Maratona de Programação da SBC 2019

Sub-Regional Brasil do ICPC

14 de Setembro de 2019

Caderno de Problemas

Informações Gerais

Este caderno contém 13 problemas; as páginas estão numeradas de 1 a 17, não contando esta página de rosto. Verifique se o caderno está completo.

A) Sobre os nomes dos programas
1) Para soluções em C/C++ e Python, o nome do arquivo-fonte não é significativo, pode ser qualquer nome.
2) Se sua solução é em Java, ela deve ser chamada *codigo_de_problema*`.java` onde *codigo_de_problema* é a letra maiúscula que identifica o problema. Lembre que em Java o nome da classe principal deve ser igual ao nome do arquivo.
3) Se sua solução é em Kotlin, ela deve ser chamada *codigo_de_problema*`.kt` onde *codigo_de_problema* é a letra maiúscula que identifica o problema. Lembre que em Kotlin o nome da classe principal deve ser igual ao nome do arquivo.

B) Sobre a entrada
1) A entrada de seu programa deve ser lida da *entrada padrão*.
2) A entrada é composta de um único caso de teste, descrito em um número de linhas que depende do problema.
3) Quando uma linha da entrada contém vários valores, estes são separados por um único espaço em branco; a entrada não contém nenhum outro espaço em branco.
4) Cada linha, incluindo a última, contém exatamente um caractere final-de-linha.
5) O final da entrada coincide com o final do arquivo.

C) Sobre a saída
1) A saída de seu programa deve ser escrita na *saída padrão*.
2) Quando uma linha da saída contém vários valores, estes devem ser separados por um único espaço em branco; a saída não deve conter nenhum outro espaço em branco.
3) Cada linha, incluindo a última, deve conter exatamente um caractere final-de-linha.

Promoção:

Sociedade Brasileira de Computação

v1.0

Problema A

Arte Valiosa

A Mona Dura é uma das obras de arte mais valiosas do museu da Nlogônia. A famosa pintura fica em exibição num salão retangular de M por N metros. A entrada do salão fica em um canto, e a Mona fica no canto diagonalmente oposto à entrada.

Para impedir roubos, o salão dispõe de sensores de movimento, que são ativados toda noite quando o museu fecha. Cada sensor tem um valor de sensibilidade S, tal que o sensor dispara um alarme se detectar qualquer movimento a no máximo S metros de distância dele.

Um ladrão invadiu o museu esta noite com a intenção de roubar a Mona Dura. Para isso, ele precisa entrar no salão e chegar até a pintura sem ser detectado por nenhum sensor de movimento. Ou seja, ele tem que manter uma distância maior do que S_i metros do i-ésimo sensor o tempo todo, para todos os sensores.

O ladrão obteve acesso às plantas do museu, e portanto sabe as dimensões do salão e as coordenadas e sensibilidades de cada um dos sensores. Dadas essas informações, sua tarefa é determinar se o roubo é possível ou não.

Entrada

A primeira linha contém três inteiros, M, N e K, as dimensões do salão e o número de sensores de movimento, respectivamente ($10 \leq M, N \leq 10^4$, $1 \leq K \leq 1000$). A entrada do salão fica no ponto $(0, 0)$ e a pintura fica no ponto (M, N).

Cada uma das K linhas seguintes corresponde a um dos K sensores e contém três inteiros, X, Y e S, onde (X, Y) indica a localização do sensor e S indica a sua sensibilidade ($0 < X < M$, $0 < Y < N$, $0 < S \leq 10^4$). Todas as dimensões e coordenadas da entrada são em metros. É garantido que todos os sensores têm coordenadas distintas.

Saída

Seu programa deve produzir uma única linha contendo o caractere '`S`' caso seja for possível roubar a pintura, ou o caractere '`N`' caso contrário.

Exemplo de entrada 1	**Exemplo de saída 1**

`10 22 2`
`4 6 5`
`6 16 5`
	`S`

Exemplo de entrada 2	**Exemplo de saída 2**

`10 10 2`
`3 7 4`
`5 4 4`
	`N`

Exemplo de entrada 3	**Exemplo de saída 3**

`100 100 3`
`40 50 30`
`5 90 50`
`90 10 5`
	`S`

Problema B

Bobo da Corte

O Reino dos Emparelhamentos é governado por um generoso Comendador. A fama do Comendador e de suas grandes qualidades é conhecida por todos, inclusive em reinos vizinhos. Uma de suas mais famosas qualidades é seu bom humor, que é nutrido diariamente por um bobo da corte, eleito anualmente no Grande Concurso de Comédia (GCC) do reino. O bobo da corte ajuda a aliviar as tensões das diversas reuniões políticas que o cargo exige, alegrando não só o Comendador como também todo o reino.

