

Problema A

Vizinhança

Arquivo fonte: vizinhanca.{c | cpp | java}

Autor: Anderson V. de Araújo (Fatec São José dos Campos)

Cinco pessoas (Alfredo, Beto, Cláudio, Darci e Ernesto) desejam comprar lotes em um terreno. Os lotes estão definidos de forma tabular como indica um exemplo de mapa da figura a baixo.

Como corretor interessado em vender os lotes, o seu problema não é convencer as pessoas a comprar os terrenos, todas se dispõem a ficar com qualquer um, desde que sejam respeitadas as diferenças pessoais que existem entre alguns. Infelizmente, as pessoas têm problemas de relacionamento e são bastante restritivas no que se refere a vizinhos. Assim, temos:

- Alfredo não quer ser vizinho de Beto, e também não quer ser vizinho de Darci;
- Beto não deve ser vizinho de Alfredo e não suportaria a vizinhança de Cláudio;
- Darci não gostaria de ter Alfredo como vizinho;
- Cláudio não quer ter Beto como vizinho;
- Ernesto não tem problemas de relacionamento com ninguém, afinal ele sempre convida todos para um samba no Brás.

Nenhum lote pode ser ocupado por mais de uma pessoa, uma pessoa pode comprar mais de um lote e uma pessoa pode ficar sem terreno. O conceito de vizinhança é definido onde um lote tem um dos lados em comum com outro lote, e não é considerado um vizinho se está na diagonal. Na figura abaixo, que mostra um mapeamento válido, o lote do Alfredo é vizinho do lote do Ernesto e de Cláudio mas não de Beto. Beto é vizinho de Ernesto mas não de Alfredo e Cláudio. Já Darci tem dois lotes e não tem vizinhos.

0	A	E	B
0	C	0	0
0	0	0	0
0	0	D	D

Seu trabalho então é verificar se os mapeamentos de entrada correspondem a uma definição de vizinhança válida, ou seja, que satisfaça as restrições.

Entrada

As primeiras informações a serem lidas são a largura (colunas-C) e comprimento (linhas-L) total do terreno onde estão os lotes a venda ($0 < C$ e $L < 1000$), nesta ordem, em uma linha separados por um espaço em branco.

Nas próximas linhas são apresentados a definição dos lotes dentro do terreno a venda. As iniciais de cada nome indicam a posição de cada uma das pessoas distribuídas nos lotes. O número 0 (zero) indica que o lote está vazio.

O terreno total a venda está definido em formato de uma matriz. Onde C é o número de colunas da matriz e L é o número de linhas. Quando for lido uma largura e comprimento igual a zero, deve-se parar de ler as entradas.

Saída

Para cada mapeamento de lotes deve ser impresso o texto “V” ou “F” (em maiúsculas e sem aspas), onde “V” indica que o mapeamento é válido e “F” que é inválido.

Exemplos

Entrada:

4 4
0 A E B
0 C 0 0
0 0 0 0
0 0 D D
3 3
A B 0
C D E
0 0 0
2 2
C E
B D
5 6
0 0 0 0 0
0 A C 0 0
0 0 0 0 0
0 0 0 B E
0 0 0 0 0
D D D D D
0 0

Saída:

V
F
F
V

Solução

O problema é um problema considerado simples que se baseia em problemas de satisfação de restrições (PSRs). Para iniciar a solução, devemos ler uma matriz de tamanho l (linhas) e c (colunas), invertidos no problema, com dois *fors*, um que vai de 0 até c e outro que vai de 0 até l . Depois, deve-se percorrer a matriz (com *fors* similares ao de leitura) verificando as restrições sobre os vizinhos definidos na descrição do problema: A não pode estar na 4-vizinhança de B e D; B não pode ser vizinho de A e C, basicamente. Para fazer a checagem de vizinhos é só verificar se a posição corrente possui restrição a uma das outras posições vizinhas: $(x+1, y)$, $(x-1, y)$, $(x, y+1)$ e $(x, y-1)$. Além disso devem ser observadas as restrições para os limites da matriz, por exemplo: se o valor ainda é válido na matriz, $x-1 \geq 0$ e $x+1 < c$.

Se a entrada possuir algum vizinho que quebre as regras de restrição imprimir “F”, caso contrário imprimir “V”.

Problema B

Será que dá, China?

Arquivo fonte: china.{c | cpp | java}

Autor: Leandro Luque (Fatec Mogi das Cruzes)

China é um aluno muito popular na Fatec Mogi e possui um passado muito peculiar. Após trabalhar por mais de 10 anos como girador de pratos no *Cirque du Soleil*, ele resolveu largar tudo e cursar ADS na Fatec Mogi das Cruzes. Segundo boatos, ele se apaixonou pela mulher barbada, que acabou fugindo com o engolidor de espadas na turnê pela Nova Zelândia.

Para impressionar suas colegas, ele costuma fazer apresentações no intervalo entre aulas. Durante a apresentação, ele coloca um prato no topo de cada vareta (são várias) e vai girando um a um. Conforme ele vai se movendo e girando novos pratos, os que ele já girou vão perdendo força e ele tem que voltar rapidamente para que os pratos não parem de girar, caiam e quebrem.

Após uma análise minuciosa de gravações das suas apresentações, ele percebeu que o tempo em segundos (T) durante o qual os pratos giram antes de parar é proporcional à força aplicada (F) a cada prato, segundo a equação $T = F * 2$. Se, por exemplo, for aplicada uma força igual a 5, o prato girará por 10 segundos antes de parar. Quando ele gira um prato que ainda não parou de girar, o tempo durante o qual o prato irá continuar girando é igual ao tempo que ainda restava de giro somado ao tempo calculado pela fórmula já citada. Para aplicar uma força F em um prato qualquer, ele gasta um tempo (TG) em segundos igual a $TG = F * 2$.

Após girar com sucesso 10 pratos simultaneamente, China deseja saber qual é a maior quantidade de pratos que ele consegue girar sem quebrar nenhum. Ele não quer testar isso com seus pratos chineses e pediu sua ajuda para escrever um programa de computador que determina, a partir de alguns dados, se algum prato quebrará.

