acm International Collegiate Programming Contest 2011 IBM event sponsor

Maratona de Programação da SBC 2011

Sub-Regional Brasil do ACM ICPC

17 de Setembro de 2011

Caderno de Problemas

Instruções

1) Este caderno contém 9 problemas; as páginas estão numeradas de 1 a 19, não contando esta página de rosto. Verifique se o caderno está completo.
2) Em todos os problemas, a entrada de seu programa deve ser lida da *entrada padrão*. A saída deve ser escrita na *saída padrão*.
3) Em todos os problemas, o final da entrada coincide com o final do arquivo.

Promoção:

SBC Sociedade Brasileira de Computação

Patrocínio:

Fundação Carlos Chagas

v1.0

Problema A

Ácido Ribonucléico Alienígena

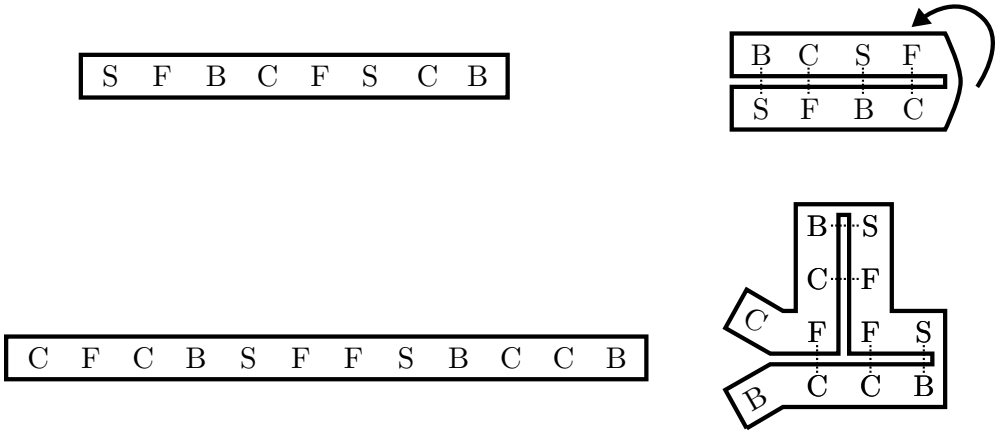
Nome do arquivo fonte: `acido.c`, `acido.cpp`, *ou* `acido.java`

Foi descoberta uma espécie alienígena de ácido ribonucléico (popularmente conhecido como RNA). Os cientistas, por falta de criatividade, batizaram a descoberta de *ácido ribonucléico alienígena* (RNAA). Similar ao RNA que conhecemos, o RNAA é uma fita composta de várias bases. As bases são `B`, `C`, `F` e `S` e podem ligar-se em pares. Os únicos pares possíveis são entre as bases `B` e `S` e as bases `C` e `F`.

Enquanto está ativo, o RNAA dobra vários intervalos da fita sobre si mesma, realizando ligações entre suas bases. Os cientistas perceberam que

- quando um intervalo da fita de RNAA se dobra, todas as bases neste intervalo se ligam com suas bases correspondentes;
- cada base pode se ligar a apenas uma outra base;
- as dobras ocorrem de forma a maximizar o número de ligações entre bases.

As figuras abaixo ilustram dobras e ligações feitas sobre fitas.



Sua tarefa será, dada a descrição de uma tira de RNAA, determinar quantas ligações serão realizadas entre suas bases se a tira ficar ativa.

Entrada

A entrada é composta por diversos casos de teste. Cada caso de teste possui uma linha descrevendo a sequência de bases da fita de RNAA.

Saída

Para cada instância imprima uma linha contendo o número total de ligações que ocorre quando a fita descrita é ativada.

Restrições

- Uma fita de RNAA na entrada contém pelo menos 1 base e no máximo 300 bases.
- Não existem espaços entre bases de uma fita da entrada.
- As bases são '`B`', '`C`', '`F`' e '`S`'.

