Maratona de Programação da SBC 2021

Sub-Regional Brasil do ICPC

30 de Outubro de 2021

Caderno de Problemas

Informações Gerais

Este caderno contém 14 problemas; as páginas estão numeradas de 1 a 25, não contando esta página de rosto. Verifique se o caderno está completo.

Este conjunto de problemas também está sendo utilizado simultaneamente nas seguintes competições: Tercera Fecha Gran Premio de México 2021, Tercera Fecha Gran Premio de Centroamérica 2021 e Torneo Argentino de Programación 2021.

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

Alocação de Prêmios

Uma competição de programação se dará na Nlogônia, para determinar quem é o melhor programador Nlogoniano de todos os tempos.

A competição terá N participantes e não há empates, ou seja, todo participante terminará colocado em uma posição de 1 a N, e todas as posições são distintas. Menores posições correspondem a melhores resultados.

Os organizadores do evento decidiram que cada participante receberá um prêmio de no máximo R pontos e, para ser justo com os participantes com melhor desempenho, um participante nunca receberá menos pontos que um outro com pior colocação.

Alguns participantes, entretanto, são gananciosos e querem receber mais pontos para serem felizes. Um competidor na colocação i precisa receber um prêmio de pelo menos p_i pontos para ser feliz.

Ina, uma organizadora muito curiosa, está se perguntando de quantas maneiras é possível distribuir prêmios aos participantes de maneira a satisfazer as condições da organização, e fazer todos os participantes felizes. Como esse número é muito grande, você deve calculá-lo módulo $10^9 + 7$.

Duas maneiras são diferentes se pelo menos um competidor recebe um prêmio diferente.

Entrada

A primeira linha contém dois inteiros N e R ($1 \leq N \leq 5000$, $1 \leq R \leq 10^9$), representando o número de participantes e o prêmio máximo que cada competidor pode receber, respectivamente.

A segunda linha contém N inteiros, p_i ($1 \leq p_i \leq 10^9$), representando a menor quantidade de pontos que o participante na colocação i precisa receber para ficar feliz.

Saída

Imprima uma linha contendo o número de diferentes maneiras de se distribuir o prêmio, módulo $10^9 + 7$.

Exemplo de entrada 1	**Exemplo de saída 1**
2 5 4 1	9

Exemplo de entrada 2	**Exemplo de saída 2**
3 10 7 1 10	1

Problema B

Belas Palavras

São dados uma string A de comprimento N e um conjunto S contendo M strings.

Uma permutação cíclica B_i de A, onde i é um número entre 1 e N, é a string

$$B_i = A_i A_{i+1} \cdots A_{N-1} A_N A_1 A_2 \cdots A_{i-2} A_{i-1}$$

e a sua pontuação é definida como o maior comprimento de uma substring de B_i que é também uma substring de uma string em S.

Uma substring é definida como uma sequência contígua de letras. Por exemplo, `ab` e `dc` são substrings de `abfdc`, mas `ad` e `fc` não são substrings de `abfdc`.

Sua tarefa é calcular a menor pontuação dentre todas as permutações cíclicas da string A.

Entrada

A primeira linha contém dois inteiros positivos N e M, ($1 \leq N \leq 10^5$, $1 \leq M \leq 10^4$), representando o comprimento da string A e o tamanho do conjunto S, respectivamente.

A segunda linha contém a string A.

Cada uma das M linhas seguintes contém uma string s_i, representando a i-ésima string em S.

Todas as strings contêm apenas letras minúsculas do alfabeto, e é garantido que a soma dos tamanhos de todas as strings em S nunca ultrapassa 10^5 caracteres.

Saída

Imprima uma linha contendo um inteiro representando a menor pontuação dentre todas as permutações cíclicas da string A.

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

`7 3`
`acmicpc`
`acm`
`icpc`
`maratona`
	`3`

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

`11 4`
`competition`
`oncom`
`petition`
`ztxvu`
`fmwper`
	`5`

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

`12 4`
`latinamerica`
`zyvu`
`okp`
`wsgh`
`kqpdb`
	`0`

Problema C

Criando Multiplos

Malba é um garoto muito inteligente que gosta de calcular. Já ganhou muitas competições, inclusive a prestigiosa competição Tahan, em que conseguiu o primeiro lugar, representando o seu país, a Logônia.

Ele inventou um problema, no qual ele considera um número N, escrito numa certa base B, e representado por L algarismos. O objetivo do jogo é reduzir não mais do que um dos algarismos de forma que o novo número, M, seja um múltiplo do número $B + 1$. Mas há um detalhe: dentre as alterações possíveis, deve-se escolher uma que minimize M.

Por exemplo, suponha que $B = 10$ e $N = 23456$. Há duas maneiras de obter M: ou reduzimos o algarismo 4 para 0 ou reduzimos o algarismo 6 para 2. Então, o 4 deve ser reduzido para 0, portanto $M = 23056$. Em alguns casos não há solução, como no caso em que $B = 10$ e $N = 102$. Nesse caso, se trocarmos o algarismo 1 por 9 obteremos um múltiplo de 11, mas não podemos aumentar o valor de um algarismo!

