

Seletiva UnB 2023

Caderno de Problemas

01 de julho de 2023



(Este caderno contém 15 problemas)

Comissão Organizadora:

Prof. Edson Alves da Costa Júnior (UnB/FGA)

Prof. Guilherme Novaes Ramos (UnB)

Prof. Vinicius Ruela Pereira Borges (UnB)

Prof. Daniel Saad Nogueira Nunes (IFB)

Alberto Tavares Duarte Neto

Eduardo Quirino

Lucas Mattioli

Jeremias Moreira Gomes

José Marcos da Silva Leite

Tiago de Souza Fernandes

Lembretes

- É permitido consultar livros, anotações ou qualquer outro material impresso durante a prova, entretanto, o mesmo não vale para materiais dispostos eletronicamente.
- A correção é automatizada, portanto, siga atentamente as exigências da tarefa quanto ao formato da entrada e saída conforme as amostras dos exemplos. Deve-se considerar entradas e saídas padrão;
- Para cada problema, além dos testes públicos, o juiz executará a sua submissão contra uma série de testes secretos para fornecer um parecer sobre a correção do programa.
- Procure resolver o problema de maneira eficiente. Se o tempo superar o limite pré-definido, a solução não é aceita. Lembre-se que as soluções são testadas com outras entradas além das apresentadas como exemplo dos problemas;
- Utilize a aba *clarification* para dúvidas da prova. Os juízes podem opcionalmente atendê-lo com respostas acessíveis a todos;

C/C++

- Seu programa deve retornar zero, executando, como último comando, `return 0` ou `exit 0`.

Java

- Não declare ‘`package`’ no seu programa Java.
- Note que a convenção para o nome do arquivo fonte deve ser obedecida, o que significa que o nome de sua classe pública deve ser uma letra maiúscula igual a letra que identifica o problema.

Python

- Tenha cuidado ao selecionar a versão correta na submissão.

Problema A

Aniversário

Limite de tempo: 2s

Autor: Alberto Tavares

Ruan, mais conhecido como MagePetrus, fez aniversário recentemente e vai chamar algumas pessoas para seu aniversário; ele decide, junto de seus $n - 1$ amigos, pegar uma aresta (gíria para pegar um ônibus) para uma lanchonete próxima para comemorar.

Na lanchonete, Arthur e Dudu propõem um jogo: serão escritos os nomes de todos as n pessoas presentes em papéis, que serão embaralhados, e cada pessoa sorteará um papel com um nome — esta pessoa deverá pagar a conta de quem ela sorteou.

Mais formalmente, um **sorteio** é uma permutação p dos números $1, \dots, n$, com cada número de 1 até n representando uma pessoa diferente, de forma que $p[i] = j$ representa i sortear j .

Uma **sequência maluca** de um determinado sorteio p é definido como uma sequência de m números **distintos** (c_1, c_2, \dots, c_m) tais que $p[c_i] = c_{i+1}$ para $1 \leq i < m$. A **diversão** de um sorteio p é o tamanho de sua maior sequência maluca.

Considerando que n pessoas participarão do aniversário, incluindo Ruan, e o sorteio será completamente aleatório, calcule o valor esperado da diversão do sorteio realizado no aniversário. Como este número pode ser muito grande (ou muito pequeno), calcule-o módulo 998244353.

Entrada

A primeira e única linha de input contém um único inteiro n ($1 \leq n \leq 5000$) — a quantidade de pessoas na festa de Ruan que participará do sorteio.

Saída

Imprima um único inteiro — o valor esperado do maior ciclo módulo 998244353.

Pode ser provado que o valor esperado pode ser escrito como uma fração irredutível p/q com $q \neq 0$. Deve ser impresso $p \cdot q^{-1} \pmod{998244353}$.

Exemplo

Entrada	Saída
1	1
2	499122178
4	540715694
2506	23772374

Notas

Assuma que todas as possibilidades de sorteio (i.e. todas as permutações) tem a mesma probabilidade de acontecer.

Consideremos o sorteio de tamanho n representado pela permutação $[2, 4, 5, 1, 3]$. As sequências $(1, 2, 4)$ e $(5, 3)$ são malucas; as sequências $(1, 2, 3)$ e $(5, 3, 5, 3)$ não são malucas.

No caso de teste 1, apenas Ruan foi em seu aniversário. Neste caso existe apenas uma possibilidade de sorteio, e sua única sequência maluca tem tamanho 1.

No caso de teste 2, as duas possibilidades de sorteio são $[1, 2]$ e $[2, 1]$ — a primeira tem diversão 1 e a segunda tem diversão 2. O valor esperado, portanto, é a fração $3/2$.

O input no caso de teste 4 é a data de aniversário do Ruan, 25/06.