Exemplos

Exemplo de entrada	**Exemplo de saída**

`SBC`
`FCC`
`SFBC`
`SFBCFSCB`
`CFCBSFFSBCCB`
	`1`
	`1`
	`0`
	`4`
	`5`

Problema B

Botas perdidas

Nome do arquivo fonte: `botas.c`, `botas.cpp`, *ou* `botas.java`

A divisão de Suprimentos de Botas e Calçados do Exército comprou um grande número de pares de botas de vários tamanhos para seus soldados. No entanto, por uma falha de empacotamento da fábrica contratada, nem todas as caixas entregues continham um par de botas correto, com duas botas do mesmo tamanho, uma para cada pé. O sargento mandou que os recrutas retirassem todas as botas de todas as caixas para reembalá-las, desta vez corretamente.

Quando o sargento descobriu que você sabia programar, ele solicitou com a gentileza habitual que você escrevesse um programa que, dada a lista contendo a descrição de cada bota entregue, determina quantos pares corretos de botas poderão ser formados no total.

Entrada

A entrada é composta por diversos casos de teste. A primeira linha de um caso de teste contém um inteiro N indicando o número de botas individuais entregues. Cada uma das N linhas seguintes descreve uma bota, contendo um número inteiro M e uma letra L, separados por um espaço em branco. M indica o número do tamanho da bota e L indica o pé da bota: $L =$ '`D`' indica que a bota é para o pé direito, $L =$ '`E`' indica que a bota é para o pé esquerdo.

Saída

Para cada caso de teste imprima uma linha contendo um único número inteiro indicando o número total de pares corretos de botas que podem ser formados.

Restrições

- $2 \leq N \leq 10^4$
- N é par.
- $30 \leq M \leq 60$
- $L \in \{\texttt{D}, \texttt{E}\}$

Exemplos

Exemplo de entrada	**Exemplo de saída**
4 40 D 41 E 41 D 40 E 6 38 E 39 E 40 D 38 D 40 D 37 E	2 1

Problema C
Campeonato de SMS

Nome do arquivo fonte: `campeonato.c`, `campeonato.cpp`, *ou* `campeonato.java`

A Só Birutas Celulares, uma renomada empresa do ramo de telefonia móvel, promove um campeonato de mensagens de texto todos os anos. Neste campeonato, ganha quem digitar uma dada mensagem mais rápido.

O aparelho oficial da competição, de uso obrigatório, tem um teclado muito simples, similar ao celular que você provavelmente teria no bolso se aparelhos eletrônicos não fossem proibidos durante a Maratona de Programação. O teclado tem o seguinte *layout*:

Como só é permitido o uso dos polegares para pressionar as teclas, todas elas foram feitas quadradas, com 1 centímetro de lado, sem espaço entre duas teclas adjacentes.

As teclas de `2` a `9` são usadas para digitar as letras de '`a`' a '`z`', e funcionam como em qualquer celular: se quisermos obter uma das letras associadas a uma das teclas, precisamos pressioná-la um número de vezes igual à posição da letra desejada. Por exemplo, pressionando a tecla `3` uma vez obtemos '`d`'. Se pressionarmos novamente, obteremos '`e`' e depois '`f`'. Se continuarmos pressionando-a obteremos o número '`3`' e depois reiniciamos em '`d`'. A tecla `0` é utilizada para inserir espaços na mensagem; as teclas `1` e `*` não são utilizadas nesta competição.

No caso de termos duas letras consecutivas na mensagem que são formadas pela mesma tecla será necessário fazer uso da tecla `#`. A função desta tecla é separar as sequências de pressionamentos de duas letras na mesma tecla. Por exemplo, para digitar a palavra "`casa`", a sequência de teclas pressionadas seria a seguinte: `2`, `2`, `2`, `#`, `2`, `7`, `7`, `7`, `7`, `2`.

Para tornar as coisas mais interessantes, a organização decidiu que este ano as mensagens devem ser digitadas em queda livre: os competidores pulam de um avião com o celular em mãos e digitam a mensagem; um sofisticado sistema computadorizado abrirá o paraquedas automaticamente quando a mensagem tiver sido digitada sem erros. Entretanto, essa modificação das regras introduziu uma dificuldade a mais: para evitar que o celular se perca durante a queda, é necessário utilizar um polegar para segurar o aparelho enquanto o outro pressiona uma tecla ou é movido; ou seja, um dos polegares está sempre fixo.