Observe que pode ser necessário reduzir o primeiro algarismo para o valor 0. Por exemplo, isto acontece se $B = 10$ e $N = 322$.

Você consegue dizer qual dígito deve ser reduzido e qual seu novo valor?

Entrada

A primeira linha contém dois inteiros B e L ($2 \leq B \leq 10^4$, $1 \leq L \leq 2 \times 10^5$), representando a base e o número de algarismos do número N, respectivamente.

A segunda linha contém L inteiros $D_1, D_2, \ldots, D_L$ ($0 \leq D_i < B$ para $i = 1, 2, \ldots, L$), representando os algarismos do número N. O primeiro algarismo, D_1, é o mais significativo e o último algarismo, D_L, é o menos significativo.

Saída

Imprima uma linha contendo dois inteiros, separados por um espaço. O primeiro inteiro é o índice do algarismo a ser alterado (lembre que o índice do primeiro algarismo, D_1, é 1 e o índice do último algarismo, D_L, é L). O segundo inteiro é o novo valor do algarismo. Se não houver solução para o problema, imprima `-1 -1`. Se N já for um múltiplo de $B + 1$, imprima `0 0`.

Exemplo de entrada 1	**Exemplo de saída 1**
`10 5` `2 3 4 5 6`	`3 0`

Exemplo de entrada 2	**Exemplo de saída 2**
`10 3` `1 0 2`	`-1 -1`

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

Exemplo de entrada 4	**Exemplo de saída 4**
`17 5` `3 0 0 0 0`	`1 0`

Exemplo de entrada 5	**Exemplo de saída 5**
16 4 15 0 13 10	1 14

Exemplo de entrada 6	**Exemplo de saída 6**
16 5 1 15 0 13 10	0 0

Problema D

Dividindo o Reino

O reino da Nlogônia historicamente tem sido um lugar rico e tranquilo. Entretanto, as atuais circunstâncias podem dar fim a esta era de paz e prosperidade: o rei é pai de gêmeos, então ambos são herdeiros do trono.

Os gêmeos não se dão muito bem e são ciumentos e competitivos um com o outro. Devido a este fato, tê-los governando o reino cooperativamente não é realmente uma opção. O reino terá de ser dividido em dois principados independentes, de forma que cada um seja dado a um príncipe. Além disso, a divisão precisa ser totalmente justa, para evitar conflitos entre os irmão invejosos.

O reino consiste em N cidades e M estradas conectando pares de cidades. Os Nlogonianos são particularmente orgulhosos de suas estradas. Cada estrada tem um valor positivo associado a ela, representando sua beleza.

O reino será dividido da seguinte maneira: primeiro, as cidades serão particionadas em dois conjuntos de forma que cada cidade está em exatamente um conjunto. Então, cada principado será composto pelas cidades em um dos conjuntos e pelas estradas conectando as cidades naquele conjunto. Estradas que conectam cidades de diferentes principados serão destruídas, uma vez que os príncipes não estão interessados em fazer negócios ou colaborações entre os principados, e manter estradas apenas aumentaria a chance de guerras.

A beleza de um principado é definida como a maior beleza dentre todas as estradas daquele principado, ou 0 (zero) se o principado não possui estradas. Por motivos óbvios, o rei gostaria que a beleza de ambos os principados fosse igual.

Ajude o rei a determinar todos os possíveis valores de beleza dos possíveis principados resultantes, dado que a divisão é feita de maneira que os principados sejam igualmente belos.

Entrada

A primeira linha contém dois inteiros N, M ($1 \leq N, M \leq 5 \times 10^5$), representando o número de cidades e o número de estradas respectivamente.

Cada uma das M linhas seguintes contém três inteiros x_i, y_i, b_i ($1 \leq x_i < y_i \leq N$, $1 \leq b_i \leq 10^9$), representando que existe uma estrada que conecta as cidades x_i e y_i e possui beleza b_i. Não existem estradas entre um mesmo par de cidades.

Saída

Se não for possível dividir o reino de maneira que ambos os principados tenham a mesma beleza, imprima uma linha com a string "`IMPOSSIBLE`". Caso contrário, imprima todos os possíveis valores resultantes da divisão do reino em principados de mesma beleza. Os valores devem ser impressos em ordem crescente, um por linha.

Exemplo de entrada 1	**Exemplo de saída 1**
9 7 1 2 3 2 3 3 3 4 3 1 3 2 2 4 2 6 7 1 8 9 1	2 3

Exemplo de entrada 2	**Exemplo de saída 2**
4 4 1 2 5 2 3 6 1 3 7 3 4 7	IMPOSSIBLE

Exemplo de entrada 3	**Exemplo de saída 3**
2 1 1 2 10	0

Problema E

Escada Rolante

Você acaba de inventar um novo tipo de escada rolante: uma escada rolante dupla. Escadas rolantes normais levam as pessoas de uma das pontas para a outra, mas não na direção contrária, enquanto que as escadas rolantes duplas podem levar pessoas de qualquer uma das pontas para a outra.