Problema B

Balão++

Limite de tempo: 1s

Autor: Guilherme Ramos

O entusiasmo dos maratonistas é contagiante! Sempre que avistam balões, eles já se animam para a próxima conquista.

Dada a descrição do que foi visto, indique a reação dos maratonistas.

Entrada

A entrada consiste de uma linha indicando o que foi visto. A informação é descrita com até 20 caracteres minúsculos entre a-z, podendo haver hífen para nomes compostos. Não há espaços ou caracteres especiais.

Saída

Se pelo menos um balão for visto, os maratonistas clamam por “Mais! Mais!”. Qualquer outra coisa merece um sonoro “Bu!”.

Exemplo

Entrada	Saída
baloos	Mais! Mais!
fantasma	Bu!
guarda-chuva	Bu!
bambalao	Bu!
balao	Mais! Mais!
balao-de-agua	Bu!

Notas

No primeiro exemplo, mais de um balão foi visto.

Problema C

Caos Center

Limite de tempo: 3s

Autor: Vinicius Borges

Um operador de call center está passando por dificuldades na empresa em que trabalha. Tal empresa é conhecida pela venda de produtos de eletrônica e de informática, mas não investe muito no relacionamento com o cliente por se tratar de uma empresa de fachada. Por isso, os produtos vendidos são de baixa qualidade, o que acaba por gerar transtornos aos clientes que necessitam de atendimento para trocas/devoluções e reclamações.

Em um determinado momento, N clientes entram simultaneamente na fila de atendimento por telefone. Os clientes são identificados por inteiros de 1 a N de acordo com a ordem inicial de atendimento na fila, isto é, o operador irá atender inicialmente o cliente 1, depois o cliente 2 e assim por diante. Acontece que a demora nas resoluções dos problemas dos clientes no call center pode levar a alguns contratempos. Assim, as seguintes situações podem ocorrer com o cliente k na fila de atendimento:

- Se o operador estiver livre ou ter acabado de finalizar um atendimento, o cliente k é prontamente atendido em t_k minutos;
- Se o operador estiver ocupado, o cliente espera até p_k minutos desde o momento de entrada na fila para ser atendido. Caso isso não ocorra, ele perde a paciência, encerra a ligação e liga novamente após r_k minutos, entrando no final da fila de atendimento. Caso dois clientes retornem ao mesmo tempo para a fila de atendimento, terá prioridade o cliente com o menor identificador.

Com uma grande quantidade de clientes furiosos, o operador de call center percebe o caos a seu redor. Ajude-o ao determinar o tempo total gasto para que ele finalize todos os N atendimentos.

Entrada

A primeira linha da entrada contém um número inteiro N ($1 \leq N \leq 10^3$) indicando a quantidade de clientes a serem realizados pelo operador de call center.

As próximas N linhas apresentam os clientes. Cada linha contém três inteiros separados por espaço t_i , p_i e r_i ($1 \leq t_i, p_i, r_i \leq 10^3$) representando o tempo que cada cliente pretende utilizar para resolver seu problema, a paciência do cliente na fila de espera e o tempo de espera para que o cliente faça uma nova ligação para o call center após o encerramento da última ligação, respectivamente.

Saída

Imprima um único número inteiro indicando o tempo total gasto para que o operador de call center finalize todos os atendimentos por telefone.

Exemplo

Entrada	Saída
3	10
5 2 1	
3 6 2	
2 4 3	
4	17
4 3 10	
5 2 3	
3 2 5	
2 4 2	
5	26
3 1 2	
2 3 5	
5 2 4	
1 1 2	
14 4 3	

Notas

No primeiro exemplo, o primeiro cliente é atendido, deixando o operador ocupado por 5 minutos. Como o cliente 2 consegue esperar 6 minutos, ele será atendido após o operador encerrar o primeiro atendimento. Enquanto isso, o cliente 3 saiu da fila após esperar 4 minutos, mas retorna após 3 minutos. Acontece que o operador ainda estará ocupado atendendo o cliente 2. O cliente 3 espera mais 1 minuto e logo é atendido pelo operador.

Problema D

Desfazendo a bagunça

Limite de tempo: 1s

Autor: Edson Alves

Uma empresa que trabalha com enfeites natalinos prepara, sob encomenda, fitas de pisca-pisca coloridos. Cada fita é composta por N lâmpadas, ligadas em série, e cada lâmpada é capaz de exibir uma dentre as 99 cores disponíveis, mediante um botão de configuração.

Contudo, o estagiário em treinamento fez confusão com os fios, botões e configurações, de modo que os botões das lâmpadas que ficaram nos extremos da fita (posições 1 e N) já não funcionam mais, e os demais botões de cada lâmpada, se apertados, ao invés de trocar a cor da lâmpada, inverte as cores de suas lâmpadas vizinhas!