Para satisfazer a curiosidade da platéia, você foi contratado para fazer um programa de computador que, dada uma mensagem de até 140 caracteres, responde o tempo mínimo necessário para um competidor ideal digitá-la no celular. Suponha que um competidor ideal consegue mover seus polegares à incrível velocidade de 30 centímetros por segundo, leva apenas 2 décimos de segundo para pressionar uma tecla, inicia a queda livre com o polegar esquerdo sobre a tecla `4`, o polegar direito sobre a tecla `6` e sempre pressiona as teclas perfeitamente em seus centros.

Entrada

A entrada é composta por diversos casos de teste. Cada caso de teste é descrito em uma linha contendo uma mensagem.

Saída

Para cada caso de teste imprima uma linha contendo o tempo, em segundos, que nosso competidor ideal levaria para digitar a mensagem dada. Utilize duas casas decimais para exibir a resposta.

Restrições

- Toda mensagem tem entre 1 e 140 caracteres, inclusive.
- Toda mensagem é composta apenas de letras minúsculas de '`a`' até '`z`' e espaços.
- Nenhuma mensagem contém acentos.
- Nenhuma mensagem começa com espaços.
- Nenhuma mensagem termina com espaços.
- Nenhuma mensagem contém espaços consecutivos.

Exemplos

Exemplo de entrada	**Exemplo de saída**

`gol`
`ogro`
`casa`
`garra`
`paraquedas com defeito`
	`1.43`
	`2.03`
	`2.19`
	`2.23`
	`10.10`

Problema D

Desvio de rua

Nome do arquivo fonte: `desvio.c`, `desvio.cpp`, *ou* `desvio.java`

A prefeitura de uma grande cidade da Nlogônia iniciou um programa de recuperação do asfalto de suas ruas. Na Nlogônia, cada rua liga diretamente dois cruzamentos, e pode ter mão única ou mão dupla. Por determinação de um antigo decreto real, sempre existe ao menos um caminho entre dois pontos quaisquer da cidade.

No programa de recuperação, uma única rua será recuperada por vez, e para isso a rua será fechada para o tráfego. Esse fechamento pode causar caos no trânsito local ao violar o decreto real, impedindo vários cidadãos de voltarem para casa dos seus trabalhos e vice-versa. A prefeitura pode converter algumas das ruas de mão única em mão dupla, mas prefere evitá-lo pois ruas de mão dupla tendem a causar acidentes mais graves; a prefeitura prefere criar desvios apenas invertendo as mãos das ruas de mão única já existentes.

O Rei da Nlogônia solicitou seus préstimos para escrever um programa que, dada a descrição das ruas de uma cidade, determine se, quando uma dada rua é interditada para recuperação, continua existindo um caminho entre entre quaisquer dois pontos da cidade, mesmo que seja necessário alterar as mãos de direção de outras ruas.

Entrada

A entrada é composta por diversos casos de teste. A primeira linha de um caso de teste contém dois inteiros N e M, representando respectivamente o número de cruzamentos e o número de ruas da cidade. Os cruzamentos são identificados por inteiros de 1 a N e as ruas são identificadas por números inteiros de 1 a M. Cada uma das M linhas seguintes descreve uma rua e contém três inteiros A, B e T, onde A e B são os cruzamentos que a rua liga diretamente, e T indica a mão de direção da rua: se $T = 1$ a rua tem mão única na direção de A para B; se $T = 2$, a rua tem mão dupla. A primeira rua descrita será interditada para recuperação.

Saída

Para cada caso de teste seu programa deve imprimir uma linha contendo um caractere que descreve o que a prefeitura deve fazer para respeitar o decreto real após o fechamento da rua para reformas:

- '`-`': não é necessário qualquer tipo de alteração nas outras ruas.
- '`*`': é impossível respeitar o decreto real, independente de quaisquer mudanças nas outras ruas.
- '`1`': é possível cumprir o decreto real apenas invertendo as mãos de algumas das ruas de mão única.
- '`2`': é possível cumprir o decreto real, mas é necessário converter algumas ruas de mão única para mão dupla.