Leva-se 10 segundos para que a escada rolante dupla leve uma pessoa de uma das pontas até a outra. Isto é, se a pessoa entra na escada rolante dupla em uma das pontas no momento T, então vai sair na outra ponta no momento $T + 10$ – esta pessoa não estará mais na escada rolante dupla no momento $T + 10$.

A todo momento que ninguém esteja usando a escada rolante dupla, ela estará parada. Portanto, inicialmente ela está parada.

Quando a escada rolante dupla está parada e uma pessoa entra por uma das pontas, a escada rolante dupla se ligará automaticamente e se moverá na direção que aquela pessoa quer ir.

Se uma pessoa chegar na escada rolante dupla e esta já estiver movendo-se na direção que a pessoa quer ir, então a pessoa entrará nela imediatamente. Caso contrário, se a escada rolante estiver se movendo na direção oposta, a pessoa terá que esperar até que a escada rolante pare e só então a pessoa poderá entrar nela. A escada rolante dupla é tão larga que ela pode acomodar inúmeras pessoas entrando nela ao mesmo tempo.

A escada rolante dupla tem um efeito bem estranho, provavelmente relacionado a alguma propriedade da física quântica (ou simplesmente ao acaso): nenhuma pessoa vai chegar na escada rolante dupla no momento exato em que ela está prestes a parar.

Agora que você sabe como a escada rolante dupla funciona, você terá a tarefa de simulá-la. Dada a informação de N pessoas, incluindo o momento em que elas chegaram na escada rolante dupla e em qual direção elas querem andar, você tem que descobrir qual o último momento em que a escada para.

Entrada

A primeira linha contém um inteiro N ($1 \leq N \leq 10^4$), representando quantas pessoas usarão a escada rolante.

Em seguida haverão N linhas contendo dois inteiros t_i e d_i cada ($1 \leq t_i \leq 10^5$, $0 \leq d_i \leq 1$), representando o momento em que a i-ésima pessoa chegará na escada rolante dupla e em qual direção ela quer ir. Se d_i é igual a 0, então a pessoa quer ir da ponta esquerda para a ponta direita, e se d_i é igual 1, então a pessoa quer ir da ponta direita para a ponta esquerda. Todos os valores de t_i são distintos e dados em ordem crescente.

Saída

Imprima uma linha contendo o momento no qual a última pessoa saiu da escada rolante.

Exemplo de entrada 1	**Exemplo de saída 1**
3 5 0 8 0 13 0	23

Exemplo de entrada 2	**Exemplo de saída 2**
3 5 0 7 1 9 0	29

Exemplo de entrada 3	**Exemplo de saída 3**
3 5 0 10 1 16 0	35

Problema F

Fuga da Prisão

Michael e seu irmão Lincoln foram presos injustamente, na mesma prisão, mas Michael tem um plano para resgatar seu irmão. Pode-se considerar que a prisão é um conjunto de polígonos convexos no plano, cujas arestas são muros. Os muros de polígonos distintos não se interceptam, mas os polígonos podem ser encaixados, isto é, um polígono pode estar dentro de outro polígono. Pode-se considerar que Michael e Lincoln são dois pontos no plano. O caminho para o resgate consiste em primeiro Michael chegar até seu irmão e então ambos precisam escapar da prisão.

Eles não têm problema para andar, mas escalar muros é perigoso e difícil, assim Michael tentará minimizar o número total de muros que ele deverá escalar. Portanto, Michael primeiro precisa escalar alguns muros para chegar ao seu irmão, caso não se encontrem na mesma área. Em seguida deve escalar mais alguns muros para deixar a prisão. Deixar a prisão significa não estar dentro de quaisquer muros, que pode-se considerar equivalente a atingir um ponto muito longe, digamos, $(10^{20}, 10^{20})$. Brad está encarregado da alocação dos prisioneiros e está sabendo do plano, de forma que ele colocará os dois prisioneiros em dois pontos diferentes do plano, mas não nos muros, e de tal forma que o número mínimo de muros que precisam ser escalados por Michael seja o maior possível. Qual é o valor do número mínimo de muros que Michael precisará escalar se Brad colocar os dois irmãos de forma ótima?

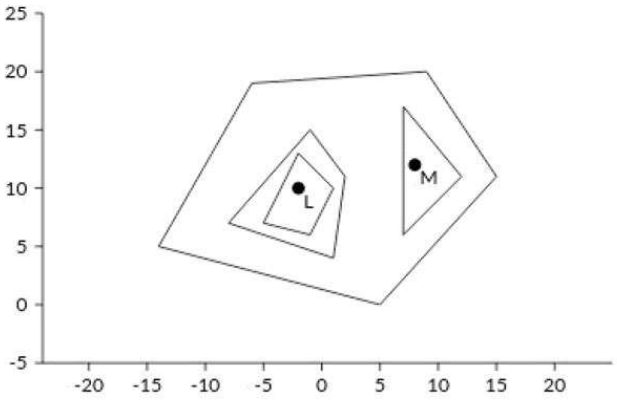

Figura 1: Ilustração de possível solução para exemplo 1. O ponto M representa Michael e o ponto L Lincoln

Entrada

A primeira linha da entrada contém um inteiro N ($1 \leq N \leq 2 \times 10^5$), que é o número de polígonos convexos. Esta linha é seguida pelas descrições de cada polígono. A primeira linha da i-ésima descrição contém um inteiro k_i ($3 \leq k_i \leq 6 \times 10^5$), seguida de k_i linhas, cada uma das quais contém um ponto (x_j, y_j) ($-10^9 \leq x_j, y_j \leq 10^9$).