O jovem Carlos é um grande comediante cujo sonho é se tornar bobo da corte na próxima temporada. Ele passou os últimos meses anotando piadas e trocadilhos dos mais diversos tipos, muitos dos quais sobre sua própria (diminuta) estatura. Chegou a época da eleição do bobo da corte, e um total de N candidatos se inscreveram. Cada um dos candidatos terá cinco minutos para se apresentar perante uma platéia. Após as apresentações, cada cidadão do Reino dos Emparelhamentos poderá votar em um dos candidatos, e o mais votado será o novo bobo da corte. Caso haja empate entre um ou mais candidatos, aquele que tiver feito a inscrição primeiro é eleito. Sabendo disso, o jovem Carlos passou noites na frente do escritório eleitoral e garantiu que sua inscrição fosse a primeira a ser feita.

Após a votação, resta apenas apurar os resultados. A urna eletrônica gera um relatório com N inteiros, correspondentes ao número de votos de cada candidato, ordenados pela ordem de inscrição. Sua missão é determinar se o jovem Carlos foi eleito ou não.

Entrada

A primeira linha da entrada contém um inteiro N, satisfazendo $2 \leq N \leq 10^4$. As N linhas seguintes conterão N inteiros positivos $v_1, \ldots, v_N$, um em cada linha, correspondentes ao número de votos recebido por cada um dos candidatos, em ordem de inscrição. Como a população do Reino dos Emparelhamentos é de 100.000 pessoas, o número total de votos não será superior a este valor, ou seja, $\sum_{i=1}^{N} v_i \leq 100.000$.

Saída

Seu programa deve produzir uma única linha contendo o caractere '`S`' caso o jovem Carlos seja eleito bobo da corte, ou o caractere '`N`' caso contrário.

Exemplo de entrada 1	**Exemplo de saída 1**
3 1000 1000 1000	S

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

Problema C

Cruzamento Perigoso

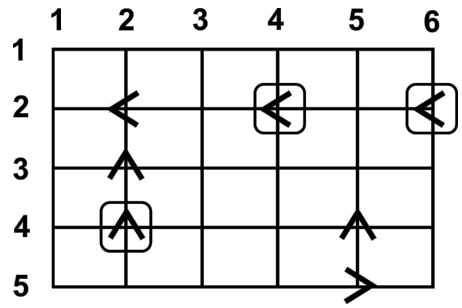
A Agência de Simulação Veicular (ASV), gerenciada pela fundadora Crishna, está trabalhando em um projeto que tem como objetivo a obtenção de dados relacionados às colisões de veículos nos cruzamentos de determinadas regiões do país.

A primeira simulação da ASV consiste em posicionar arbitrariamente C veículos em cruzamentos de uma determinada região. Inicialmente, haverá no máximo um veículo em cada cruzamento. Uma região é representada por N vias horizontais que se cruzam com M vias verticais.

Após o início da simulação, cada veículo irá se deslocar na sua direção inicial (Norte, Sul, Leste ou Oeste) com velocidade constante de 1 cruzamento por segundo.

Se dois ou mais veículos chegarem em um mesmo cruzamento ao mesmo tempo, eles irão colidir e não se movimentarão mais. Se um veículo passar por um cruzamento no qual houve uma colisão anteriormente, ele também colidirá com os veículos parados. Por motivos desconhecidos, quando dois veículos colidem horizontalmente **entre** dois cruzamentos, ambos vão parar no cruzamento ao leste, e quando dois veículos colidem verticalmente **entre** dois cruzamentos, ambos vão parar no cruzamento ao norte.

Abaixo temos um exemplo de simulação com $N = 5$, $M = 6$ e $C = 7$ veículos representados por setas indicando suas direções. Nota-se que os 3 veículos destacados irão colidir eventualmente:



Como o software de simulação da ASV ainda não é totalmente confiável, eles irão prover exemplos de configurações iniciais para que seja possível contabilizar a quantidade de veículos que nunca irão colidir.

Entrada

A primeira linha contém três inteiros N, M e C ($2 \leq N, M \leq 10^5$ e $1 \leq C \leq \min(10^5, N \times M)$), representando a quantidade de vias horizontais, a quantidade de vias verticais e a quantidade de veículos na simulação, respectivamente. Cada uma das próximas C linhas conterá dois inteiros A_i e B_i e um caractere D ($1 \leq A_i \leq N$ e $1 \leq B_i \leq M$), representando que o i-ésimo veículo está inicialmente no cruzamento da A_i-ésima via horizontal com a B_i-ésima via vertical, com direção indicada por D ('`N`', '`S`', '`L`' ou '`O`').

Saída

Seu programa deve produzir uma única linha com um inteiro representando a quantidade de veículos que não colidirão.