Restrições

- $1 \leq N \leq 10^3$
- $1 \leq M \leq 10^5$
- $1 \leq A, B \leq N$
- $T \in \{1, 2\}$

Exemplos

Exemplo de entrada	**Exemplo de saída**
3 3 1 2 2 2 3 2 3 1 2 3 3 1 2 1 2 3 1 3 1 1 2 1 1 2 2 5 6 3 4 1 1 3 2 2 4 2 3 5 2 2 1 1 4 1 1	- 2 * 1

Problema E

Estacionamento

Nome do arquivo fonte: `estacionamento.c`, `estacionamento.cpp`, *ou* `estacionamento.java`

Um estacionamento utiliza um terreno em que os veículos têm que ser guardados em fila única, um atrás do outro. A tarifa tem o valor fixo de R$ 10,00 por veículo estacionado, cobrada na entrada, independente de seu porte e tempo de permanência. Como o estacionamento é muito concorrido, nem todos os veículos que chegam ao estacionamento conseguem lugar para estacionar.

Quando um veículo chega ao estacionamento, o atendente primeiro determina se há vaga para esse veículo. Para isso, ele percorre a pé o estacionamento, do início ao fim, procurando um espaço que esteja vago e tenha comprimento maior ou igual ao comprimento do veículo. Para economizar seu tempo e energia, o atendente escolhe o primeiro espaço adequado que encontrar; isto é, o espaço mais próximo do início.

Uma vez encontrada a vaga para o veículo, o atendente volta para a entrada do estacionamento, pega o veículo e o estaciona no começo do espaço encontrado. Se o atendente não encontrar um espaço adequado, o veículo não entra no estacionamento e a tarifa não é cobrada. Depois de estacionado, o veículo não é movido até o momento em que sai do estacionamento.

O dono do estacionamento está preocupado em saber se os atendentes têm cobrado corretamente a tarifa dos veículos estacionados e pediu para você escrever um programa que, dada a lista de chegadas e saídas de veículos no estacionamento, determina o faturamento total esperado.

Entrada

A entrada é composta por diversos casos de teste. A primeira linha de um caso de teste contém dois números inteiros C e N que indicam respectivamente o comprimento em metros do estacionamento e o número total de eventos ocorridos (chegadas e saídas de veículos). Cada uma das N linhas seguintes descreve uma chegada ou saída. Para uma chegada de veículo, a linha contém a letra '`C`', seguida de dois inteiros P e Q, todos separados por um espaço em branco. P indica a placa do veículo e Q o seu comprimento. Para uma saída de veículo, a linha contém a letra '`S`' seguida de um inteiro P, separados por um espaço em branco, onde P indica a placa do veículo. As ações são dadas na ordem cronológica, ou seja, na ordem em que acontecem.

No início de cada caso de teste o estacionamento está vazio. No arquivo de entrada, um veículo sai do estacionamento somente se está realmente estacionado, e a placa de um veículo que chega ao estacionamento nunca é igual à placa de um veículo já estacionado.

Saída

Para cada caso de teste seu programa deve imprimir uma linha contendo um número inteiro representando o faturamento do estacionamento, em reais.

Restrições

- $1 \leq C \leq 1000$
- $1 \leq N \leq 10000$
- $1 \leq Q \leq 100$
- $1000 \leq P \leq 9999$

Exemplos

Exemplo de entrada	**Exemplo de saída**
10 7 C 1234 5 C 1111 4 C 2222 4 C 4321 3 S 1111 C 2002 6 C 4321 3 30 10 C 1000 10 C 1001 10 C 1002 10 S 1000 S 1002 C 1003 20 S 1001 C 1004 20 S 1004 C 1005 30 20 10 C 1234 20 C 5678 1 S 1234 C 1234 20 C 5678 1 S 1234 C 5678 1 C 1234 20 C 5555 1 S 5678	30 50 40

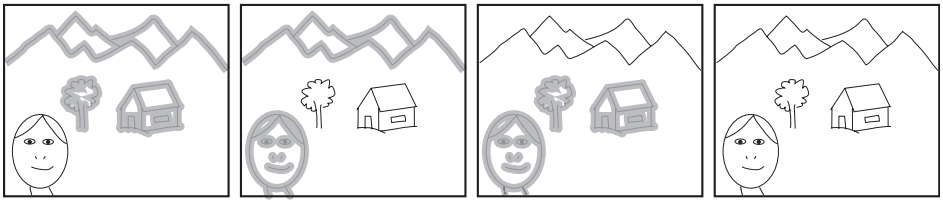
Problema F

Foco

Nome do arquivo fonte: `foco.c`, `foco.cpp`, *ou* `foco.java`

Daniel está fazendo um curso de Visão Computacional e decidiu reproduzir um trabalho muito interessante visto em aula: ele tirou várias fotos de uma mesma cena, variando apenas o foco, para depois combiná-las em uma única imagem onde todos os objetos da cena estão nítidos simultaneamente. Para tal, ele precisa que cada objeto apareça nítido em ao menos uma foto.