Entrada

A entrada é composta por vários casos de teste. A primeira linha de cada caso de teste contém dois números inteiros V ($1 \leq V \leq 50$) e TM ($1 \leq TM \leq 10$), representando o número de varetas e o tempo em segundos que ele leva para, logo que terminou de girar um prato, sair da vareta onde está e ir para outra logo ao lado. A próxima linha contém um número inteiro I ($1 \leq I \leq V$), representando a vareta onde ele estará logo que a apresentação iniciar. A linha seguinte contém um número inteiro N ($1 \leq N \leq 100$), que representa a quantidade de movimentos que ele pretende fazer. As próximas N linhas contêm dois números inteiros VV ($1 \leq VV \leq V$) e F ($1 \leq F \leq 10$), representando, respectivamente, a vareta para a qual ele vai se mover e a força que ele aplicará no prato da vareta logo que chegar lá. Os movimentos serão realizados em sequência, ou seja, logo após aplicar a força no prato da vareta especificada na primeira linha, ele se moverá para a vareta especificada na segunda linha e aplicará a força especificada, e assim por diante. A entrada é finalizada por uma linha com V e TM iguais a 0.

Saída

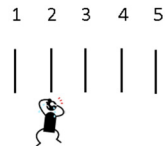
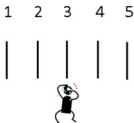
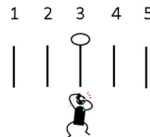
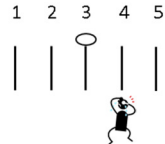
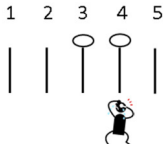
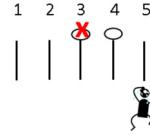
Para cada caso de teste, imprima “OK” se nenhum prato quebrar durante a sua apresentação. Caso algum prato venha a quebrar, especifique o segundo no qual o prato quebrará e qual será o prato quebrado por meio de dois números inteiros separados por um espaço em branco. Caso dois ou mais pratos quebrem ao mesmo tempo, especifique o prato com o menor número. Assuma que o tempo é contado a partir de 0.

Exemplos

Entrada:	Saída:
5 1	9 3
2	OK
10	
3 2	
4 1	
5 1	
1 10	
2 3	
5 4	
4 1	
2 3	
1 2	
2 4	
3 1	
1	
4	
1 13	
2 6	
3 2	
2 1	
0 0	

Explicação do caso de teste:

O cenário ilustrado pelo primeiro caso de teste contém 5 varetas, conforme a figura seguinte:

<p>Instante: 0s</p>  <p>China começa na vareta 2.</p>	<p>Instante: 1s</p>  <p>Como ele demora 1s para se mover entre varetas contíguas, ele chegara a vareta 3 no instante 1s. Ele demorará 4s para girar o prato da vareta 3, antes de seguir em frente.</p>	<p>Instante: 5s</p>  <p>No instante 5s, ele já girou o prato da vareta 3 e irá se dirigir a próxima vareta: 4.</p>
<p>Instante: 6s</p>  <p>Como ele demora 1s para se mover entre varetas contíguas, ele chegou a vareta 4 no instante 6s. Ele demorará 2s para girar o prato da vareta 4, antes de seguir em frente.</p>	<p>Instante: 8s</p>  <p>No instante 8s, ele já girou o prato da vareta 4 e irá se dirigir a próxima vareta: 5.</p>	<p>Instante: 9s</p>  <p>Como ele demora 1s para se mover entre varetas contíguas, ele chegou a vareta 5 no instante 9s. No entanto, o prato da vareta 3 para de girar (pela força aplicada, ele giraria por 4s – como começou no instante 5s, para no instante 9s)</p>

Solução

O problema poderia ser resolvido por simulação discreta. Uma das estratégias seria executar um *loop* que simula, a cada iteração, 1 segundo da demonstração do China. Desta forma, a cada passagem no *loop*, seria possível verificar o estado atual de cada prato e, de acordo com o segundo atual, se algum quebraria. Caso algum quebrasse, bastaria exibir a identificação do prato e o tempo (em segundos) atual.

Problema C

Sistema de Matrícula

Arquivo fonte: matricula.{c | cpp | java}

Autor: Julio Fernando Lieira (Fatec Lins)

“Todo ano é o mesmo sacrifício, passamos a noite na Fila!”, reclama Dona Maria do Socorro, 54, se referindo ao método de matrícula por ordem de chegada utilizado pela Escola pública do seu bairro. Para conseguir uma vaga é preciso chegar muito cedo, ou até mesmo chegar no dia anterior. Ciente do problema, o novo Secretário da Educação determinou que fosse criado um Sistema Informatizado de Matrícula para evitar as Filas e que também trouxesse maior transparência e justiça na distribuição das vagas. Foi elaborado um conjunto de critérios para classificar as inscrições, listados a seguir em ordem de prioridade:

- 1º – Proximidade da moradia com a Escola; (maior prioridade)
- 2º – Renda Familiar- Aluno com situação sócio-econômica menos favorável terá prioridade;
- 3º - Bolsa Família - Beneficiar quem já faz parte de outro programa do governo;
- 4º- Irmão matriculado – Alunos que tiverem irmãos naquela escola terão prioridade de matrícula na mesma Escola.
- 5º - Mãe que trabalha fora de casa e precisa ter o filho em tempo integral na Escola.

A maior prioridade é se o aluno mora próximo à Escola. Por exemplo, um aluno que mora próximo à Escola, mas que não atenda a nenhum dos outros critérios, só perderá (ficará atrás) de outro aluno que também mora próximo à Escola, mas que possua uma Renda Familiar menor. Um aluno que não mora perto da Escola, mas que atenda a todos os outros critérios (2, 3, 4 e 5) não pode ficar a frente de um aluno que mora próximo à Escola. A Tabela a seguir mostra uma lista de alunos já classificados obedecendo aos critérios estabelecidos.

