inteiro T indicando o número de instâncias.

Cada instância é composta por uma linha que contém dois inteiros N e M.

A entrada deve ser lida da entrada padrão.

Restrições

$$2 \leq N \leq 4000$$
$$1 \leq M \leq \frac{N}{2}$$

Saída

Para cada instância imprima uma linha contendo o número de jeitos diferentes que os casais poderiam ocupar as poltronas de forma que não fiquem separados.

A resposta dada deve ser módulo 1300031.

A saída deve ser escrita na saída padrão.

Exemplo de entrada	**Saída para o exemplo de entrada**
2 10 2 20 6	224 574954

Problema B: Festival de Estátuas de Gelo

Arquivo: `festival.[c|cpp|java]`

O Festival de Estátuas de Gelo de Harbin é um dos eventos mais fascinantes do inverno chinês. Todos os anos, artistas de todo o mundo se reúnem na cidade, onde fazem esculturas de gelo gigantescas. A cidade vira uma galeria de arte ao céu aberto, uma vez que as esculturas ficam expostas na rua por semanas, sem derreter. Afinal, a temperatura média no inverno de Harbin (época em que ocorrerá a final mundial do ICPC) é de -20 graus.

O primeiro passo para fazer a escultura é montar um grande bloco de gelo da dimensão pedida pelo artista. Os blocos são recortados das geleiras de Harbin em blocos de altura e profundidade padrão e vários comprimentos diferentes. O artista pode determinar qual o comprimento que ele deseja que tenha o seu bloco de gelo para que a escultura possa começar a ser esculpida.

Os comprimentos disponíveis dos blocos são $\{a_1, a_2, \ldots, a_N\}$ e o comprimento que o artista deseja é M. O bloco de comprimento 1 é muito usado, por este motivo ele sempre aparece na lista de blocos disponíveis. Sua tarefa é determinar o número mínimo de blocos tal que a soma de seus comprimentos seja M.

Entrada

A entrada é composta por diversas instâncias. A primeira linha da entrada contém um inteiro T indicando o número de instâncias.

A primeira linha de cada instância contém dois inteiros N e M representando o número de tipos de blocos e o comprimento desejado pelo artista, respectivamente. A próxima linha contém os inteiros $a_1, a_2, \cdots, a_N$ separados por espaço.

A entrada deve ser lida da entrada padrão.

Restrições

$$
\begin{array}{lllll}
1 & \leq & N & \leq & 25 \\
1 & \leq & M & \leq & 1000000 \\
1 & \leq & a_i & \leq & 100 \qquad \text{para todo } i = 1, 2, \cdots, N
\end{array}
$$

Saída

Para cada instância imprima o número mínimo de blocos necessários para obter um bloco de comprimento M.

A saída deve ser escrita na saída padrão.

Exemplo de entrada	**Saída para o exemplo de entrada**
2 6 100 1 5 10 15 25 50 2 103 1 5	2 23

Problema C: Mestre Xu: Kung Fu Louva-a-Deus

Arquivo: `mestrexu.[c|cpp|java]`

Li Tae Xu sempre trabalhou na montagem de blocos de gelo para artistas que vêm para o Festival de Estátuas de gelo de Harbin. Neste festival o artista determina o comprimento do bloco de gelo que deseja para sua escultura (a profundidade e altura são padrão) e a prefeitura de Harbin contrata especialistas para escolher um número mínimo de blocos que devem ser juntados para dar o comprimento desejado. Lembre que os blocos são recortados da geleira em vários comprimentos diferentes $\{a_1, a_2, \ldots, a_N\}$, e o bloco de comprimento 1 sempre aparece na lista de blocos disponíveis.