Exemplo de entrada 1	**Exemplo de saída 1**
5 6 7 2 2 O 3 2 N 4 2 N 4 5 N 2 6 O 5 5 L 2 4 O	4

Exemplo de entrada 2	**Exemplo de saída 2**
2 2 3 1 1 L 1 2 O 2 2 N	0

Exemplo de entrada 3	**Exemplo de saída 3**
2 2 3 1 1 L 1 2 O 2 1 N	1

Problema D

Delação Premiada

A polícia da Nlogônia está investigando a máfia local. Eles já conhecem todos os membros e a estrutura da organização: a máfia nlogoniana tem N membros no total, e cada um é identificado por um inteiro entre 1 e N, onde 1 é o ID do chefão. Além disso, todo membro é subordinado direto de um outro membro, exceto o chefão.

Mesmo após meses de investigação, a polícia ainda não tem informação suficiente para prender nenhum membro da máfia por nenhum crime. Por isso, resolveram pedir a ajuda de um vidente: dado um membro da máfia, o vidente pode magicamente adivinhar os crimes que ele cometeu, e a polícia pode então confirmá-los através de interrogatório.

Além disso, quando um mafioso nlogoniano é interrogado, ele não só admite os seus crimes, mas também delata os crimes de seu superior direto, em troca de uma pena mais leve. Se este já não tiver sido preso, a polícia pode interrogá-lo também, e ele vai então delatar o superior dele, e assim por diante, até chegarem no chefão.

Infelizmente, o vidente só tem energia suficiente para adivinhar os crimes de no máximo K mafiosos, e a polícia quer usar seus poderes cuidadosamente pra prender o máximo possível de bandidos. Dado o valor de K e a estrutura completa da máfia, qual a quantidade máxima de mafiosos que a polícia consegue prender?

Entrada

A primeira linha contém dois inteiros, N e K, onde N é o número de membros da máfia e K é o número máximo de mafiosos cujos crimes o vidente pode adivinhar ($3 \leq N \leq 10^5$, $1 \leq K < N$). A segunda linha contém $N - 1$ inteiros, onde o i-ésimo deles identifica o superior direto do mafioso de ID $i + 1$.

É garantido que todos os inteiros da segunda linha estão entre 1 e N, e que todos os membros da máfia são subordinados do chefão, direta ou indiretamente.

Saída

Seu programa deve produzir uma única linha com um inteiro representando o número máximo de mafiosos que a polícia pode prender.

Exemplo de entrada 1	**Exemplo de saída 1**
8 2 1 1 2 3 4 4 6	7

Exemplo de entrada 2	**Exemplo de saída 2**
10 3 1 1 2 2 3 3 4 4 5	8

Problema E

Exibição de Peixes

O Grande Aquário da Nlogônia recebe milhares de visitantes todo mês. Uma das suas atrações mais populares é a exibição de peixes-palhaço, um salão com vários tanques com cardumes dessa bela espécie branca e laranja. Os visitantes têm a oportunidade de aprender muitas curiosidades sobre os peixes-palhaço, incluindo sua organização social: cardumes de peixes-palhaço são liderados por fêmeas, e quando a última fêmea morre ou deixa o grupo, um dos machos restantes sofre mutação, vira uma fêmea e passa a liderar o cardume!

Assim que aprendeu isso, Zélio, o Zelador, decidiu pregar uma peça no Aquário e fazer todos os peixes-palhaço da exibição virarem fêmeas! Pra isso, ele vai mover os peixes de um tanque para o outro durante a noite, quando o Aquário está fechado. Se ao final da noite algum tanque ficar com um ou mais machos e nenhuma fêmea, no dia seguinte um deles já terá se transformado em fêmea.

Para não levantar suspeitas dos outros funcionários, Zélio só pode mover um peixe-palhaço a cada noite, e cada peixe só pode ser movido entre tanques da exibição. Cada tanque é grande o bastante para conter uma quantidade ilimitada de peixes, e Zélio pode deixar tantos tanques vazios quanto quiser. Podemos assumir que nenhuma outra pessoa irá mexer nos peixes, e que nenhum peixe vai nascer, morrer, ser adicionado ou removido dos aquários.

Zélio contou quantos peixes machos e fêmeas vivem atualmente em cada tanque da exibição. Agora ele precisa da sua ajuda pra planejar seus movimentos de forma a transformar todos os peixes-palhaço em fêmeas no menor tempo possível.

Entrada

A primeira linha contém um único inteiro N, a quantidade de tanques da exibição ($2 \leq N \leq 3000$). Cada uma das N linhas seguintes corresponde a um dos tanques e contém dois inteiros, M e F, as quantidades de peixes machos e fêmeas naquele tanque, respectivamente ($0 \leq M, F \leq 10^5$, $M = 0$ ou $F > 0$).

Saída

Seu programa deve produzir uma única linha com um inteiro representando a quantidade mínima de movimentos necessários.