Classificação	Nome	Mora Próximo à Escola	Renda Familiar	Bolsa Família	Irmão já Matriculado	Mãe Trabalha Fora
1	João Sanches	SIM	400	SIM	NÃO	NÃO
2	Maria Silva	SIM	800	SIM	SIM	SIM
3	José Frausino	SIM	1200	SIM	SIM	SIM
4	Antonio Alves	SIM	1200	SIM	SIM	NÃO
4	Bruno Souza	SIM	1200	SIM	SIM	NÃO
5	Gabriel Torres	NÃO	650	NÃO	NÃO	NÃO
6	Clara Nunes	NÃO	651	NÃO	SIM	SIM
7	Felizbaldo Feliz	NÃO	1800	NÃO	SIM	SIM

Note que um critério maior não pode perder para a soma dos critérios de prioridade menor. Ou seja, se o aluno atende ao critério 1 (mora próximo à Escola), ele não vai perder para quem atende todos os outros critérios (2, 3, 4, e 5), mas não atende o critério 1. Se o aluno atende ao critério 2 (Renda Familiar) ele não vai perder para quem atende aos critérios 3, 4 e 5, mas possui uma Renda Familiar maior. Se o aluno só atende ao critério 3 ele deve ficar à frente de um aluno que somente atende aos critérios 4 e 5.

Como visto na Tabela anterior, mesmo com todos os critérios ainda é possível haver empate. Nestes casos, o sistema deve imprimir os alunos que empataram com o mesmo número de classificação e em ordem alfabética do nome.

Entrada

Cada linha da entrada corresponde aos dados de um aluno, contendo as seguintes informações separadas por ponto e vírgula: nome;S ou N indicando se mora próximo à Escola;Renda Familiar sem casa decimal;S ou N indicando se possui Bolsa Família; S ou N indicando se possui irmãos já matriculados na Escola;S ou N indicando se a Mãe trabalha fora

Toda linha termina com um caractere fim de linha.

O nome do aluno possui no máximo 30 caracteres. A Renda Familiar pode variar de 0 a 10000.

O arquivo de entrada pode possuir no máximo 500 linhas. A entrada termina com o fim do arquivo de entrada.

Saída

Imprima na saída a classificação contendo a posição e o nome do respectivo aluno, separados por um espaço em branco. A última linha da saída deve terminar com um caractere de fim de linha.

Exemplos

Entrada:	Saída:
Gabriel Torres;N;650;N;N;N	1 João Sanches
Bruno Souza;S;1200;S;S;N	2 Maria Silva
Jose Frausino;S;1200;S;S;S	3 José Frausino
Joao Sanches;S;400;S;N;N	4 Antonio Alves
Felizbaldo Feliz;N;1800;N;S;S	4 Bruno Souza
Clara Nunes;N;651;N;S;S	5 Gabriel Torres
Maria Silva;S;800;S;S;S	6 Clara Nunes
Antonio Alves;S;1200;S;S;N	7 Felizbaldo Feliz

Solução

Uma possível solução para o problema seria atribuir valores de acordo com cada critério utilizado na classificação. Todos recebem um valor inicial igual e depois adiciona-se o valor associado ao critério, caso o aluno o tenha, gerando uma pontuação. Por exemplo:

- Valor Inicial: 50000
- Moradia Próximo: + 40000
- Renda Familiar: subtrair o valor da renda
- Bolsa Família: 0.5
- Irmão: 0.2
- Mãe Trabalha Fora: 0.1

Existe a necessidade do valor inicial, pois o valor da renda é subtraído da pontuação do aluno, pois quanto maior a renda, pior deve ser a classificação. Note que os valores escolhidos satisfazem as regras de classificação, no que diz respeito a um critério maior não pode perder para a soma dos critérios menores.

Assim, o algoritmo se resume a:

- 1- Entrada de dados (um vetor de *structs*);
- 2- Atribuir a pontuação segundo os critérios;
- 3- Ordenar pela pontuação;
- 4- Ordenar alfabeticamente os casos onde houve empate na pontuação;
- 5- Imprimir considerando o mesmo lugar os casos onde houve empate.

Problema D

Paint não, filho!

Arquivo fonte: paint.{c | cpp | java}

Autor: Leandro Luque (Fatec Mogi das Cruzes)

Kaio Damásio Ernane, o famoso KDE, recebeu seu nome numa homenagem prestada pelo seu pai, um professor de Computação fanático por Linux, ao ambiente *desktop* deste sistema operacional que leva o mesmo nome. Para o pai de Kaio, o Windows e tudo feito para ele não presta.

Em toda oportunidade que tem, o pai de Kaio procura desmerecer o Windows. É piada para cá e para lá: “Windows é igual rodízio de carros. Quando você começa a entender, vem alguém e muda tudo de novo...”, “O assistente do Windows é como irmão mais novo, tá sempre enchendo o saco...”, “O Windows é tão lento, tão lento, que não tem tempo de resposta, tem prazo de entrega...”, e por aí vai.

Ironicamente, o KDE tem uma certa queda pelo Windows e, quando seu pai não está por perto, abre o sistema numa máquina virtual escondida no seu Ubuntu, e passa horas se divertindo. Quando era pequeno, o pai de KDE o encontrou escrevendo um documento no WordPad em pleno dia de Natal. Foi a maior tristeza! Durante a ceia, ficou comentando com os amigos “Poxa... você não sabe o que me aconteceu hoje...” e contava o ocorrido com o filho.

Quando KDE cresceu e entrou no curso de ADS da Fatec Mogi, para dar uma lição no filho, seu pai passou a lhe dar castigos quando o encontrava mexendo no Windows: “Lembra quando você mexeu com Wordpad? Então... vai ter que implementar um editor de textos melhor que ele para o Linux”. E assim foi para toda vez que ele pegou KDE mexendo no Windows.

Na última semana, após chegar do trabalho, o pai de KDE o pegou fazendo um desenho no Paint. Foi um grito que o bairro inteiro escutou: “*Paint* não! Filho. *Paint* não!” e logo veio o castigo: implementar uma versão melhor que o *Paint* para o Linux.