Mestre Xu, que também é um mestre de Kung Fu estilo Louva-a-Deus (daqueles que conseguem andar no ar), sempre usou a mesma estratégia, que foi lhe ensinada pelo pai Tae Ming Xu, que por sua vez aprendeu com o avô Ming Ling Xu, que por sua vez ... (vocês já entenderam a idéia). A estratégia da família Xu (uma das mais respeitadas no negócio em Harbin) era ir juntando sempre em cada passo o maior bloco possível que fosse possível juntar ao bloco atual sem ultrapassar as medidas pedidas pelo artista. No ano passado, o filho de mestre Xu, Mi Li Xu voltou da Universidade de Harbin dizendo que a estratégia milenar da família nem sempre usava o menor número possível de blocos de gelo. Depois de dar uma surra no garoto insolente, Mestre Xu resolveu verificar se ele dizia mesmo a verdade.

Sua tarefa neste problema é verificar, dado um conjunto de comprimentos distintos de blocos de gelo, se de fato a estratégia da família Xu encontra um conjunto de tamanho mínimo de blocos para qualquer comprimento pedido ou, tragédia, o filho estava com a razão. O bloco de comprimento 1 é muito usado, por este motivo ele sempre aparece na lista de blocos disponíveis.

Entrada

A entrada é composta por diversas instâncias. A primeira linha da entrada contém um inteiro T indicando o número de instâncias.

A primeira linha de cada instância contém um inteiro N representando o número de tipos de blocos diferentes. A próxima linha contém os inteiros $a_1, a_2, \cdots, a_N$ separados por espaço.

A entrada deve ser lida da entrada padrão.

Restrições

$$
\begin{array}{lllll}
1 & \leq & N & \leq & 400 \\
1 & \leq & a_i & \leq & 1000000 \quad \text{para todo } i = 1, 2, \cdots, N \\
 & & a_i & \neq & a_j \quad \text{para todo } i = 1, \cdots, N,\ j = 1, \cdots, N \text{ e } i \neq j
\end{array}
$$

Saída

Para cada instância imprima `sim` se a estratégia da família Xu encontra um conjunto de tamanho mínimo de blocos para qualquer comprimento pedido. Caso contrário, imprima `nao`.

A saída deve ser escrita na saída padrão.

Exemplo de entrada	**Saída para o exemplo de entrada**

`2`
`6`
`1 5 10 15 25 50`
`5`
`1 10 15 25 50`
	`sim`
	`nao`

Problema D: Harbin é muita grana!

Arquivo: `asfalto.[c|cpp|java]`

Harbin é uma cidade organizada, mas construída de forma bastante econômica. Todas as ruas são de mão dupla, e é possível ir de qualquer ponto da cidade a qualquer outro, sempre passando apenas por ruas asfaltadas, mas não existem dois caminhos asfaltados diferentes ligando quaisquer dois pontos da cidade. O prefeito responsável pelo asfaltamento das ruas diz ter aplicado inclusive um algoritmo de um certo holandês, mas nunca ninguém entendeu o nome do algoritmo para poder verificar se ele de fato dizia a verdade.

Na época do festival das estátuas de gelo de Harbin as estátuas são espalhadas em vários pontos da cidade. Os turistas são convidados a percorrer as ruas asfaltadas da cidade de forma a visitar todas elas. Sempre pensando na economia, o prefeito deseja saber qual é o comprimento total, em quilômetros, dos caminhos que ligam todos os pares de esculturas (cada par deve ser contado uma só vez, ou seja, se você já contou o caminho de a até b, não deve contar o caminho de b até a). Sua tarefa neste problema é, dadas as posições das estátuas e os comprimentos das ruas asfaltadas que as ligam, determinar este comprimento total.

Entrada

A entrada é composta por diversas instâncias. A primeira linha da entrada contém um inteiro T indicando o número de instâncias.

A primeira linha de cada instância contém um inteiro N representando o número de estátuas. As estátuas são enumeradas de 1 a N. Cada uma das $N - 1$ linhas seguintes contém três inteiros a, b e c indicando que a rua asfaltada que liga as estátuas a e b tem comprimento c.

A entrada deve ser lida da entrada padrão.

Restrições

$$1 \leq N \leq 10000$$
$$1 \leq c \leq 50$$

Saída

Para cada instância imprima a soma dos comprimentos dos caminhos que ligam todos os pares (não-ordenados) de esculturas.

A resposta dada deve ser módulo 1300031.