Exemplo de entrada 1	**Exemplo de saída 1**
2 2 1 0 2	2

Exemplo de entrada 2	**Exemplo de saída 2**
2 2 5 1 3	7

Exemplo de entrada 3	**Exemplo de saída 3**
4 2 3 0 0 3 1 0 0	5

Problema F

Florestas em Risco

Devido ao avanço do desmatamento nas última décadas, os rios da Nlogônia registraram uma significativa redução em sua vazão. Como a Nlogônia é uma nação desenvolvida que baseia suas decisões em dados técnicos, o líder da nação encomendou uma série de estudos para compreender que medidas devem ser tomadas para garantir água para as próximas gerações.

O relatório técnico elaborado pelos cientistas envolvidos no projeto foi categórico: uma porcentagem do território do país precisa ter sua vegetação conservada. Mais do que isso, as áreas próximas das margens dos rios devem ser as mais preservadas.

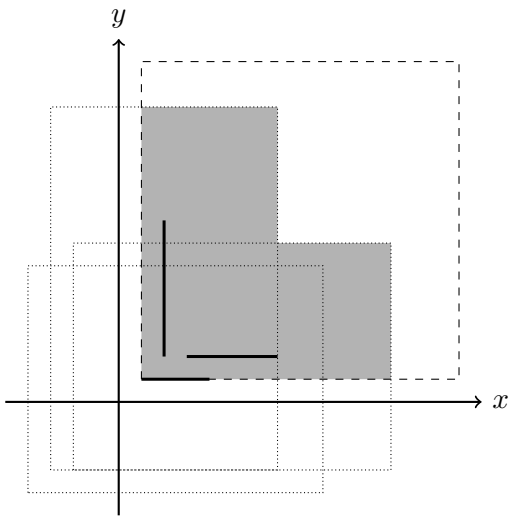
Uma nova legislação ambiental entrará em vigor, na qual áreas até certa distância das margens dos rios farão parte da área de preservação. O valor ideal dessa distância ainda é desconhecido, mas o relatório técnico já determinou o percentual do território da nação que precisa ser preservado.

Tendo em vista suas capacidades técnicas, você foi procurado para ajudar a determinar a distância ao redor dos rios que deve ser preservada, de forma a atingir o percentual necessário de área conservada.

Os rios da Nlogônia podem ser representados no plano como segmentos de reta paralelos aos eixos. Fixada uma distância r, a área do território a ser preservada é determinada da seguinte forma: Para cada rio, a área preservada ao seu redor corresponde ao menor retângulo que contém o segmento que representa o rio, respeitando uma distância mínima de r unidades entre qualquer ponto do segmento e qualquer ponto fora do retângulo. O território da Nlogônia é definido como um retângulo com lados paralelos aos eixos, de forma que todo rio é paralelo a alguma fronteira.

Dado um valor inteiro P entre 1 e 100, você deve determinar o menor valor inteiro r que garanta a preservação de $P\%$ do território da Nlogônia.

A figura abaixo ilustra o primeiro exemplo da entrada. O território da Nlogônia é representado pela região tracejada, e a área preservada é representada pela região cinza:



Entrada

A primeira linha contém um inteiro N ($1 \leq N \leq 10^4$) indicando a quantidade de segmentos de reta representando os rios da Nlogônia. Cada uma das N linhas seguintes contém 4 inteiros: x_1, y_1, x_2 e y_2, onde (x_1, y_1) e (x_2, y_2) são os extremos de um segmento de reta representando um rio. Como os rios da Nlogônia são paralelos às fronteiras, é garantido que $x_1 = x_2$ ou $y_1 = y_2$.

A próxima linha contém um inteiro P ($1 \leq P \leq 100$) indicando o percentual mínimo do território que deve ser preservado. A última linha contém 4 inteiros x_1, y_1, x_2, y_2, onde (x_1, y_1) é o canto inferior

esquerdo e (x_2, y_2) é o canto superior direito do retângulo que representa o território da Nlogônia, com lados paralelos aos eixos coordenados.

Cada coordenada descrita na entrada é um inteiro entre 0 e 10^5. Você pode assumir que todos os rios estão totalmente contidos no território da Nlogônia.

Saída

Seu programa deve produzir uma única linha com um inteiro r representando o valor mínimo que pode ser usado para garantir a preservação de $P\%$ do território da Nlogônia.

Exemplo de entrada 1	**Exemplo de saída 1**
3 1 1 4 1 2 2 2 8 3 2 7 2 50 1 1 15 15	5

Exemplo de entrada 2	**Exemplo de saída 2**
1 0 0 0 4 50 0 0 4 4	2

Problema G

Guardando Enfeites

Sicrana adora enfeites. Em casa, ela tem N enfeites que são exibidos enfileirados em uma grande prateleira. Cada enfeite é identificado por um inteiro distinto entre 1 e N.