Daniel sabe que, para cada objeto, existe um intervalo fechado de planos de foco no qual aquele objeto fica nítido. Por exemplo, na figura abaixo, (i), (ii) e (iii) são três fotos da mesma cena, cada uma tirada com um foco diferente; (iv) é a imagem combinada gerada por Daniel.



Como o cartão de memória de sua câmera é pequeno, ele pediu sua ajuda para, dados os intervalos de foco de todos os objetos da cena que pretende fotografar, determinar o número mínimo de fotos que ele deve tirar para que seja possível deixar cada objeto nítido em pelo menos uma foto.

Entrada

A entrada é composta por diversos casos de teste. A primeira linha de cada caso de teste contém um inteiro N indicando o número de objetos na cena. Cada uma das N linhas seguintes contém dois inteiros A e B indicando os extremos do intervalo de foco de cada objeto.

Saída

Para cada caso de teste, imprima uma linha contendo um inteiro indicando o menor número de fotos que Daniel deve tirar.

Restrições

- $1 \leq N \leq 10^6$
- $1 \leq A \leq B \leq 10^9$

Exemplos

Exemplo de entrada	**Exemplo de saída**
3 1 3 2 5 4 6 5 1 2 5 6 3 4 5 6 1 2	2 3

Problema G

Guarda Costeira

Nome do arquivo fonte: `guarda.c`, `guarda.cpp`, *ou* `guarda.java`

"Pega ladrão! Pega ladrão!" Roubaram a bolsa de uma inocente senhora que caminhava na praia da Nlogônia e o ladrão fugiu em direção ao mar. Seu plano parece óbvio: ele pretende pegar um barco e escapar!

O fugitivo, que a essa altura já está a bordo de sua embarcação de fuga, pretende seguir perpendicularmente à costa em direção ao limite de águas internacionais, que fica a 12 milhas náuticas de distância, onde estará são e salvo das autoridades locais. Seu barco consegue percorrer essa distância a uma velocidade constante de V_F nós.

A Guarda Costeira pretende interceptá-lo, e sua embarcação tem uma velocidade constante de V_G nós. Supondo que ambas as embarcações partam da costa exatamente no mesmo instante, com uma distância de D milhas náuticas entre elas, será possível a Guarda Costeira alcançar o ladrão antes do limite de águas internacionais?

Assuma que a costa da Nlogônia é perfeitamente retilínea e o mar bastante calmo, de forma a permitir uma trajetória tão retilínea quanto a costa.

Entrada

A entrada é composta por diversos casos de teste. Cada caso de teste é descrito em um linha contendo três inteiros, D, V_F e V_G, indicando respectivamente a distância inicial entre o fugitivo e a Guarda Costeira, a velocidade da embarcação do fugitivo e a velocidade da embarcação da Guarda Costeira.

Saída

Para cada caso de teste imprima uma linha contendo '`S`' se for possível que a Guarda Costeira alcance o fugitivo antes que ele ultrapasse o limite de águas internacionais ou '`N`' caso contrário.

Restrições

- $1 \leq D \leq 100$
- $1 \leq V_F \leq 100$
- $1 \leq V_G \leq 100$

Exemplos

Exemplo de entrada	**Exemplo de saída**
5 1 12 12 10 7 12 9 10 10 5 5 9 12 15	S N N N S

Problema H

Homem, Elefante e Rato

Nome do arquivo fonte: `homem.c`, `homem.cpp`, *ou* `homem.java`

Um jogo muito popular na Nlogônia é o Homem, Elefante e Rato. Ele é tipicamente jogado com apenas dois jogadores, e funciona da seguinte forma: cada jogador secretamente escolhe um dos três símbolos e, após uma contagem regressiva, ambos revelam simultaneamente o símbolo escolhido através de sinais manuais, estendendo à sua frente uma das mãos sinalizando sua escolha.