A saída deve ser escrita na saída padrão.

Exemplo de entrada	**Saída para o exemplo de entrada**
2 3 1 2 1 1 3 1 3 1 2 2 2 3 1	4 6

Problema E: Contando em chinês

Arquivo: `censo.[c|cpp|java]`

A China é um dos maiores países do mundo e o mais populoso. Realizar um censo no país é quase uma operação de guerra. O governo envia para cada um dos cantões matrizes imensas que devem ser preenchidas com as características de todos os cidadãos. Cada uma dessas matrizes tem o mesmo tamanho: nas linhas estão as várias etnias (são milhares) e nas colunas as características que se deseja medir (pode chegar a milhões). Sabemos que poucos elementos de cada uma dessas matrizes são de fato preenchidos com valores diferentes de zero.

O trabalho da empresa governamental que faz o censo é, então, receber várias matrizes $N \times N$, cada uma dada através de seus elementos não nulos e calcular a matriz soma dessas várias matrizes.

Entrada

A entrada é composta por diversas instâncias. A primeira linha da entrada contém um inteiro T indicando o número de instâncias.

A primeira linha de cada instância contém dois inteiros, N e L representando respectivamente a dimensão das matrizes e o número total de entradas não nulas. As L linhas seguintes contêm quatro inteiros P_k, l_k, c_k e v_k indicando que a matriz P_k tem valor v_k na posição de linha l_k e coluna c_k.

A entrada deve ser lida da entrada padrão.

Restrições

$$
\begin{array}{lcccl}
1 & \leq & N, l_k, c_k, & \leq & 1000000 \\
1 & \leq & L & \leq & 10000000 \\
1 & \leq & P_k & \leq & 1000 \\
1 & \leq & v_k & \leq & 100
\end{array}
$$

Saída

Para cada instância imprima as entradas não nulas da matriz soma. Para cada entrada não nula da matriz, imprima a linha, coluna e valor correspondente, separados por espaço. A saída não precisa estar ordenada.

Após cada instância imprima uma linha em branco.

A saída deve ser escrita na saída padrão.

Exemplo de entrada	**Saída para o exemplo de entrada**

`3`
`1000 4`
`1 1 1 1`
`2 1 1 2`
`3 1 2 100`
`1 1 2 1`
`2 2`
`1000 2 2 1`
`500 2 2 1`
`50 4`
`1 50 1 1`
`2 48 1 2`
`3 50 1 100`
`1 49 2 1`
	`1 1 3`
	`1 2 101`
	
	`2 2 2`
	
	`48 1 2`
	`49 2 1`
	`50 1 101`

Problema F: Estiagem

Arquivo: `estiagem.[c|cpp|java]`

Devido às constantes estiagens que aconteceram nos últimos tempos em algumas regiões do Brasil, o governo federal criou um órgão para a avaliação do consumo de água destas regiões com finalidade de verificar o comportamento da população na época de racionamento. Este órgão responsável irá selecionar algumas cidades (por amostragem) e verificará como está sendo o consumo de cada uma das pessoas da cidade e o consumo médio por habitante de cada cidade.

Entrada

A entrada contém dados de várias cidades. A primeira linha da descrição dos dados de cada cidade contém um inteiro N, indicando a quantidade de casas na cidade. As N linhas seguintes contém um par de valores (X e Y) indicando a quantidade de moradores de cada casa e o seu respectivo consumo total (em m^3). Com certeza, nenhuma residência consome mais do que 200 m^3 por mês. O final da entrada é representado por uma linha contendo apenas o número zero.

A entrada deve ser lida da entrada padrão.

Restrições

$$1 \leq N \leq 1000$$
$$1 \leq X \leq 10$$
$$1 \leq Y \leq 200$$

Saída

Para cada descrição de cidade, deve-se apresentar a mensagem `Cidade# n:`, onde n é o número da cidade seguindo a sequência $(1, 2, \ldots)$. Em seguida deve-se listar, por ordem crescente de consumo, o número de pessoas que tem o mesmo consumo médio seguido de um hífen e este consumo médio, arredondando o valor para baixo. Na terceira linha da saída deve-se mostrar o consumo médio por pessoa da cidade, com 2 casas decimais sem arredondamento, considerando o consumo real total, como mostrado no exemplo.