Apesar de ter resolvido com facilidade os últimos castigos, desta vez, KDE está com uma dúvida. Ele está implementando um algoritmo similar ao da lata de tinta do *Paint*. Em resumo, quando um usuário clica com o mouse sobre um pixel de uma imagem, a lata de tinta preenche todos os pixels que possuem exatamente a mesma cor e que estão na 4-vizinhança do pixel em questão e faz isso em seguida com os pixels pintados e assim por diante. A 4-vizinhança é o conjunto de pixels que se encontram acima, abaixo, à esquerda ou à direita do atual.

Na figura seguinte (à esquerda), por exemplo, assumindo que o primeiro pixel (canto superior esquerdo) está na linha 1, coluna 1, caso o usuário clique com a lata de tinta sobre o pixel (4,5) – linha 4, coluna 5 -, 10 pixels serão pintados – eles estão destacados na figura da direita.

10	10	35	200	78	115
10	10	35	75	212	78
115	35	200	75	75	78
212	112	75	75	75	75
25	145	75	75	75	200
36	255	10	212	78	75
115	0	0	0	150	150

10	10	35	200	78	115
10	10	35	75	212	78
115	35	200	75	75	78
212	112	75	75	75	75
25	145	75	75	75	200
36	255	10	212	78	75
115	0	0	0	150	150

KDE pediu sua ajuda para escrever um algoritmo que, dada uma matriz que representa uma imagem, com L linhas e C colunas, os valores das células desta matriz, que representam cores e um pixel onde será aplicada a lata de tinta, retorna o número de pixels que serão preenchidos por meio da lata de tinta.

Entrada

A entrada é composta por vários casos de teste. A primeira linha de cada caso de teste contém dois números inteiros L ($1 \leq L \leq 1000$) e C ($1 \leq C \leq 1000$), representando o número de linhas e colunas da imagem, respectivamente. As próximas L linhas da entrada contém C números inteiros cada, com valores maiores ou iguais a 0 e menores ou iguais a 255, representando os valores dos pixels da imagem. A próxima linha da entrada contém dois números inteiros X ($1 \leq X \leq L$) e Y ($1 \leq Y \leq C$), representando a linha e coluna do pixel onde será aplicada lata de tinta, respectivamente. O pixel do canto superior esquerdo da imagem é considerado como estando na posição (1,1). A entrada termina quando forem informados valores L e C iguais a 0.

Saída

Para cada caso de teste, imprima um número inteiro Q ($1 \leq Q \leq 1000000$) representando o número de pixels que serão pintados pela lata de tinta.

Exemplos

Entrada: 7 6 10 10 35 200 78 115 10 10 35 75 212 78 115 35 200 75 75 78 212 112 75 75 75 75 25 145 75 75 75 200 36 255 10 212 78 75 115 0 0 0 150 150 4 5 0 0	Saída: 10
--	---------------------

Solução

Trata-se de um problema no qual a análise dos limites da entrada torna-se relevante. Embora ele possa ser resolvido mais facilmente por recursão, como a entrada compreende imagens de até 1000x1000, pode haver estouro da pilha de chamadas recursivas. Caso a solução recursiva fosse uma alternativa, bastaria escrever uma função que recebe o pixel inicial, onde foi clicada a lata de tinta, e que, por meio de “ifs”, verificar se os pixels vizinhos (acima, abaixo, à esquerda e à direita) ainda não foram visitados e possuem a mesma cor. Ao encontrar um ainda não visitado e com a mesma cor, a função deve ser chamada novamente para esse, e assim sucessivamente. Uma variável global poderia ser usada para armazenar a quantidade de pixels. A solução não recursiva, que deveria ser implementada para este problema, está relacionada a uma busca em largura (BFS). A ideia envolve o uso de uma lista, pilha ou similar, para armazenar os pixels ainda não visitados e, por meio de um *loop*, visitar os pixels vizinhos e adicioná-los à lista (pilha ou similar). O *loop* só terminará quando a estrutura de dados utilizada para armazenar os pixels estiver vazia.

Problema E

Número de Euler

Arquivo fonte: euler.{c | cpp | java}

Autor: Antonio Cesar de Barros Munari (Fatec Sorocaba)

“Na matemática, o número de Euler, denominado em homenagem ao matemático suíço Leonhard Euler, é a base dos logaritmos naturais. As variantes do nome do número incluem: número de Napier, constante de Néper, número neperiano, constante matemática, número exponencial etc. A primeira referência à constante foi publicada em 1618 na tabela de um apêndice de um trabalho sobre logaritmos de John Napier. No entanto, este não contém a constante propriamente dita, mas apenas uma simples lista de logaritmos naturais calculados a partir desta.” (Wikipedia). Uma aplicação desse valor especial ocorre, por exemplo, na Matemática Financeira, ao se calcular juros compostos. Da mesma forma que outras constantes famosas da matemática, o número de Euler (às vezes chamados apenas de e) é um número irracional, ou seja, não pode ser expresso com precisão por nenhuma fração com termos inteiros, e seu valor é, aproximadamente 2,718281828459045235360287. Uma forma de se calcular o valor aproximado desse número é por meio da série de Taylor, onde $x!$ é o fatorial do número x :

$$e = \frac{1}{0!} + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \dots + \frac{1}{n!}$$

Por exemplo, se considerarmos o valor limite n como sendo 4, teremos:

$$e = \frac{1}{0!} + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} = \frac{1}{1} + \frac{1}{1} + \frac{1}{2} + \frac{1}{6} + \frac{1}{24} \cong 2,708333$$

É lógico que o valor produzido por essa sequência de 5 termos está um pouco longe do valor real de e , mas quanto mais termos acrescentarmos no cálculo, mais o resultado se aproxima do valor exato do número de Euler. Caso você não se lembre do conceito de fatorial utilizado na fórmula, ele é muito simples: o fatorial de um número inteiro x é a multiplicação de todos os inteiros de 1 até x , ou seja, o fatorial de 4 é $4*3*2*1 = 24$; o fatorial de 5 é $5*4*3*2*1 = 120$. O fatorial de zero é 1, por definição.