Um dia, enquanto jogava bola dentro de casa, Fulano, filho de Sicrana, acertou a prateleira de enfeites de sua mãe, derrubando todos no chão. Mas felizmente nenhum enfeite foi danificado com a queda. Por isso, se Fulano conseguir colocar todos os enfeites de volta na prateleira exatamente como estavam antes, pode ser que sua mãe não perceba que algo de errado aconteceu.

Mas como Fulano tem uma péssima memória, ele não consegue se lembrar da ordem em que os enfeites estavam originalmente, e por isso ele precisa da sua ajuda. Para cada enfeite i, Fulano te informará N valores entre 1 e 100, onde o j-ésimo valor indica quanta confiança Fulano tem de que o enfeite i estava originalmente na posição j da prateleira. Para maximizar sua confiança de que não receberá uma bronca e ficará de castigo, ao escolher uma ordem e posicionar os enfeites, Fulano irá multiplicar as confianças de que cada enfeite esteja no lugar certo. Mais formalmente, a confiança total de Fulano em uma determinada ordem dos enfeites é calculada do seguinte modo: Se p_i é a posição ocupada pelo i-ésimo enfeite e $a(i, j)$ é a confiança de Fulano de que o i-ésimo enfeite estava originalmente na posição j, então a confiança total é dada por $\prod_{i=1}^{N} a(i, p_i)$.

Como existem muitas possibilidades diferentes para posicionar os enfeites, sua missão, caso queira aceitar, é encontrar a ordem na qual Fulano mais pode confiar.

Entrada

A primeira linha da entrada contém um inteiro N ($1 \leq N \leq 100$), indicando o número de enfeites. Cada uma das N linhas seguintes contém N inteiros entre 1 e 100. O j-ésimo inteiro na i-ésima linha indica o quanto Fulano confia que o enfeite i estava originalmente na posição j da prateleira.

Saída

Seu programa deve produzir uma única linha com N inteiros representando a ordem em que Fulano deve colocar os enfeites de modo a maximizar sua confiança total. Caso exista mais de uma ordem que resulte na confiança máxima, qualquer uma delas será aceita.

Exemplo de entrada 1	**Exemplo de saída 1**
3 1 15 37 42 8 25 77 2 1	3 1 2

Exemplo de entrada 2	**Exemplo de saída 2**
2 15 1 33 42	1 2

Problema H

Hora da Corrida

Vinicius leva muito a sério seu condicionamento físico e, diariamente às 6h da manhã, chova ou faça sol, no verão e no inverno, ele corre no entorno de uma lagoa. Ao longo da pista de corrida existem N placas igualmente espaçadas. Para não desanimar do exercício, Vinicius conta o número de placas pelas quais ele já passou e verifica se ele já correu pelo menos 10%, pelo menos 20%, ..., pelo menos 90% do percurso.

Vamos ajudar o Vinicius, calculando para ele o número de placas que ele precisa contar para ter completado pelo menos 10%, 20%, ..., 90% da corrida, dados o número de voltas que ele pretende correr e o número total de placas ao longo da pista.

Por exemplo, suponhamos que Vinicius queira dar 3 voltas e o número de placas seja 17. Então, para garantir ter corrido pelo menos 30% do percurso, ele precisa contar 16 placas. Para garantir pelo menos 60%, ele precisa contar 31 placas.

Entrada

A entrada consiste de uma única linha que contém dois inteiros, V e N ($1 \leq V, N \leq 10^4$), onde V é o número pretendido de voltas e N é o número de placas na pista.

Saída

Seu programa deve produzir uma única linha com nove inteiros representando os números de placas que devem ser contadas para garantir o cumprimento, respectivamente, de 10%, 20%, ..., 90% da meta.

Exemplo de entrada 1	**Exemplo de saída 1**
`3 17`	`6 11 16 21 26 31 36 41 46`

Exemplo de entrada 2	**Exemplo de saída 2**
`5 17`	`9 17 26 34 43 51 60 68 77`

Exemplo de entrada 3	**Exemplo de saída 3**
`3 11`	`4 7 10 14 17 20 24 27 30`

Problema I

Interplanetário

Estamos no ano de 2306 e, com o avanço da nanotecnologia, viagens interplanetárias estão cada vez mais acessíveis. Bibika trabalha na maior agência de viagem interplanetária do universo e recebe clientes interessados diariamente.

Os clientes de Bibika são exigentes e fazem várias demandas antes de fechar o roteiro de suas viagens, como minimizar a distância total percorrida. Mas as maiores restrições são com relação às temperaturas dos planetas visitados no percurso (excluindo os planetas de origem e de destino). A temperatura de um planeta, medida em graus Anidos, pode variar de 10^9 graus Anidos negativos até 10^9 graus Anidos positivos. Os clientes de Bibika são oriundos de planetas de climas variados e, consequentemente, possuem preferências diferentes em relação a temperatura: alguns se incomodam com planetas muito frios e outros com planetas muito quentes. Bibika precisa planejar a rota das viagens de forma a poupar seus clientes de qualquer desconforto, mesmo que para isso o comprimento total da rota não seja o menor possível (ou até mesmo que não exista uma rota: nesse caso Bibika simplesmente informa os clientes de que a viagem é impossível).