Após cada caso de teste imprima uma linha em branco.

A saída deve ser escrita na saída padrão.

Exemplo de entrada	**Saída para o exemplo de entrada**
3 3 22 2 11 3 39 5 1 25 2 20 3 31 2 40 6 70 2 1 1 3 2 0	Cidade# 1: 2-5 3-7 3-13 Consumo medio: 9.00 m3. Cidade# 2: 5-10 6-11 2-20 1-25 Consumo medio: 13.28 m3. Cidade# 3: 3-0 1-1 Consumo medio: 0.75 m3.

Problema G: Acampamento de Férias

Arquivo: `acampamento.[c|cpp|java]`

Nas férias de Julho, várias escolas de uma mesma região resolveram se organizar e levaram uma parte de seus alunos para um acampamento de férias por uma semana. Nestes acampamentos os alunos são divididos em chalés coletivos por gênero e idade, sempre com um supervisor ou supervisora que, além de dormirem com o grupo no chalé, também são responsáveis por criar e executar várias atividades interessantes e animadas, para todas as idades. Dentre as diversas atividades podem-se citar jogos, excursões, Gincana Musical, Gincanas Noturnas, etc.

No primeiro dia de acampamento, devido à forte chuva, as atividades recreativas ficaram limitadas e as crianças foram levadas para o ginásio de esportes. Foi realizada uma gincana e uma das atividades da mesma consistiu em agrupar as crianças em um círculo (organizado no sentido anti-horário) do qual seriam retiradas uma a uma até que sobrasse apenas uma criança, que seria a vencedora.

No momento em que entra no círculo, cada criança recebe uma pequena ficha que contém um valor de 1 a 500. Depois que o círculo é formado, conta-se, a partir da segunda criança que entrou no círculo, o número correspondente à ficha que a primeira criança detém. A criança onde o número contado cair, deve ser retirada do grupo, e a contagem inicia novamente segundo a ficha da criança que acabou de ser eliminada. Para ficar mais interessante, quando o valor que consta na ficha é par, a contagem é feita no sentido horário e quando o valor que consta na ficha é ímpar, a contagem é feita no sentido anti-horário.

A brincadeira fez muito sucesso e o administrador do acampamento pediu para que sua equipe desenvolva um programa para que no próximo evento ele saiba previamente qual criança irá ser a vencedora de cada grupo, com base nas informações fornecidas.

Entrada

A entrada contém vários casos de teste. A primeira linha de cada caso de teste contém um inteiro N ($1 \leq N \leq 100$), indicando a quantidade de crianças que farão parte de cada círculo. Em seguida, N linhas indicarão o nome e o valor que consta na ficha de cada criança, separados por um espaço, na ordem de entrada na formação do círculo inicial. O nome de cada criança não deverá ultrapassar 30 caracteres e contém apenas letras maiúsculas e minúsculas, sem acentos, e o caractere "_". O final da entrada é indicado pelo número zero.

A entrada deve ser lida da entrada padrão.

Restrições

1	$\leq$	N	$\leq$	100
1	$\leq$	valor da ficha	$\leq$	500
1	$\leq$	caracteres no nome	$\leq$	30

Saída

Para cada caso de teste, deve-se apresentar a mensagem `Vencedor(a): xxxxxx`, indicando qual é a criança do grupo que venceu a brincadeira.

A saída deve ser escrita na saída padrão.

Exemplo de entrada	Saída para o exemplo de entrada
3 Fernanda 7 Fernando 9 Gustavo 11 5 Maria 7 Pedro 9 Joao_Vitor 5 Isabel 12 Laura 8 0	Vencedor(a): Fernanda Vencedor(a): Pedro

Problema H: Lasers

Arquivo: `lasers.[c|cpp|java]`

Estamos no ano de 2048. Na órbita do planeta Terra existem várias plataformas flutuantes que defendem nosso planeta contra asteróides e outras coisas que possam vir a colidir aqui. Essas plataformas possuem canhões que atiram projéteis em qualquer coisa que chegue perto.