Os pontos para formar cada polígono convexo são dados na ordem anti horária e não há três pontos consecutivos colineares. As arestas de dois polígonos distintos não se interceptam. O número total de arestas não passa de 6×10^5, ou seja, $\sum_{i=1}^{N} k_i \leq 6 \times 10^5$.

Saída

Imprima um inteiro, o número mínimo de muros que precisarão ser escalados por Michael para resgatar seu irmão, supondo que Brad os colocou em lugares que torna esse número de muros o maior possível.

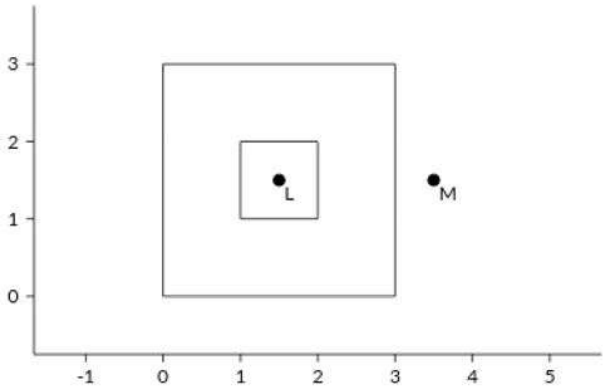

Figura 2: Ilustração de possível solução para exemplo 2

Exemplo de entrada 1	**Exemplo de saída 1**
4 4 1 10 -2 13 -5 7 -1 6 5 15 11 9 20 -6 19 -14 5 5 0 4 -1 15 -8 7 1 4 2 11 3 7 17 7 6 12 11	6

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

Problema G

Garantindo o Treino

Juan decidiu começar a se exercitar e está começando a preparar uma sessão de treinamento.

Ele sabe que em alguns dias pode não querer fazer todos os exercícios de sua sessão. Ele então decidiu criar algumas regras para evitar pular a sessão inteira e acabar não se exercitando, mas de forma que ainda possa evitar alguns exercícios de vez em quando.

As regras são:

- Haverá dois tipos de exercícios: A e B.
- Após terminar um exercício do tipo B ele fará o próximo exercício, se houver algum. Caso contrário, a sessão termina.
- Após terminar um exercício do tipo A, há duas possibilidades: ele pode iniciar o exercício seguinte ou pulá-lo, executando o exercício imediatamente posterior.
- O último exercício da sessão será sempre do tipo B.

Assim, podem existir diferentes maneiras de se completar a sessão de exercícios. Por exemplo, se os tipos dos exercícios são `BAAB`, existem 3 maneiras de se completar a sessão: fazendo todos os exercícios, pulando o terceiro ou pulando o último.

Juan quer preparar sua sessão de exercícios de maneira que existam exatamente N maneiras diferentes de completá-la. Você pode ajudá-lo?

Entrada

A entrada é composta por uma única linha contendo um inteiro N ($2 \leq N \leq 10^{15}$), representando o número de maneiras que a sessão de exercícios pode ser completada.

Saída

Imprima uma linha, formada pelos caracteres '`A`' e '`B`', representando os tipos dos exercícios na sessão. Se houver múltiplas respostas, imprima aquela que é lexicograficamente menor. Se não houver sessões válidas, imprima uma linha contendo a string "`IMPOSSIBLE`" (sem aspas).

Exemplo de entrada 1	**Exemplo de saída 1**
2	AB

Exemplo de entrada 2	**Exemplo de saída 2**
4	ABAB

Exemplo de entrada 3	**Exemplo de saída 3**
7	IMPOSSIBLE

Problema H

Haja Ordenação

Um amigo seu inventou um jogo e quer saber se você consegue resolvê-lo ou se ele é impossível.

Ele montou uma sequencia de N blocos. Cada bloco tem um número gravado e uma cor. Todos os números são números distintos entre 1 e N, e blocos diferentes podem ter a mesma cor.

O jogo funciona da seguinte maneira: você pode jogar quantos turnos você quiser. Em um turno, você escolhe dois blocos diferentes que têm a mesma cor e os troca de posição.

Você deve dizer se é possível fazer com que a sequência inteira fique em ordem crescente ou não.

Entrada

A primeira linha contém dois inteiros N e K ($1 \leq N \leq 10^5$, $1 \leq K \leq N$), representando o número de blocos na sequência e o número de cores diferentes, respectivamente.

Cada uma das N linhas seguintes contém dois inteiros n_i e c_i ($1 \leq n_i \leq N$, $1 \leq c_i \leq K$), representando o número e a cor do i-ésimo bloco, respectivamente.