Sua tarefa é, dado um valor de n , calcular o valor aproximado de e pela série de Taylor descrita acima.

Entrada

A entrada possui diversos valores inteiros N menores ou iguais a 12 e ela se encerra quando o valor -1 for lido.

Saída

Para cada caso de teste imprima o valor de e com 6 casas decimais. Utilize nos cálculos dados do tipo real de precisão simples (tipo `float`).

Exemplos

Entrada:	Saída:
2	2.500000
10	2.718282
4	2.708333
-1	

Solução

É um problema de manipulação de números reais de baixo grau de dificuldade. Para resolvê-lo bastava implementar a fórmula de cálculo diretamente, tomando apenas o cuidado em se utilizar nos cálculos números reais de precisão simples (tipo `float`). Como os valores de N foram dimensionados na faixa de 0 a 12, o fatorial poderia ser calculado sem dificuldades pois caberia em uma variável de tipo inteiro padrão. Para evitar problemas de arredondamento é recomendável calcular o fatorial com variáveis inteiras e converter o valor para `float` ao utilizá-lo no cálculo da série de Taylor. Imprimir o resultado com seis casas depois da vírgula é a formatação padrão para a máscara “%f” das linguagens C e C++, embora pudesse ser obtida também com “%.6f”. Em Java esse mesmo efeito pode ser obtido por exemplo com a instrução `System.out.println(String.format(Locale.US, "%6f", valor))` ou então com `System.out.printf("%.6f\n", valor)`. No enunciado as saídas correspondentes aos segundo e terceiro casos de teste estavam invertidas e esse erro foi corrigido por meio de *clarification* pública logo no início da prova.

Problema F

Banheiro!

Arquivo fonte: banheiro.{c | cpp | java}

Autor: Anderson V. de Araújo (Fatec São José dos Campos)

Billy é o cara. Ele é famoso na noite carioca. Ele é muito conhecido por organizar festas, eventos e shows pela noite a fora.

Mesmo já trabalhando neste ramo a mais de 10 anos ele sempre tem dificuldade de dimensionar o número de banheiros femininos e masculinos que devem ser disponibilizados para que as pessoas não esperem muito e o pior aconteça ali na fila mesmo.

Ele sabe que em alguns eventos as pessoas demoram mais que em outros e calculou uma média de tempo para cada tipo de evento. Com essa informação, ele criou uma tabela com os dados, mostrada a seguir.

Evento	Tempo médio - Homem	Tempo médio - Mulher	Proporção homens (H) e mulheres (M)
Aniversário (A)	3	10	1H para cada 2M
Casamento (C)	5	15	1H para cada 3M
Show (S)	1	5	2H para cada 1M

Como ele dorme o dia todo e trabalha todas as noites, pediu pra você, um amigo que sempre quer ganhar cortesia das festas dele, criar um programinha que faça esse cálculo.

O problema consiste em descobrir em quanto tempo e qual banheiro uma certa pessoa da fila vai ocupar baseando-se no tempo médio que cada pessoa leva no banheiro. Suponha que sempre a primeira pessoa da fila é um homem. No caso de um aniversário, por exemplo, a fila poderia ser composta de: H, M, M, H, M, M, e assim por diante. As B primeiras pessoas da fila tem tempo de espera 0 (zero), onde B é o número de banheiros disponíveis no evento.

Por exemplo, suponha que o evento seja um casamento e o salão tenha dois banheiros (B=2), 1 e 2. Suponha que a pessoa número 35 na fila queira saber quanto tempo ela vai demorar pra entrar no banheiro. A pessoa na posição 1 da fila (homem) irá ocupar o banheiro 1 e a pessoa na posição 2 (mulher) irá ocupar o banheiro 2. Depois de 5 minutos a pessoa de número 3 da fila (mulher) ocupará o banheiro 1. Mais adiante depois de mais 10 minutos (total de 15, pois era mulher em um casamento) o homem que ocupa a quarta posição da fila irá para o banheiro 2, e assim por diante.

Caso dois ou mais banheiros fiquem vazios ao mesmo tempo, a primeira pessoa da fila deve seguir para o banheiro livre de menor número, e os outros respectivamente.

Entrada

As primeiras informações a serem lidas são: B que é a quantidade de banheiros ($0 < B < 100$) e T o tipo do evento (A, C ou S) em uma linha. Ambos estão separados por um espaço em branco. Quando for lido um número de banheiros (B) igual a zero, deve-se parar de ler as entradas.

Depois tem-se um número N ($0 < N < 1000$) que corresponde ao número de consultas que serão feitas. Cada consulta representa uma pergunta referente a quanto tempo a pessoa esperará e qual banheiro ela usará. Na próxima linha, tem-se N posições P ($0 < P_n < 1000$) de pessoas separadas por um espaço em branco.

Saída

Para cada grupo de consultas deve ser impresso o texto “Consulta X:” onde X indica o número do grupo. Para cada posição de pessoa lida, deve-se imprimir, em cada linha, quanto tempo em minutos a pessoa deve esperar para ocupar o banheiro, qual banheiro vai usar e se é um Homem (H) ou uma Mulher (M). Todos os números são inteiros e são separados por um espaço em branco.

Exemplos

Entrada:	Saída:
2 C	Consulta 1:
9	0 1 H
1 2 3 4 5 6 7 8 9	0 2 M
3 S	5 1 M
4	15 2 M
3 56 88 12	20 1 H
1 A	25 1 M
5	30 2 M
22 38 1 15 105	40 1 M
0	45 2 H
	Consulta 2:
	0 3 M
	41 1 H
	65 3 H
	7 1 M
	Consulta 3:
	161 1 H
	279 1 M
	0 1 H
	105 1 M
	795 1 M

Solução

Este foi um problema considerado médio na primeira fase da maratona interfatecs 2014. Segue uma das possíveis soluções do problema.