Um civilização alienígena, no entanto, pretende invadir nosso planeta, mas para isso precisa passar pela barreira de plataformas flutuantes. Para tal, vai se utilizar de um super multi-canhão a laser, que consegue disparar, ao mesmo tempo, milhares de feixes de raio laser, a partir de um único ponto. Assim, eles podem mirar em todas as plataformas do nosso planeta e destruí-los todos ao mesmo tempo, sem nos dar tempo de reação.

O problema maior é que os raios são tão poderosos que não se limitam a destruir as plataformas flutuantes, eles acabam atingindo a Terra depois disso. Ao atingir o solo, cada feixe de laser se transforma em uma esfera de energia, e cada esfera de energia se liga a todas as outras através de fios, também de energia, formando assim uma malha energética que não pode ser tocada. Pior que isso: a região cercada pelos feixes é destruída imediatamente. É um ataque altamente perigoso e destrutivo. Dá muita raiva.

Nosso espião intergalático Mal el Kawa conseguiu adentrar no canhão e descobriu a partir de onde os alienígenas pretendem soltar os raios. Como ele não consegue simplesmente desativar os canhões, a base de controle de espionagem decidiu que, já que não será possível salvar as plataformas, temos que estimar qual será o estrago feito em solo terrestre pelos lasers.

Assim, sabendo onde estão as plataformas e de onde sairão os lasers, você deverá definir qual será a área comprometida pela "malha de energia" que se formará na Terra.

Vale lembrar que no ano de 2033 descobriu-se que a Terra é, na verdade, plana.

Entrada

A entrada contém várias instâncias.

Cada instância começa com um inteiro N, que representa o número de plataformas flutuantes. A linha seguinte contém coordenadas inteiras (x_p,y_p,z_p), representando o local em que os lasers serão disparados. As N linhas seguintes contêm, cada uma, coordenadas inteiras (x_i,y_i,z_i), representando os locais das plataformas flutuantes. O solo terrestre é representado pelo plano XY. O ponto de disparo dos lasers sempre estará acima das plataformas. As plataformas estão em pontos distintos, e nenhum laser atinge o solo em uma coordenada (x ou y) menor que -100000 ou maior que 100000. A entrada termina quando $N = 0$.

A entrada deve ser lida da entrada padrão.

Restrições

$$\begin{array}{rcccl} 3 & \leq & N & \leq & 1000 \\ -100000 & \leq & x_p,y_p,x_i,y_i & \leq & 100000 \\ 0 & \leq & z_p,\ z_i & \leq & 100000 \end{array}$$

Saída

Para cada instância na entrada, imprima uma linha contendo o valor da área comprometida pela malha energética formada pelos lasers no solo terrestre, com duas casas decimais arredondadas.

A saída deve ser escrita na saída padrão.

Exemplo de entrada	Saída para o exemplo de entrada
4 2 2 2 1 1 1 1 3 1 3 1 1 3 3 1 3 10 13 11 5 5 5 2 9 8 2 2 1 0	16.00 25.88

Problema I: Bactérias

Arquivo: `bacterias.[c|cpp|java]`

Pietro Demazio é um terrorista italiano condenado que fugiu para o Brasil, onde conseguiu um disfarce trabalhando como programador de jogos.

Em seu novo plano de destruição do planeta, Pietro desenvolveu um novo tipo de bactéria mortal, capaz de dizimar toda a população terrestre.

Durante 4 dias, Demazio criou colônias desses microorganismos, mas ao fim do quarto dia, descobriu que o código genético das mesmas possuía um grave erro, que fazia com que as bactérias morressem depois de 4 dias de vida. Como a primeira colônia fora criada 3 dias atrás, ele rapidamente modificou o código genético de todas as bactérias já produzidas (através de radiação), de modo que elas se reproduzissem todos os dias. Tal reprodução é assexuada, e é feita por bipartição (ou seja, uma bactéria gera exatamente outra bactéria por dia).