O Homem é representado pela mão fechada, como a cabeça de um homem. O Elefante é representado pela mão aberta, exibindo os cinco dedos, como a pata do elefante nlogonense. Por fim, o Rato é representado pela mão fechada, com o dedo indicador e o dedo médio esticados, como as orelhas do pequeno animal.

Figura 1: Os três símbolos do jogo Homem, Elefante e Rato.

Para determinar o vencedor é muito simples: o Homem sempre perde para o Elefante (pois é esmagado debaixo de sua pata), o Elefante sempre perde para o Rato (pois tem medo dele e foge correndo) e o Rato sempre perde para o Homem (que espalha ratoeiras para capturá-lo). Se dois jogadores utilizarem o mesmo símbolo, ocorre um empate e joga-se novamente.

Os habitantes da Nlogônia, que são estrategistas natos de Homem, Elefante e Rato, utilizam a seguinte técnica no campeonato nacional, realizado todos os anos: começam sempre jogando Homem até o momento em que este símbolo causa empates com a maioria dos oponentes. Eles então trocam sua estratégia para o símbolo que ganha daquele que usavam anteriormente. Assim, os jogadores vão mudar de Homem para Elefante, depois para Rato, depois de volta a Homem.

Para auxiliar um famoso competidor estrangeiro de um jogo parecido, que é quase, mas não completamente diferente de Homem, Elefante e Rato, você irá desenvolver um programa que contabiliza quantos jogadores irão utilizar cada símbolo.

Suponha que todos os N jogadores são dispostos em fila e identificados pela sua posição, de 1 a N. Seu programa deverá processar M comandos, de dois tipos: mudança de símbolo e contar a frequência dos símbolos. Ambos os comandos recebem um intervalo contíguo de jogadores na fila a serem considerados.

Entrada

A entrada é composta por diversos casos de teste. Cada caso de teste começa com uma linha contendo dois inteiros N e M que representam, respectivamente, o número de jogadores no campeonato e o número de operações.

As próximas M linhas contêm cada uma a descrição de uma operação. Operações de mudança de estratégia serão representadas por uma linha da forma "`M` A B" onde A e B são inteiros. Os jogadores cuja estratégias serão alteradas são aqueles cuja posição na fila está entre A e B, inclusive.

Operações de contagem serão representadas por uma linha da forma "`C` A B" onde A e B são inteiros representando o intervalo de jogadores que deverão ser considerados na contagem. Levaremos em conta os jogadores cuja posição na fila está entre A e B, inclusive.

Saída

Para cada operação de contagem, imprima uma linha contendo três inteiros indicando respectivamente o número de símbolos Homem, Elefante e Rato que são usados pelos jogadores no intervalo dado.

Imprima também uma linha em branco após cada caso de teste, inclusive após o último caso de teste da entrada.

Restrições

- $1 \leq N \leq 10^5$
- $0 \leq M \leq 10^6$
- $1 \leq A \leq B \leq N$

Exemplos

Exemplo de entrada	**Exemplo de saída**

`10 7`
`C 1 10`
`M 5 6`
`C 5 6`
`M 6 7`
`C 4 8`
`M 1 10`
`C 1 10`
`5 6`
`M 1 5`
`M 2 4`
`M 1 2`
`M 4 5`
`C 1 5`
`C 3 4`
	`10 0 0`
	`0 2 0`
	`2 2 1`
	`1 7 2`
	