Durante a leitura das entradas, verificar qual é a pessoa na maior posição da fila para cada caso de teste. Crie um laço para simular a passagem do tempo. Esse laço para quando um contador de número de pessoas que já foram ao banheiro for igual ao máximo encontrado na leitura inicial. Dentro deste laço, faça outro laço para o número de banheiros (B). Dentro do segundo laço, verifique se o tempo é zero ou tem alguma pessoa saindo do banheiro corrente para decidir se a pessoa na primeira posição da fila deve seguir para o banheiro corrente no tempo atual. Para descobrir se tem alguma pessoa saindo do banheiro, deve-se verificar se o tempo corrente módulo o tempo da pessoa no banheiro, dependendo do tipo do evento e do sexo da pessoa que está ocupando o banheiro, é zero. Se a verificação anterior for verdadeira, armazene a pessoa do início da fila no banheiro e guarde o tempo em que ela entrou e o seu sexo.

Para a saída, percorra os dados armazenados para cada caso de teste perguntado e responda o tempo em que ela entrou no banheiro e o seu sexo separados por um espaço.

Problema G

Não matem os Cangurus!

Arquivo fonte: canguru.{c | cpp | java}

Autor: Julio Fernando Lieira (Fatec Lins)









Canguru é o nome genérico dado a um mamífero marsupial pertencente a quatro espécies do gênero *Macropus* da família *Macropodidae*, que também inclui os *wallabees*. As características incluem patas traseiras muito desenvolvidas e a presença de uma bolsa (o marsúpio) presente apenas nas fêmeas na qual o filhote completa seu desenvolvimento. Conhecido por seus pulos é bastante encontrado na Austrália.

O canguru-vermelho é o maior marsupial do mundo. Os machos podem medir 1,4 metro da cabeça aos pés e pulam usando suas fortes pernas em uma grande velocidade, chegando a alcançar 56Km/h. Cada salto pode cobrir até 9 metros de distância em uma altura de 1,8 metros.

Na Austrália já existe uma superpopulação de cangurus, a ponto de o governo australiano permitir que 10% a 14% sejam caçados anualmente, em uma tentativa de diminuir os prejuízos que os saltitantes engraçadinhos estão causando nas fazendas e campos de golfe.

Uma organização não governamental denominada ADC (Associação de Defesa dos Cangurus), está pesquisando novos métodos de controle da superpopulação dos “bichinhos” sem ter que sacrificá-los. Para tanto a ADC mantém alguns cangurus em cativeiro. O cativeiro é composto por 16 cercados de 3 metros quadrados, formando uma matriz 4x4. Neste cercado são mantidos sempre 8 cangurus, cada qual em um cercado individual. Por questões ainda desconhecidas, não pode haver nem mais nem menos do que 2 cangurus nos cercados tanto na horizontal quanto na vertical (pensando em matriz, cada linha e cada coluna da matriz tem que ter 2 cangurus).

O problema é que Jack, nome dado ao canguru mais teimoso do cativeiro, sempre dá um jeito de saltar para outro cercado vazio, desfazendo a harmonia de 2 cangurus em cada linha ou coluna. Sua tarefa é, dada uma matriz representando os cangurus em cada cercado (valor 0 representa cercado vazio e valor 1 representa cercado contendo um canguru), descobrir se a composição está correta ou, caso não esteja, para qual cercado Jack tem que voltar. No exemplo abaixo, temos que mover o canguru da célula (2,3) para a célula (4,2).

	1	2	3	4
1				
2				
3				
4				

1	1	0	0
1	0	1	1
0	0	1	1
0	0	1	0

Entrada

Cada caso de teste inicia com o número do caso de teste N ($1 \leq N \leq 100$). As quatro linhas seguintes representam a matriz 4x4 do cercado. Valor zero (0) significa cercado vazio e um (1) significa cercado com canguru. Cada valor (0 ou 1) separado por um espaço em branco.

Saída

Para cada caso de teste, imprima na saída a identificação do caso de teste no formato “Caso N”, onde N é o número do caso de teste, seguido por um sinal de dois pontos (:), seguido de um espaço e, caso a matriz esteja em harmonia (somente 2 cangurus em cada linha e cada coluna) escreva CORRETO, caso contrário, diga qual canguru tem que ser movido de onde para onde, conforme o exemplo abaixo. Cada palavra deve ser separada por um espaço em branco.

Exemplos

Entrada: 1 1 1 0 0 1 0 1 1 0 0 1 1 0 0 1 0 2 0 1 0 1 1 1 0 0 0 0 1 1 1 0 1 0	Saída: Caso 1: MOVER CANGURU DE (2,3) PARA (4,2) Caso 2: CORRETO
---	---

Solução

Este problema se resume em calcular o total em cada linha e cada coluna. Caso todas as linhas e colunas apresentem total de 2, significa que tem 2 cangurus em cada linha e coluna e, portanto, está em harmonia. Caso não esteja, o canguru a ser movido será aquele que estiver na célula da matriz cujo total for 3 tanto na linha quanto na coluna. No exemplo abaixo, seria o canguru da linha 2 e coluna 3. Este canguru deve ser movido para a linha e coluna que estiverem somente com um total de 1 canguru. No exemplo abaixo seria a linha 4 coluna 2.

					Total
	1	1	0	0	2
	1	0	1	1	3
	0	0	1	1	2
	0	0	1	0	1
Total	2	1	3	2	

Problema H

Valor do Troco

Arquivo fonte: troco.{c | cpp | java}

Autor: Antonio Cesar de Barros Munari (Fatec Sorocaba)

Juquinha está fazendo um bico nas férias como caixa da mercearia da família. Tudo estava correndo bem até que uma pane inutilizou o software de cobrança do caixa e todos os aplicativos para interface gráfica, ele só pode usar linha de comando e programas em modo texto. Então ele pediu a você, o amigo nerd, para fazer um programa capaz de ajudá-lo até que o técnico venha fazer a manutenção da máquina, daqui a 3 dias. Ele precisa de um programa que receba a quantidade de cada produto comprado e o respectivo preço unitário, o valor fornecido em dinheiro e, com base nesses valores, determine o valor total da compra bem como o troco a ser devolvido ao cliente.