Assim, se Pietro criou 3 bactérias no dia 1, 4 no dia 2, 2 no dia 3 e 5 no dia 4, terá no total 14 bactérias ao final do quarto dia, quando ele faz a mutação. Logo após tal mutação, elas se reproduzem, e aí teremos 28 bactérias. Como a primeira colônia (com 3 bactérias) morre ao final desse quarto dia, o número de bactérias no início do quinto dia é 25. Ao final do quinto dia, essas 25 se reproduzem, resultando em 50 bactérias. Mas como a segunda colônia (com 4 bactérias) morre ao final desse dia, no início do sexto dia tem-se 46 bactérias.

Demazio observa com atenção tal crescimento da população de bactérias, e já está planejando quando vai liberá-las para fazer o serviço. Para tal, ele precisa saber quantas bactérias existirão depois de um determinado número de dias. Ele pede a você que faça um programa que determine a quantidade de bactérias existentes depois de N dias, dadas as populações das 4 primeiras colônias.

Entrada

A entrada contém várias instâncias.

Cada instância possui duas linhas. A primeira linha possui um inteiro N, representando o dia para o qual Pietro deseja saber a população de bactérias que ele terá. A segunda linha contém quatro inteiros a_1, a_2, a_3, a_4, onde a_k representa a quantidade de bactérias criadas no dia k.

A entrada termina quando $N = 0$

A entrada deve ser lida da entrada padrão.

Restrições

$$\begin{array}{rcccl} 5 & \leq & N & \leq & 1.000.000.000 \\ 1 & \leq & a_1,\, a_2,\, a_3,\, a_4 & \leq & 1000 \end{array}$$

Saída

Para cada instância na entrada, imprima uma linha contendo a quantidade de bactérias que Pietro terá no início do dia N.

A resposta dada deve ser módulo 13371337.

A saída deve ser escrita na saída padrão.

Exemplo de entrada	**Saída para o exemplo de entrada**
5 1 2 3 4 7 9 2 3 4 0	19 101

Problema J: Combinações de dias

Arquivo: `dias.[c|cpp|java]`

Estamos no ano de 2433, e a nave Pythanic acabou de ser lançada com a primeira leva de humanos a habitar outro planeta. Tal viagem tem sido muito esperada desde que as condições de vida na Terra se tornaram extremamente difíceis após uma tentativa frustrada de um terrorista de acabar com os humanos usando bactérias mutantes, há pouco mais de 400 anos atrás. Como as bactérias foram muito malfeitas, com muitas gambiarras de última hora, tudo o que ele conseguiu fazer foi deixar um incrível mal cheiro no ambiente.

Antes que a viagem fosse feita, algumas decisões tiveram que ser tomadas com relação ao modo de vida que tais pessoas levariam no outro planeta. Uma dessas decisões foi de que a duração do dia seria a mesma em todos os planetas habitados pelos humanos. Ou seja, a palavra "dia" passa a ser simplesmente um termo que significa "24 horas", e não mais um termo que especifica uma rotação completa do planeta em torno de si mesmo. No entanto, ficou decidido que a duração do mês poderá variar de planeta para planeta. Outra decisão foi que, em um mesmo planeta, todos os meses terão o mesmo número de dias.

Preocupados com a confusão que isso poderia causar, os analistas da comissão de colonização interplanetária pediram a você para criar um programa que, dadas as durações dos meses (em dias) em dois diferentes planetas, diga quantas combinações diferentes existirão de pares (d1, d2), onde d1 é um dia do mês no planeta 1, e d2 é esse mesmo dia, mas com a numeração do mês do planeta 2 (Não precisam ser dias do mesmo mês). Você deve assumir que o primeiro dia 1/1 (ou seja, primeiro dia do primeiro mês) ocorre ao mesmo tempo nos dois os planetas.

Por exemplo, se um planeta possui 2 dias num mês e outro possui 3, teremos 6 combinações diferentes de dias: (1,1), (2,2), (1,3), (2,1), (1,2) e (2,3). Se um planeta tiver 4 dias num mês e outro possuir 2, existirão apenas 4 combinações: (1,1), (2,2), (3,1), (4,2).