	`2 0 3`
	`1 0 1`

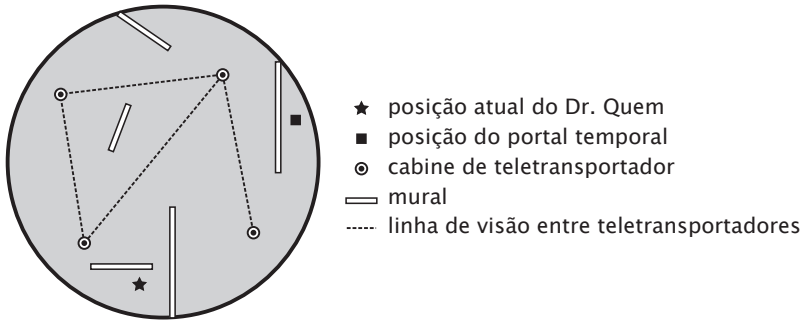
Problema I

Incidente em Atlântida

Nome do arquivo fonte: `incidente.c`, `incidente.cpp`, *ou* `incidente.java`

No ano de 2222 B.C.E., uma terrível tragédia ocorreu na ilha de Atlântida, mas até hoje só se sabe do ocorrido pelos relatos históricos; ninguém nunca encontrou evidência física do incidente. Para remediar essa situação, o Doutor Quem resolveu usar a sua máquina do tempo para voltar a Atlântida logo antes do desastre.

Precavido, assim que chegou a Atlântida o Doutor quem instalou um portal temporal, que o trará imediatamente ao tempo atual. Ele iniciou então sua exploração, e o que descobriu foi surpreendente: Atlântida é uma ilha circular, sobre a qual foi fundada uma civilização extremamente avançada; prova disso são as várias cabines da rede de teletransportadores que foi construída na ilha. As cabines operam através de raios de luz e portanto só é possível teletransportar se existir uma linha de visão desobstruída entre a cabine de origem e a cabine de destino. Não por acaso, elas são algumas das construções mais imponentes da ilha, com dezenas de metros de altura, mais baixas apenas do que os gigantes murais que descrevem a história de Atlântida desde a sua fundação. Os murais são enormes muros de concreto, que dificultam a exploração do Doutor Quem pois não podem ser transpostos, obrigando o famoso cientista a contorná-los para prosseguir em seu caminho.



Ao explorar a ilha, o Doutor Quem se mostrou surpreso em encontrá-la deserta. Depois de muito refletir, tornou-se óbvio o porquê: ele programou mal sua máquina do tempo, e chegou imediatamente antes do maremoto que destruiu Atlântida (os geólogos atlântidos já haviam previsto o maremoto e ordenado a evacuação imediata da ilha). Tomando ciência desse fato, o Doutor Quem imediatamente tratou de planejar a sua fuga.

Para voltar ao portal temporal, ele pretende utilizar os teletransportadores, mas ele descobriu que, devido ao maremoto, o sistema de energia só tem capacidade para um número limitado de teletransportes. Ele quer saber a menor distância que ele precisa andar até chegar no portal temporal, e por isso precisa da sua ajuda.

Entrada

A entrada é composta por diversos casos de teste. A primeira linha de cada caso de teste contém três inteiros T, M e C, indicando respectivamente a quantidade de vezes que a rede de teletransporte ainda pode ser usada, o número de murais em Atlântida e o número de cabines de teletransporte.

As M linhas seguintes descrevem os murais, que são todos retilíneos: cada linha contém quatro inteiros X_1, Y_1, X_2 e Y_2 indicando as coordenadas das extremidades de cada mural. Os murais têm espessura desprezível, e nenhum par de murais se intersecta, nem mesmo nas extremidades.

As C linhas seguintes descrevem a posição das cabines: cada linha contém dois inteiros X_C e Y_C indicando a posição de uma cabine de teletransporte.

Finalmente, a última linha do caso de teste contém quatro inteiros X_Q, Y_Q, X_P e Y_P indicando as coordenadas da posição inicial do Doutor Quem e do portal, respectivamente.

Saída

Para cada caso de teste na entrada o seu programa deve imprimir em uma única linha, com uma casa decimal, a distância que o Doutor Quem precisa andar para chegar em seu portal, sem contar as distâncias percorridas nos teletransportadores.

Restrições

- $0 \leq T, M, C \leq 50$
- As coordenadas de cada ponto na entrada têm valor absoluto menor ou igual a 2×10^4.
- O centro da ilha é o ponto $(0, 0)$ e seu raio é 10^5.
- As posições do Doutor Quem, do portal temporal e das cabines de teletransporte são todas distintas e não estão sobre murais.

Exemplos

Exemplo de entrada	**Exemplo de saída**
1 1 3 5 -4 5 4 1 0 5 5 9 0 0 0 10 0 1 1 3 10 -10 10 10 1 0 5 0 9 0 0 0 11 0 1 1 2 5 0 5 5 1 0 9 0 0 0 10 0	8.1 21.1 10.0