Entrada

Inicialmente um valor N é informado, indicando a quantidade de casos de teste a serem processados. Cada caso inicia com um inteiro P ($0 < P \leq 50$) indicando a quantidade de produtos comprados. Seguem P linhas contendo cada uma um inteiro Q ($0 < Q < 100$) e um inteiro V ($0 < V < 1000$), separados por um espaço em branco, representando a quantidade comprada e o valor unitário do produto. Em seguida é informado um inteiro D indicando a quantidade de dinheiro fornecida pelo cliente.

Saída

Para cada caso de teste imprima o valor do troco a ser devolvido. Caso o dinheiro fornecido pelo cliente seja insuficiente para pagar o valor devido, imprimir a mensagem “DINHEIRO INSUFICIENTE”, em maiúsculas.

Exemplos

Entrada:	Saída:
3	66
3	DINHEIRO INSUFICIENTE
1 2	0
2 1	
3 10	
100	
5	
1 5	
64 14	
77 2	
64 7	
62 40	
50	
2	
2 3	
1 2	
8	

Solução

Este problema era talvez o mais fácil de toda a prova e requeria apenas a manipulação numérica básica de inteiros. Recebe-se inicialmente a quantidade de casos de teste e em seguida, para cada caso, montar um *looping* para receber a quantidade vendida e o valor unitário e acumular o produto desses valores. Na saída do *looping* receber o total de dinheiro apresentado pelo cliente. Se essa quantia for menor que o total acumulado, imprimir a mensagem “DINHEIRO INSUFICIENTE”, em caso contrário, subtrair o valor calculado do total de dinheiro fornecido pelo cliente e imprimir essa diferença. Repetir o processamento para todos os casos a processar.

Problema I

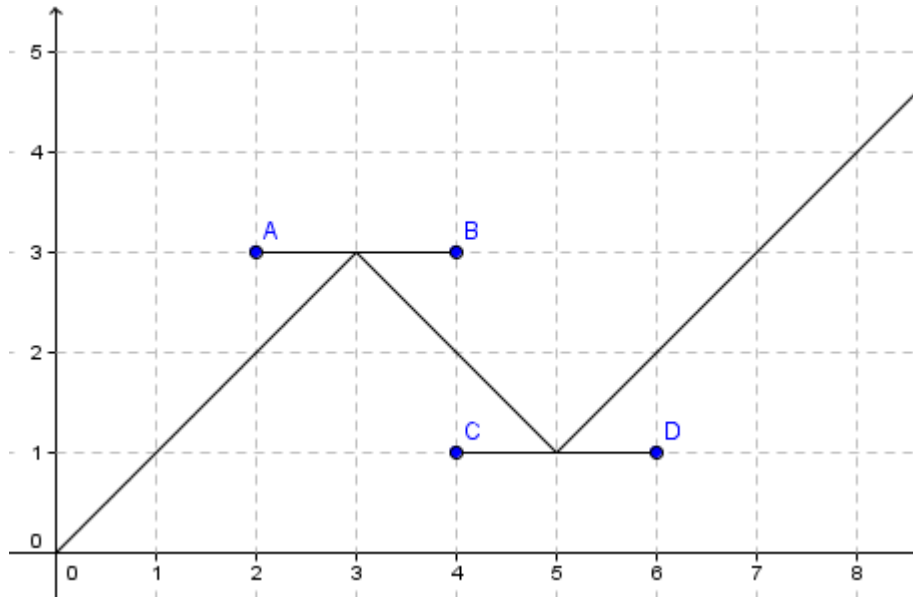
Reflexão de Espelhos

Arquivo fonte: `reflexao.{c | cpp | java}`

Autor: *Gabriel Luís Mello Dalalio (ITA)*

Adaptado por: *Anderson V. de Araújo (Fatec SJC)*

Você será designado a programar uma simulação de um laser que poderá ser refletido por dois espelhos.



A simulação acontece no primeiro quadrante de um eixo cartesiano ortogonal. Um laser será emitido a partir da origem em direção ao ponto (1;1). Haverá dois espelhos representados por dois segmentos situados totalmente no primeiro quadrante. O laser se propaga em linha reta e reflete nos segmentos de forma que o ângulo de incidência é igual ao ângulo de reflexão.

O seu trabalho será dizer se o laser será refletido pelos dois espelhos ou não. A figura mostrada acima representa um caso onde ocorre as duas reflexões.

Observações:

- Caso o laser toque um dos segmentos exatamente em um de seus extremos, o laser não é refletido e não se propaga mais;
- O laser pode refletir em ambos os lados dos espelhos;
- O espelho que pode refletir o raio pela segunda vez sempre vai estar paralelo a um dos eixos cartesianos.

Entrada

A entrada é iniciada por um inteiro T , $0 < T \leq 1000$, que indica a quantidade de casos de teste a serem processados. Seguem-se T linhas cada uma contendo 8 inteiros $Xa, Ya, Xb, Yb, Xc, Yc, Xd, Yd$. Todos os inteiros são positivos e menores que 1000. Os inteiros indicam respectivamente as coordenadas dos pontos A, B, C e D , primeiro a coordenada horizontal e depois a coordenada vertical. Os espelhos são representados pelos segmentos AB e CD e eles podem refletir o laser independente de qual dos lados do segmento o laser atinja. Não haverá intersecção entre os dois espelhos. Caso haja dúvida, o primeiro exemplo de entrada representa o caso mostrado na figura do enunciado.

Saída

Para cada caso de teste, o programa deve imprimir uma linha contendo a letra S caso haja a reflexão do laser nos dois espelhos, ou contendo a letra N caso contrário. Vale ressaltar que pode ocorrer mais reflexões após as duas primeiras, mesmo assim nesse caso a resposta é S, só estamos interessados nas duas primeiras reflexões.