Dados D1 e D2, seu programa deve determinar quantas combinações de dias existem.

Entrada

A entrada contém várias instâncias.

Cada instância é composta por apenas uma linha contendo dois inteiros $D1$ e $D2$, que correspondem ao número de dias no mês nos dois diferentes planetas. A entrada termina quando $D1 = D2 = 0$.

A entrada deve ser lida da entrada padrão.

Restrições

$1 \leq D1,D2 \leq 1.000.000.000$

Saída

Para cada instância na entrada, imprima uma linha contendo a quantidade de combinações de dias diferentes entre os dois planetas.

A resposta deve ser dada em módulo 1713371337.

A saída deve ser escrita na saída padrão.

Exemplo de entrada	**Saída para o exemplo de entrada**
4 2 10 25 0 0	4 50

Problema K: Monopólio

Arquivo: `monopolio.[c|cpp|java]`

A Maratona de Programação é um evento que acontece todos os anos com a ajuda de organizações e pessoas que são entusiastas da idéia de competições científicas. Dentre as motivações para participar dela, estão o aumento dos conhecimentos relacionados a algoritmos (e todas as áreas da matemática em geral), a melhora na estrutura de raciocínio, a melhora na velocidade de digitação e também a rede social criada com pessoas de alto nível na área de computação. Para os competidores da Unicamp, no entanto, a motivação é outra. O que leva essas pessoas a participarem da maratona é a oportunidade de viajar para ficar num hotel jogando jogos de tabuleiro[1].

Na última maratona houve competições acirradas de várias modalidades de jogos, entre eles um chamado Banco Imobiliário. O objetivo desse jogo é garantir o maior acúmulo de capital possível através de especulação imobiliária.

André foi um dos participantes do jogo, e acabou perdendo todas as partidas para seu colega de time Felipe, que foi acusado de bruxaria pela magnitude das vitórias obtidas, num jogo que teoricamente depende apenas de sorte e persuasão.

Uma das coisas que mais indignou André foi ele não ter conseguido comprar os terrenos mais caros, enquanto Felipe sempre os comprava. Felipe tentou convencer André de que não praticava mais bruxaria há anos, e que ele não caiu nos territórios caros porque a probabilidade era muito pequena.

Para ajudar Felipe a convencer André das probabilidades no tabuleiro, você escreverá um programa que, dada a descrição de um tabuleiro, calcule qual é a probabilidade de se cair numa determinada casa do tabuleiro após um número infinito de rodadas.

O tabuleiro é descrito como uma sequência de N casas. O tabuleiro é circular, ou seja, após a N-ésima casa, você vai para a primeira casa. Algumas casas são especiais, e o mandam imediatamente para outra casa. No início do jogo, todos estão na casa de número 1, e a cada rodada os jogadores lançam um dado de D lados, que vai dizer quantas casas o jogador deve avançar.

A descrição do tabuleiro é uma sequência de N inteiros, cada inteiro é:

- -1 se é uma casa normal
- K se for uma casa especial, onde K é um inteiro representando o índice da casa para onde o jogador será imediatamente enviado (o índice da primeira casa é 0). Você pode assumir que essa K-ésima casa é normal.

Por exemplo, se o tabuleiro for descrito como -1 -1 0 1 -1 0, temos um tabuleiro com 6 casas, sendo que a primeira, a segunda e a quinta casa são normais. Se o jogador cai na terceira ou sexta casa, é enviado imediatamente (na mesma rodada) para a casa de índice 0 (a primeira). Se cai na quarta casa, é enviado para a casa de índice 1 (a segunda).

Além disso, será dada a quantidade de lados do dado usado, que pode ter de 3 a 20 lados.

Entrada

A entrada contém várias instâncias.

Cada instância é composta por duas linhas. Na primeira linha, serão dados dois números inteiros N e D, separados por um espaço em branco, indicando respectivamente a quantidade de casas no tabuleiro e o número de lados no dado.