Saída

Imprima uma linha contendo um caracter. Se a sequência puder ser ordenada em ordem crescente, imprima a letra maiúscula '`Y`'; caso contrário, imprima a letra maiúscula '`N`'.

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

`4 2`
`3 1`
`4 2`
`1 1`
`2 2`
	`Y`

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

`4 2`
`2 1`
`4 2`
`1 1`
`3 2`
	`N`

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

`3 1`
`1 1`
`2 1`
`3 1`
	`Y`

Problema I

Invertendo Ferrovias

O governo de Nlogônia está incomodado com a falta de eficiência de seu sistema ferroviário. Todo par de cidades possui uma única ferrovia que as liga, porém, devido a problemas monetários, algumas delas estão inativas.

Uma configuração ideal da malha ferroviária é tal que, para qualquer par de cidades, existe um único caminho ligando essas duas cidades usando somente ferrovias ativas.

O governador de Nlogônia te contratou para transformar seu conjunto de ferrovias em uma configuração ideal. Infelizmente, você não fala Nlogoniano e nem tem controle sobre as ferrovias: somente os líderes de cada cidade, que só falam Nlogoniano, podem ativar ou desativar as ferrovias.

Nlogoniano possui algumas expressões extremamente específicas. Para tentar te ajudar, e te desafiar ao mesmo tempo, um amigo te ensinou uma frase que você pode tentar usar: "`lupDujHomwIj luteb gharghmey`". Ele disse que, ao falar isso para o líder de uma cidade, tal líder ativará todas as ferrovias adjacentes a essa cidade que estavam previamente desativadas e desativará todas as ferrovias adjacentes a essa cidade que estavam previamente ativas. Em outras palavras, o status de "ativação" de todas as ferrovias adjacentes à cidade em questão será trocado.

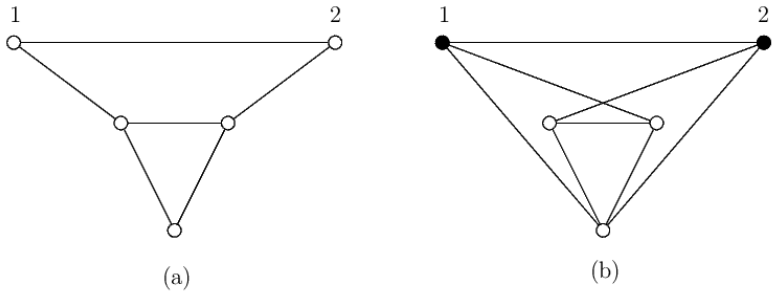

Figura 3: Resultado de "`lupDujHomwIj luteb gharghmey`" cidades 1 e 2

Sabendo essa frase, você pode ligar para alguns líderes e fazê-los "`lupDujHomwIj luteb gharghmey`" as ferrovias de suas cidades.

Seu amigo duvida que você conseguirá transformar o sistema de ferrovias em uma configuração ideal apenas utilizando essa frase. Você realmente quer provar que ele está errado. Você não só vai achar uma solução, mas, mais do que isso, vai lhe dizer de quantas maneiras diferentes pode completar o objetivo. Mais precisamente, digamos que um conjunto de líderes é *bom* se contatando precisamente cada um dos líderes desse conjunto uma única vez, a configuração ideal é obtida. Você vai dizer ao seu amigo quantos conjuntos bons distintos existem.

Dois conjuntos de líderes são considerados distintos se existe pelo menos um líder contido em um conjunto que não está contido no outro.

Como esse número pode ser muito grande, você deverá informá-lo módulo $10^9 + 7$.

Entrada

A primeira linha contém dois inteiros, N e M ($1 \leq N \leq 100$, $0 \leq M \leq N \times (N-1)/2$), representando o número de cidades e o número de ferrovias inicialmente ativas, respectivamente.

Cada uma das próximas M linhas contém dois inteiros, u e v ($1 \leq u < v \leq N$), representando a existência de uma ferrovia inicialmente ativa entre as cidades u e v. É garantido que não existem duas dessas linhas iguais.

Saída

Imprima uma linha contendo a quantidade de conjuntos de líderes que tornam a configuração ideal módulo $10^9 + 7$.

Exemplo de entrada 1	**Exemplo de saída 1**
5 6 1 2 1 3 2 4 3 4 3 5 4 5	8

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

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

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

Exemplo de entrada 5	**Exemplo de saída 5**
2 0	2

Exemplo de entrada 6	**Exemplo de saída 6**
10 15 1 6 1 2 1 5 6 7 6 10 2 3 2 9 7 3 7 8 3 4 8 9 8 5 4 5 4 10 9 10	0

Problema J

Jogo Duro

Futebol nem sempre foi o esporte mais popular das Américas. Historiadores encontraram registros de um antigo esporte que era jogado em muitas civilizações pelo continente. Por causa da falta da tradição de se falar sobre este esporte, o nome original é desconhecido, mas em tempos modernos ele é criativamente chamado de "butefol".