Exemplos

Entrada:	Saída:
4	S
2 3 4 3 4 1 6 1	N
3 3 4 3 4 1 6 1	N
1 2 2 1 1 3 3 3	S
8 1 8 3 4 5 6 5	

Solução

Este é um problema que envolve geometria simples que foi considerado de complexidade média na prova. Um conceito importante aqui é o da equação da reta: $y=Ax+B$.

Passos:

1. Descobrir A e B do espelho 1, utilizando os pontos lidos (Xa, Ya, Xb, Yb), substituindo na equação da reta e resolvendo o sistema linear simples de 2 equações e duas variáveis;
2. Descobrir A e B do espelho 2, com (Xc, Yc, Xd, Yd) similar ao passo anterior;
3. Suponha que o raio laser que sai da origem corta o espelho 1, então existe um ponto de intersecção entre os dois onde $x=y$ (equação da reta correspondente ao laser);
 - a. Devemos descobrir qual é o ponto em que o raio corta o espelho ($x1r1=y1r1$). Para isso, usar a equação: $x1r1 = 1 - AE2 == 0 ? BE2 : BE2 / Math.abs(1 - AE2)$
 - b. Verificar a intersecção dos segmentos de reta definidos por ($0, 0, x1r1, y1r1, Xa + 1, Ya, Xb - 1, Yb$)
 - c. Caso os segmentos se interceptem, descobrir as constantes AE2 e BE2, lembrando que o segundo espelho que pode refletir o raio é sempre paralelo a um dos eixos;
 - d. Com AE2 e BE2 descubra $xr2$ e $yr2$ (pontos do raio 2 - refletido) e verifique se eles interceptam o espelho 2.
4. Suponha que o raio laser que sai da origem corta o espelho 2, então existe um ponto de intersecção entre os dois onde $x=y$ (equação da reta correspondente ao laser);
 - a. Devemos descobrir qual é o ponto em que o raio corta o espelho ($x2r1=y2r1$). Para isso, usar a equação: $1 - AE2 == 0 ? BE2 : BE2 / Math.abs(1 - AE2)$
 - b. Verificar a intersecção dos segmentos de reta definidos por ($0, 0, x2r1, y2r1, Xc + 1, Yc, Xd - 1, Yd$)
 - c. Caso os segmentos se interceptem, descobrir as constantes AE2 e BE2, lembrando que o segundo espelho que pode refletir o raio é sempre paralelo a um dos eixos;
 - d. Com AE2 e BE2 descubra $xr1$ e $yr1$ (pontos do raio 1 - refletido) e verifique se eles interceptam o espelho 1.
5. Se $abs(Xa - Xb) = 1$ então o raio nunca corta o espelho 1;
6. Se $abs(Xc - Xd) = 1$ então o raio nunca corta o espelho 2;
7. Se o raio reflete em ambos os espelhos escreva 'S' caso contrário 'N';

Problema J

Product of Products

Arquivo fonte: products.{c | cpp | java}

Autor: Antonio Cesar de Barros Munari (Fatec Sorocaba)

Our company sells various products and we need to sum both the cost involved as well as the profit in this market. The input data is defined in two matrices, one indicating the products sold per month and another indicating the cost and profit per unit associated with each product. We need to produce a similar matrix containing the aggregate cost of the products sold, together with the profit. The Figure 1 illustrates a case where we have three different pieces and three months, with its result.

<u>Quantity per Month</u>				<u>Values per Product</u>		
	P1	P2	P3		COST	PROFIT
JAN	1	2	3	P1	1	0
FEB	4	5	6	P2	2	1
MAR	7	8	9	P3	4	2

<u>Total per Month</u>		
	COST	PROFIT
JAN	17	8
FEB	38	17
MAR	59	26

Fig. 1: Data received and the corresponding result.

Input

The input have many test cases. Initially, a N value is informed, indicating the quantity of test cases to be processed. Each test case starts with two integers P ($0 < P \leq 20$) and M ($0 < M \leq 12$) indicating, respectively, the quantity of sold products and the months to be considered. Next, two matrices must be read, the first containing M rows with P integers Q ($0 \leq Q \leq 100$), separated by a single white space, indicating the sold products sold in each month and the second containing P rows with two integers C and L ($0 \leq C, L \leq 25$) in each row, indicating respectively the cost and profit by unit of each product.

Output

For each test case, print the number of the test case, as shown in the examples, followed by the matrix containing M rows, each one with two integers, that represents the total cost and profit of each month.

Examples

Input:	Output:
2	Case 1:
3 3	17 8
1 2 3	38 17
4 5 6	59 26
7 8 9	Case 2:
1 0	2333 2892
2 1	5174 5976
4 2	3518 4194
5 12	5904 6962
25 18 32 38 49	4504 5493
64 92 62 52 67	3867 3789
1 39 83 62 43	3114 4573
99 88 89 74 45	4020 5403
33 89 87 73 13	6319 7376
69 77 62 7 15	3443 4159
9 46 15 77 87	4630 5937
3 57 77 95 48	5127 5155
86 70 93 78 95	
22 53 49 49 55	
60 97 26 65 73	
32 95 63 13 100	
13 14	
18 20	
21 14	
6 25	
16 16	

Solução

Este é um problema clássico de multiplicação de matrizes, que pode ser resolvido com o algoritmo padrão para esse tipo de situação. Temos uma matriz de Quantidades com dimensões M por P e outra de Valores Unitários com dimensões P por 2, e devemos calcular a soma do produto dos valores unitários de cada produto pela quantidade vendida. Isso pode ser obtido pela combinação de três *loopings*:

```
for( i=0; i<M; i++ )
    for( j=0; j<2; j++ )
        for( k=0; k<P; k++ )
            Resultado[i][j] += Qtde[i][k] * ValorUnitario[k][j];
```

Calculada a matriz de resultados, bastava imprimir os seus valores, para o que era necessário atenção para que não ficasse um espaço em branco no final da linha na saída. Soluções corretas que descuidavam desse aspecto receberam veredito “NO – Presentation Error”, o que ocorreu com 11% das submissões na prova.