Na segunda linha serão dados N números inteiros a_1, a_2, ... a_N separados por espaços em branco, que é a descrição do tabuleiro como explicado no enunciado.

A entrada termina quando $N = D = 0$.

A entrada deve ser lida da entrada padrão.

[1] Nota do Carlinhos: coisa de campineiro isso de reunir um monte de macho num quarto ...

Restrições

$$\begin{array}{rcrcl} 3 & \leq & N & \leq & 1000 \\ 3 & \leq & D & \leq & 20 \\ -1 & \leq & a_i & \leq & N-1 \end{array}$$

Saída

Para cada instância na entrada, imprima uma linha com N números p_1, p_2, ... , p_N, onde p_k é a probabilidade (em porcentagem) de, após um número infinito de rodadas, o jogador parar na casa k. Os números devem ser separados por espaços e arredondados em 3 casas decimais arredondadas.

A saída deve ser escrita na saída padrão.

Exemplo de entrada	**Saída para o exemplo de entrada**

`7 3`
`-1 0 0 0 -1 -1 -1`
`7 12`
`-1 6 -1 2 -1 -1 -1`
`0 0`
	`100.000 0.000 0.000 0.000 0.000 0.000 0.000`
	`15.385 0.000 29.664 0.000 14.363 13.284 27.304`

Problema L: Montagem

Arquivo: `montagem.[c|cpp|java]`

Estamos no ano de 2931. Cientistas detectaram um meteoro que, em 15 meses, irá colidir com a Terra e extinguir a vida no planeta. Não há mais tempo para preparar ofensivas contra o meteoro, então só nos resta realizar nossos últimos desejos e esperar a colisão.

Um grupo de pessoas resolve se unir e realizar o último sonho de centenas de milhares de pessoas: ver o Corinthians campeão da Copa Libertadores da América. Para tal, será necessária a contratação de jogadores de grande habilidade e em forma, que também são muito caros.

Para conseguir fazer isso, eles estudaram a personalidade dos melhores jogadores do mundo, e chegaram à conclusão que alguns aceitariam jogar no Corinthians mais facilmente (isto é, seria contratado por um preço menor) se percebessem que seriam as únicas "estrelas" do time. Já outros, viriam mais facilmente caso percebessem que no time já existem outras estrelas.

Assim, através de um estudo mais detalhado das personalidades, conseguiram definir, para cada jogador, qual seria o preço para contratá-los em vários cenários.

Por exemplo, o jogador X poderia ser contratado por \$3 se fosse a única estrela do time ou por \$5 se já houvesse 1 estrela no time antes dele entrar. Já o jogador Y seria contratado por \$4 se fosse a única estrela do time, ou \$2 se já houvesse uma estrela no time.

Nesse cenário, a melhor maneira de contratar X e Y seria contratar primeiro o jogador X por \$3 e depois Y por \$2, gastando \$5 no total.

Você receberá os dados dos custos de contratação dos jogadores em cada cenário, e deverá dizer quanto os torcedores do Timão deverão arrecadar para montar o time dos sonhos e conquistar a tão sonhada Libertadores.

Entrada

A entrada contém várias instâncias.

A primeira linha contém um número inteiro N, representando a quantidade de jogadores a serem contratados. Cada uma das próximas N linhas representa um jogador. Cada uma possui N inteiros c_0, c_1, c_2, ... c_{N-1} separados por espaços, onde c_k representa o custo para se contratar o jogador se já tiverem sido contratados k jogadores.

A entrada termina quando $N = 0$.

A entrada deve ser lida da entrada padrão.

Restrições

$$2 \leq N \leq 18$$
$$1 \leq c_i \leq 1000$$

Saída

Para cada instância na entrada, imprima uma linha com um inteiro representando a quantidade mínima de dinheiro que deverá ser gasto para a contratação dos N jogadores.

A saída deve ser escrita na saída padrão.

Exemplo de entrada	Saída para o exemplo de entrada
3 4 2 4 2 2 3 3 1 5 2 1 2 2 2 0	7 3