Nós não sabemos muito sobre butefol, nem as regras básicas. Porém, arqueólogos encontraram muitas anotações feitas pelos técnicos enquanto eles montavam seus times, o que nos deu dicas sobre como os times eram formados. Estas anotações estão lotadas de números e cálculos. Os técnicos de butefol tentaram minuciosamente otimizar seus times ao atribuir os jogadores às melhores posições possíveis. Para facilitar esta tarefa, eles desenvolveram uma métrica para determinar a performance de cada arranjo.

Há M posições em um campo de butefol, que são distribuídas em uma linha. Um time de butefol é composto de N jogadores, cada um o qual é designado uma posição (todos os jogadores devem ser designados a uma posição, e cada posição pode ser ocupada por zero ou mais jogadores).

Naturalmente, os jogadores não são todos iguais: cada jogador pode ter performances diferentes quando joga em posições diferentes. Concretamente, para cada jogador i e cada posição j, há um inteiro positivo $P_{i,j}$ que representa a performance do jogador i quando jogando na posição j.

Para complicar as coisas ainda mais, os treinadores também consideravam o aspecto de interação dos jogadores. Há alguns pares de jogadores que são "melhores amigos". Quando melhores amigos estão longes um do outro no campo, isto tem um impacto negativo na performance do time. Há um inteiro positivo C que representa a penalidade de performance quando melhores amigos estão longes um do outro.

Uma vez que os jogadores estejam distribuídos no campo, o valor da performance do time é calculado da seguinte maneira: primeiro, nós somamos a performance de cada jogador em sua determinada posição. Em seguida, para cada par de jogadores que são melhores amigos, nós subtraímos C multiplicado pela distância entre os dois jogadores, onde a distância entre dois jogadores é definida pela diferença (em valor absoluto) das posições onde os dois jogadores estão designados.

Nós gostaríamos de saber o quão bem os treinadores de butefol estavam formando seus times. Para isto, nós gostaríamos de saber qual é a maior performance possível de ser alcançada ao designar os jogadores de maneira ótima, dadas as performances dos jogadores em cada posição e os pares de jogadores que são melhores amigos.

Entrada

A primeira linha contém quatro inteiros N, M, K e C ($1 \leq N, M \leq 50$, $0 \leq K \leq 50$, $0 \leq C \leq 10^6$), representando a quantidade de jogadores, a quantidade de posições, a quantidade de pares de melhores amigos e a penalidade por colocar amigos longe uns dos outros.

Cada uma das N linhas seguintes contém M inteiros. O j-ésimo inteiro da i-ésima linha é $P_{i,j}$, representando a performance do jogador i se ele for designado na posição j ($0 \leq P_{i,j} \leq 10^6$).

Cada uma das K linhas seguintes contém 2 inteiros a_i e b_i ($1 \leq a_i < b_i \leq N$), que representa que os jogadores a_i e b_i são melhores amigos. Nenhum par de jogadores estará repetido nesta lista.

Saída

Imprima uma linha contendo um inteiro, representando o máximo de performance possível para o time.

Exemplo de entrada 1	**Exemplo de saída 1**
3 3 2 5 5 2 1 3 2 8 1 9 3 1 2 1 3	14

(Neste caso, a solução ótima é designar os jogadores 1 e 3 na posição 2, e o jogador 2 na posição 3, para que a soma da performance dos jogadores seja 2+8+9=19, a penalidade seja 5 pelos jogadores 1 e 2 estarem distantes por 1 posição, e a penalidade seja 0 pelos jogadores 1 e 3 estarem na mesma posição).

Problema K

Katmandu

Finalmente a pandemia está melhorando e você finalmente pode fazer a coisa com a qual esteve sonhando nos últimos anos: comer no seu restaurante favorito. Acontece que esse restaurante fica em Katmandu, mas tudo bem, é só você ir de avião.

O problema é que viajar de avião quase sempre te deixa muito cansado. Você se considera descansado se você consegue dormir por T minutos sem interrupção, ou seja, você nunca está acordado de um certo momento t até $t + T$. Além disso você dorme com muita facilidade: você consegue dormir no começo de qualquer minuto e acordar ao fim de qualquer minuto.

Claro que se você dormir demais você vai acabar perdendo todas as refeições que servem no vôo! Isso é completamente inaceitável: nenhuma oportunidade de comer de graça pode passar em branco.

Felizmente, a companhia aérea te mandou o cronograma completo do vôo: a duração do vôo, D minutos, o número de refeições que serão servidas, M, e o tempo a partir do início do vôo em que essas refeições serão servidas, y_i. Você precisa estar acordado no início do minuto em que a refeição será servida para comê-la, caso contrário você não será servido. Como você está sempre com fome, a refeição será devorada instantaneamente.

Agora você quer saber, para ter o vôo perfeito, você consegue ficar descansado e ainda assim comer todas as refeições durante o vôo?

Entrada

A primeira linha da entrada contém três inteiros, T, D, M ($1 \leq T, D \leq 10^5$, $0 \leq M \leq 1000$), que representam, respectivamente, o número de minutos consecutivos que você precisa dormir para ficar descansado, a duração do vôo e o número de refeições que serão servidas durante o vôo.