Bibika lhe forneceu a temperatura média histórica de cada um dos N planetas e as R rotas que ligam pares de planetas diretamente (é garantido que entre dois planetas existe no máximo uma rota direta), juntamente com suas respectivas distâncias. Ela lhe fornecerá também os pedidos de viagem de Q clientes. Cada pedido consiste de um planeta de origem A, um planeta de destino B, e a restrição do cliente em relação às temperaturas dos planetas intermediários: cada cliente pode exigir passar apenas por planetas com temperaturas entre as K menores ou K maiores dentre todos os N planetas.

Sua tarefa é, para cada pedido de viagem, encontrar a menor distância percorrida possível dadas as restrições descritas, ou dizer que a viagem é impossível.

Entrada

A primeira linha contém dois inteiros N e R ($2 \leq N \leq 400$ e $0 \leq R \leq N \cdot (N-1)/2$), representando a quantidade de planetas conhecidos e a quantidade de rotas diretas entre eles. O primeiro planeta é representado pelo número 1, o segundo pelo número 2, ..., até o N-ésimo pelo número N. A segunda linha contém N inteiros T_i ($-10^9 \leq T_i \leq 10^9$), representando a temperatura média de cada um dos planetas. A seguir haverá R linhas, cada uma contendo três inteiros X, Y e D ($1 \leq X, Y \leq N$ com $X \neq Y$ e $1 \leq D \leq 10^3$), representando uma rota direta de comprimento D entre os planetas X e Y. Em seguida haverá um inteiro Q ($1 \leq Q \leq 10^5$), representando a quantidade de pedidos de viagens dos clientes. Por fim, cada uma das próximas Q linhas conterá quatro inteiros A, B, K e T ($1 \leq A, B, K \leq N$ com $A \neq B$ e $T \in \{0, 1\}$), representando um cliente que deseja ir do planeta A para o planeta B passando apenas por planetas que estejam entre os K mais frios se $T = 0$ ou K mais quentes se $T = 1$.

Saída

Seu programa deve produzir uma linha para cada cliente contendo um inteiro que representa a menor distância total de viagem entre os dois planetas dadas as restrições do cliente, ou -1 caso a viagem não seja possível.

Exemplo de entrada 1	**Exemplo de saída 1**
7 9 -53 -180 456 420 -210 15 150 1 2 2 1 3 1 2 3 4 2 4 2 2 5 5 3 4 6 6 4 10 4 5 4 3 7 2 4 1 5 2 1 1 2 1 1 5 6 1 0 1 7 2 1	11 2 -1 3

Exemplo de entrada 2	**Exemplo de saída 2**
6 5 5 10 20 10 10 8 1 2 5 2 3 5 3 4 5 4 5 5 5 6 5 4 1 6 2 1 1 6 1 1 4 5 1 0 2 4 1 1	25 -1 5 10

Problema J

Jogo de Baralho

O cronograma do dia das competições de programação normalmente segue o mesmo padrão: aquecimento de manhã, seguido do horário de almoço, um tempo de descanso, ajustes finais do ambiente de competição e então o início da prova.

No tempo de descanso, alguns competidores preferem relaxar, outros preferem socializar e uma parte tem o costume de jogar algum jogo de baralho. Luciano e seus amigos gostam de jogar um jogo conhecido como "Copo d'Água". Cansado de não ser o vencedor, Luciano quer escrever um programa que, dadas as cartas iniciais de todos os jogadores (não me pergunte como ele sabe disso), determine se ele irá vencer ou não. Se ele não for vencer, ele pode então inventar uma desculpa qualquer e pedir para não participar daquela rodada.

O jogo funciona da seguinte maneira:

- O baralho utilizado possui as cartas: "A23456789DQJK" (nessa ordem, de menor para maior valor), onde os naipes são ignorados. Além disso, o baralho possui mais uma única carta extra: o curinga.
- N competidores sentam lado a lado em círculo. O competidor 1 está imediatamente à esquerda do 2, que está imediatamente à esquerda do 3, e assim por diante até completar o círculo com o N-ésimo competidor imediatamente à esquerda do 1. Um competidor K é sorteado para iniciar o jogo.
- Em um jogo com N competidores, existirão quatro cartas de N diferentes valores e um curinga. No começo do jogo, o competidor K recebe o curinga; as demais cartas são embaralhadas e distribuídas entre os jogadores, de modo que cada jogador receba quatro delas.
- Em cada rodada, o jogador da vez escolhe uma de suas cartas e a passa para o jogador à sua direita. O jogador que recebeu uma carta será o próximo jogador da vez.
- Dizemos que um jogador está em estado vencedor se possuir exatamente quatro cartas em mãos e elas forem todas iguais. O jogo termina assim que ao menos um competidor estiver em estado vencedor. Nesse caso, o competidor de menor índice em estado vencedor será declarado o jogador vencedor.