Cada uma das M linhas seguintes contém um inteiro y_i ($0 \leq y_i \leq D$). Esses inteiros representam os tempos nos quais cada refeição será servida, e são dados em ordem cronológica.

Saída

Imprima uma única linha contendo um único caractere. Se você consegue descansar durante o vôo e ainda assim comer todas as refeições, imprima '`Y`'; caso contrário, imprima '`N`'.

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

`3 10 3`
`2`
`4`
`7`
	`Y`

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

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

Exemplo de entrada 3	**Exemplo de saída 3**
`5 5 0`	`Y`

Exemplo de entrada 4	**Exemplo de saída 4**
4 8 2 5 7	Y

Exemplo de entrada 5	**Exemplo de saída 5**
4 8 2 3 4	Y

Problema L

Lembre sua Senha

Michael é o gerente de um escritório pouco conhecido, e dentro de sua sala existe um cofre com o dinheiro para pagar seus funcionários. Infelizmente, Michael esqueceu a senha do cofre, e agora é a responsabilidade de Dwight ajudar seu chefe.

A senha é uma sequência de N dígitos, contendo apenas zeros e uns, e Michael lembra do valor de apenas algumas posições da sequência, mas não da senha inteira. Michael também lembra de M intervalos da senha que são palíndromos – sua mente memoriza palíndromos, por algum motivo.

Um intervalo é um palíndromo se, e somente se, o primeiro e o último dígitos do intervalo são iguais, o segundo e o penúltimo dígitos são iguais, e assim por diante.

Agora Dwight quer saber o quão difícil vai ser recuperar a senha inteira. Você pode ajudar Dwight ao calcular o número de senhas possíveis que respeitam a memória de Michael.

Como a resposta pode ser muito grande, imprima-a módulo $10^9 + 7$.

Entrada

A primeira linha contém dois inteiros N e M ($1 \leq N \leq 3 \times 10^5$, $1 \leq M \leq 3 \times 10^5$), representando quantos dígitos a senha tem, e a quantidade de intervalos que Michael lembra que são palíndromos, respectivamente.

A segunda linha contém N caracteres s_i, representando qual dígito Michael lembra sobre cada posição da senha. Se s_i é '0' ou '1', então isto significa que o i-ésimo dígito da senha é 0 ou 1, respectivamente. Se s_i é '?', então isto significa que Michael não lembra qual é o i-ésimo dígito.

Cada uma das M linhas seguintes contém dois inteiros l_i e r_i ($1 \leq l_i \leq r_i \leq N$), que significa que o intervalo da senha iniciando da posição l_i até a posição r_i, inclusive, é um palíndromo.

Saída

Imprima o número de possíveis senhas que formam uma senha válida módulo $10^9 + 7$. Como a memória de Michael pode ser conflitante, caso não haja nenhuma senha que respeite suas memórias, imprima '0'.

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

Exemplo de entrada 2	**Exemplo de saída 2**
3 2 ??? 1 1 1 3	4

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

Problema M

Monarquia em Vertigem

A ordem de sucessão da monarquia é um tópico complexo, pois pode envolver múltiplos fatores tal como descendência, gênero, legitimidade, e religião. Geralmente a Coroa é herdada pelo filho do soberano, ou pela linhagem lateral do soberano caso ele não tenha filhos. Não é muito óbvio, não é mesmo? E este é um dos motivos pelos quais, em todos os lugares do mundo, a monarquia está quase acabando.

A Nlogônia ainda é dominada pela monarquia, mas felizmente com regras de sucessão simples. Em resumo há dois aspectos para levar em consideração: "filhos vêm antes de irmãos" e "os mais velhos vêm antes dos mais novos".

Os servos do reino mantêm uma linda e gigante tapeçaria onde a linhagem de Constante, o primeiro rei da Nlogônia, é desenhada em forma de uma árvore. Sempre que um novo membro da família nasce um novo ramo do pai para o filho é desenhado na tapeçaria. Este evento é tão importante que a lenda diz que quando um descendente de Constante tem um filho ele não vai morrer até que o nome do seu filho seja desenhado na tapeçaria. Quando alguém morre, uma cruz é desenhada perto do nome do falecido na tapeçaria. Quando o monarca morre, a tapeçaria é usada pelos servos para determinar quem deve ser o novo monarca. Para que isso seja feito, os servos analisam a árvore iniciando pelo Constante e atravessam os ramos de acordo com as regras descritas antes, "filhos vêm antes de irmãos" e "os mais velhos vêm antes dos mais novos". Eles visitam cada nó da árvore iniciando pelo Constante, seguido pelo filho mais velho de Constante, seguido pelo filho mais velho daquele filho, e assim por diante, até que encontram alguma pessoa ainda viva, ou até que um membro da família não tenha filhos, e neste caso eles voltam para o pai daquela pessoa e movem para o seu segundo filho, repetindo este processo até que um novo monarca seja encontrado.