A carta que será passada de um competidor para o próximo é definida pela seguinte regra:

- O curinga nunca pode ser passado logo depois de ser recebido. Isso também se aplica ao jogador inicial, que recebeu o curinga do distribuidor de cartas logo antes da primeira rodada.
- O competidor irá, sempre que possível, passar o curinga para o próximo.
- Caso não passe o curinga, o competidor irá escolher a carta que menos aparece em sua mão e passar para o próximo. Caso exista mais de uma carta que aparece uma menor quantidade de vezes, ele irá passar, dentre essas, a carta de menor valor de acordo com a ordem descrita anteriormente.

Sabendo das regras, ajude Luciano escrevendo um programa que, dada a configuração inicial do jogo, diga qual jogador será declarado vencedor.

Entrada

A primeira linha contém dois inteiros N e K ($2 \leq N \leq 13$ e $1 \leq K \leq N$) representando, respectivamente, a quantidade de competidores e o competidor que iniciará o jogo. Cada uma das próximas N linhas conterá quatro caracteres, representando as cartas iniciais do i-ésimo competidor (com exceção do curinga).

Saída

Seu programa deve produzir uma única linha com um inteiro representando o competidor que será declarado vencedor.

Exemplo de entrada 1	**Exemplo de saída 1**
2 1 33J3 JJJ3	2

Exemplo de entrada 2	**Exemplo de saída 2**
2 2 A2A2 22AA	2

Exemplo de entrada 3	**Exemplo de saída 3**
4 2 774Q JJQ7 44Q7 4QJJ	3

Exemplo de entrada 4	**Exemplo de saída 4**
3 1 JQAA JJJA QQQA	3

Problema K

Keep Calm e Venda Balões

Walter vende balões de porta em porta. Todo dia ele escolhe uma rua da sua cidade e visita todas as casas nela, oferecendo seus coloridos balões.

Cada rua da cidade de Walter tem a mesma quantidade de casas dos dois lados, e todas as casas da cidade são do mesmo tamanho. Dessa forma, cada rua pode ser vista como uma matriz $2 \times N$, onde cada célula é uma casa, e N é a quantidade de casas ao longo de cada lado da rua.

Depois de escolher a rua do dia, Walter visita cada casa dessa rua exatamente uma vez. Ele pode começar seu caminho em qualquer casa, mas só pode se mover entre casas adjacentes horizontalmente, verticalmente ou diagonalmente.

1	2	3	4	5	6
7	8	9	10	11	12

A tabela acima ilustra um exemplo para $N = 6$. Após visitar a casa de número 1, Walter só poderia seguir imediatamente para as casas de número 2, 7 e 8 (isto é, se ele já não tiver visitado elas antes). E após visitar a casa de número 11, a próxima casa do caminho só poderia ser uma das seguintes: 4, 5, 6, 10 ou 12.

Hoje, antes de sair de casa, Walter olhou o mapa da cidade para contar a quantidade N de casas de cada lado da rua escolhida. Agora ele quer saber de quantas maneiras distintas ele pode visitar todas as $2N$ casas da rua, seguindo as regras descritas. Duas maneiras de visitar as casas são diferentes se e somente se a ordem das casas varia: isto é, se existem duas casas A e B tais que A é visitada antes de B em uma ordem e B é visitada antes de A na outra.

Entrada

A entrada consiste de uma única linha que contém um inteiro N ($1 \leq N \leq 10^9$).

Saída

Seu programa deve produzir uma única linha com um inteiro representando o número de maneiras possíveis de visitar todas as casas da rua. Dado que este número pode ser muito grande, você deve fornecer o resto da divisão deste número por $10^9 + 7$.

Exemplo de entrada 1	**Exemplo de saída 1**
2	24

Exemplo de entrada 2	**Exemplo de saída 2**
3	96

Exemplo de entrada 3	**Exemplo de saída 3**
4	416

Exemplo de entrada 4	**Exemplo de saída 4**
61728	654783381

Problema L

Lançando Moedas

Carla e Daniel decidiram jogar cara-ou-coroa para decidir quem vai lavar os pratos hoje. Eles vão jogar com uma das moedas antigas da coleção de Carla. Isso deixa Daniel preocupado, pois essas moedas são tortas e desbalanceadas: no lançamento de uma moeda, as probabilidades da obtenção de cara e de coroa não são necessariamente iguais.

Carla conhece bem suas moedas, e pode escolher uma que maximize suas chances de vencer. Por isso, Daniel inventou um esquema para fazer com que o sorteio seja completamente justo, independentemente da moeda escolhida. Primeiro, a cada um deles será atribuído um conjunto não-vazio de cadeias binárias de tamanho N. Nenhuma cadeia pode pertencer a ambos, e algumas cadeias podem não ser incluídas no conjunto de nenhum dos dois. Por exemplo, para $N = 3$, uma forma válida de dividir as cadeias seria:

- "010" e "110" para Carla;
- "001" e "011" para Daniel;
- "000", "100", "101" e "111" para nenhum dos dois.

Após a divisão das cadeias, Carla e Daniel vão jogar a mesma moeda N vezes e anotar a sequência de resultados, onde cada cara equivale a um 0 e cada coroa equivale a um 1. Se a cadeia binária resultante pertencer ao conjunto de Carla, ela é a vencedora. Se pertencer ao conjunto de Daniel, ele é o vencedor. Se a cadeia não pertencer a nenhum dos dois, a moeda é jogada mais N vezes para gerar uma nova cadeia. O processo é repetido tantas vezes quanto necessário, até conseguirem um vencedor.

O justo funcionamento desse esquema depende da repartição das cadeias entre Carla e Daniel: é preciso que a probabilidade de gerar uma cadeia do conjunto de Carla seja igual à probabilidade de gerar uma cadeia do conjunto de Daniel. Em outras palavras, seja $P(S)$ a probabilidade de que uma cadeia binária S de comprimento N seja gerada por uma sequência de N lançamentos de uma mesma moeda, possivelmente desbalanceada. O total de P para todas as cadeias do conjunto de Carla deve ser o mesmo que o total de P para todas as cadeias do conjunto de Daniel.

Além de repartir as cadeias de forma justa, Carla e Daniel querem evitar ao máximo ter que repetir os lançamentos da moeda, e por isso querem minimizar a quantidade de cadeias que não pertençam a nenhum dos dois. Dado o valor de N, determine o menor número possível de cadeias não atribuídas.

Entrada

A entrada consiste de uma única linha que contém contém um inteiro N, o número de lançamentos da moeda e o comprimento das cadeias binárias ($2 \leq N \leq 10^{18}$).

Saída

Seu programa deve produzir uma única linha com um inteiro representando o número mínimo de cadeias não utilizadas na divisão.

Exemplo de entrada 1	**Exemplo de saída 1**
3	4

Exemplo de entrada 2	**Exemplo de saída 2**
5	4

Exemplo de entrada 3	**Exemplo de saída 3**
8	2

Problema M

Maratona Brasileira de Comedores de pipocas

A Maratona Brasileira de Comedores de pipocas é uma competição que ocorre anualmente com o intuito de descobrir qual a equipe mais organizada, preparada e bem-treinada na arte de comer pipoca. Ela é organizada pela SBCp (Sociedade Brasileira de Comedores de pipocas), que periodicamente se reúne para discutir as regras e o formato da competição.

A competição consiste em N sacos de pipocas colocados lado a lado, onde cada saco possui uma quantidade arbitrária de pipoca. Para proporcionar uma maior diversão, a competição ocorre em equipes, cada uma composta por C competidores. Como a Maratona Brasileira de Comedores de pipocas é um evento sério que preza, além de tudo, pela saúde dos competidores, a comissão médica impôs que cada competidor poderá comer, no máximo, T pipocas por segundo, a fim de evitar um possível mal-estar.

A SBCp, em sua última reunião, definiu duas novas regras para a edição de 2019:

- Cada competidor da equipe deverá comer uma sequência contígua de sacos de pipoca. É perfeitamente válido que um competidor não coma nenhuma pipoca.
- Todas as pipocas de um mesmo saco devem ser comidas por um único competidor.

O objetivo da competição é comer todas as pipocas no menor tempo possível, dado que os C competidores podem comer em paralelo e eles respeitarão todas as regras impostas pela SBCp.

Entrada

A primeira linha contém três inteiros N, C e T ($1 \leq N \leq 10^5$, $1 \leq C \leq 10^5$ e $1 \leq T \leq 50$), representando a quantidade de sacos de pipoca, a quantidade de competidores de uma mesma equipe e quantidade máxima de pipoca por segundo que um competidor pode comer. A segunda linha conterá N inteiros P_i ($1 \leq P_i \leq 10^4$), sendo estes a quantidade de pipoca em cada um dos N sacos.

Saída

Seu programa deve produzir uma única linha com um inteiro representando a quantidade mínima de segundos necessária para a equipe comer todas as pipocas se ela se organizar da melhor maneira possível.

Exemplo de entrada 1	**Exemplo de saída 1**
5 3 4 5 8 3 10 7	4

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

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