Depois de milhares de anos no poder, a linhagem de Constante é gigante. Manter a tapeçaria e, quando a hora chega, determinar quem é o novo monarca são processos demorados e os servos Nlogonianos decidiram que está na hora de modernizar. Eles querem escrever um programa que seja usado para manter a linhagem de Constante e que também possa definir quem é o novo monarca quando o monarca anterior tragicamente morre. Dada a importância desta tarefa, os servos da monarquia querem testar o programa garantindo que ele produza a saída correta para todos os eventos que aconteceram até agora. Só existe um problema: nenhum deles é bom em programação. Por isso eles querem a sua ajuda.

Mais tecnicamente, cada pessoa na linhagem de Constante vai ser representado por um identificador inteiro positivo único. Sempre que um novo filho nasce, ele é atribuído o próximo menor inteiro único. O identificador de Constante é o número 1, e inicialmente ele é a única pessoa viva. Você terá que processar vários eventos em ordem cronológica. Sempre que uma pessoa morrer, você deve ajudar os servos a descobrirem quem é o atual monarca. É garantido que sempre haverá alguém vivo para governar.

Entrada

A primeira linha contém um inteiro Q ($1 \leq Q \leq 10^5$), representando quantos eventos devem ser processados. As próximas Q linhas conterão dois inteiros t_i e x_i cada, representando o tipo e o argumento do i-ésimo evento. Se t_i é igual a 1, então isso significa que a pessoa com identificador x_i teve um filho. Se t_i é igual a 2, então isso significa que a pessoa com identificador x_i morreu.

Saída

Para cada evento em que uma pessoa morre, você deve imprimir uma linha com um inteiro, representando o identificador do monarca atual.

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

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

Problema N

Na Trave!

Vini é um pintor de carros muito dedicado. Desde que ele aprendeu como pintar carros, o seu sonho tem sido participar da Internacional Competição de Pintores de Carros (`ICPC`).

Todo ano a região de Vini tem uma competição local para classificar todos os times competitivos de pintores de carro da região. Pintores em times que se classificaram nas melhores x posições avançam para competir na ICPC. É uma competição muito emocionante com muitos competidores novos em todos os anos, até que a fumaça nociva das tintas eventualmente faz com que os competidores se aposentem permanentemente.

Por causa de variações de verba e também por restrições da ICPC, a quantidade x pode variar de ano pra ano, o que pode acabar causando desgosto de alguns dos competidores.

No último ano de Vini como competidor, o seu time estava a uma posição de se qualificar para a ICPC. Que azar! Para fazer com que o seu sentimento de "má sorte" ficasse ainda mais forte, no ano seguinte o time que obteve a mesma colocação se classificou para a ICPC! Apesar do sentimento, depois de falar com outros antigos competidores, ele notou que muitos deles já haviam se sentido azarados de uma forma ou de outra.

Antigos competidores geralmente seguem os resultados das competições regionais por alguns anos após se aposentarem. Portanto, um competidor não se sentiria azarado pelas mudanças em x que acontecessem muitos anos após se aposentar. Mais precisamente, cada antigo competidor participou da sua última competição no ano a_i, se posicionando na posição p_i e, após se aposentar, seguiu os resultados pelos f_i anos seguintes.

Um competidor que não se qualificou para a ICPC em sua última participação se sentiu azarado em todos os anos em que ele seguiu os resultados e nos quais ele teria se classificado se tivesse competido. Em outras palavras, para cada ano até f_i anos após se aposentar, se ele não se qualificou em sua última participação, ele se sentiu azarado se o número de times qualificados para a ICPC naquele ano foi ao menos p_i.

Dado o número de vagas por ano, e as informações sobre cada antigo competidor, nós gostaríamos de saber em quantos anos cada antigo competidor se sentiu azarado.

Entrada

A primeira linha contém dois inteiros Y e N ($1 \leq Y, N \leq 3 \times 10^5$), representando a quantidade de anos de competições e a quantidade de antigos competidores com quem Vini conversou, respectivamente. (Sim, pintar carros é uma tradição milenar, e bem popular!).

A próxima linha contém Y inteiros x_1, x_2, ..., x_Y ($0 \leq x_i \leq 10^5$), representando quantas vagas para a ICPC a região teve em cada ano.

Cada uma das seguintes N linhas contém três inteiros a_i, p_i e f_i ($1 \leq a_i \leq Y$, $1 \leq p_i \leq 10^5$, $0 \leq f_i \leq Y - a_i$), representando o ano em que o i-ésimo antigo competidor teve sua última participação, a colocação do time do i-ésimo antigo competidor naquele ano, e por quantos anos o i-ésimo antigo competidor seguiu os resultados após se aposentar, respectivamente.

Saída

Imprima N linhas, onde a i-ésima linha deve conter um inteiro representando quantos anos o i-ésimo antigo competidor se sentiu azarado.

Exemplo de entrada 1	**Exemplo de saída 1**
5 3 1 2 3 4 5 1 3 4 2 6 3 3 4 1	3 0 1

Exemplo de entrada 2	**Exemplo de saída 2**
4 1 8 8 8 8 1 7 3	0