Resta a você desfazer a bagunça do estagiário: dado o valor de N e as cores que a fita exibe no momento, e sem mexer na fiação (ou seja, usando o novo comportamento dos botões depois do “excelente trabalho” do estagiário), você deve tentar fazer com que a fita fique na configuração solicitada pelo cliente, se possível.

Entrada

A primeira linha da entrada contém o valor do inteiro N ($3 \leq N \leq 2 \times 10^5$).

A segunda linha contém N inteiros x_i ($1 \leq x_i \leq 99, 1 \leq i \leq N$), separados por um espaço em branco, onde x_i é a cor da lâmpada i , após a bagunça do estagiário.

A terceira linha contém N inteiros y_j ($1 \leq y_j \leq 99, 1 \leq j \leq N$), separados por um espaço em branco, onde y_j é a cor que o cliente deseja para a lâmpada j .

Saída

Imprima, em uma linha, a mensagem “Sim”, caso seja possível desfazer a bagunça do estagiário; ou a mensagem, “Nao”, caso contrário.

Exemplo

Entrada	Saída
5	Sim
5 4 3 2 1	
1 2 3 4 5	
3	Nao
3 1 1	
2 1 3	
10	Sim
1 1 3 2 5 7 2 5 7 7	
5 7 2 7 7 1 3 2 1 5	

Notas

No primeiro caso, uma maneira de se obter a configuração solicitada pelo cliente é descrita a seguir. Inicialmente, as lâmpadas estão na seguinte ordem:

5 4 3 2 1

Ao apertar o botão da quarta lâmpada, suas lâmpadas vizinhas trocam de cor, de modo que a fita passa a ter a seguinte configuração:

5 4 1 2 3

Agora, ao apertar o botão da segunda lâmpada se obtém

1 4 5 2 3

Um toque no botão da terceira lâmpada segue a configuração

1 2 5 4 3

Por fim, ao pressionar novamente o botão da quarta lâmpada chega-se à configuração desejada:

1 2 3 4 5

Problema E

Egito Antigo

Limite de tempo: 1s

Autor: Daniel Saad

Cleiton, o bom de guerra, após causar altas confusões na Grécia antiga e nas regiões nórdicas, resolveu se aventurar no Egito antigo. Junto a seu filho Arturo, eles precisam resolver vários mistérios para poder explorar a nova região.

Ao se deparar com a esfinge mítica, ela propôs um desafio que envolviam frações. Os povos egípcios tinham o costume de expressar qualquer fração através de uma soma de frações com numerador 1. Por exemplo, a fração $\frac{14}{15}$ pode ser expressa como:

$$\frac{14}{15} = \frac{1}{2} + \frac{1}{3} + \frac{1}{10}$$

O desafio proposto pela esfinge consistia em determinar a fração equivalente na norma egípcia. Preocupados com o desafio, Cleiton e Arturo recorreram a um método que aprenderam durante suas aventuras na Roma antiga, com um tal de Fibonacci. Para representar uma fração $\frac{x}{y}$ através deste método, é necessário, a cada passo, subtrair de $\frac{x}{y}$ a maior fração $\frac{1}{d}$ possível e reaplicar a estratégia com o valor que restar, até que não seja mais possível.

Ajude Cleiton e Arturo a resolver o desafio proposto pela esfinge com o método de Fibonacci para que eles possam continuar a desbravar o Egito antigo.

Entrada

A primeira linha da entrada possui um inteiro n , contendo o número de desafios propostos pela esfinge.

Cada uma das próximas n linhas, descreve um desafio, que é da forma x/y , sendo x o numerador e y o denominador.

Restrições

- $1 \leq n \leq 10^3$
- $1 \leq x < y \leq 10^5$

Saída

Para cada desafio da esfinge, imprima as frações com o método de Fibonacci cuja soma é equivalente ao desafio. Cada fração é da forma $1/d$ e elas devem estar separadas pelo símbolo $+$. As frações devem estar organizadas em ordem decrescente.

Exemplo

Entrada	Saída
4	$1/2$
$2/4$	$1/3 + 1/9 + 1/99$
$5/11$	$1/3 + 1/8 + 1/120$
$7/15$	$1/2 + 1/3 + 1/10$
$14/15$	

Problema F

Farejando Deadlocks

Limite de tempo: 3s

Autor: José Leite

Em sistemas distribuídos, é comum o conceito de processos e recursos onde um processo pode pedir acesso exclusivo a recursos.

Recursos podem ser uma linha de uma tabela de banco de dados, leitura de entrada de teclado, uma impressora, etc.

Para este problema, um processo é uma lista de instruções no formato:

- 1 d , fazendo um processamento durante d ms;
- 2 r , pedir o recurso r ;
- 3 r , devolver o recurso r .

O tempo de pedir ou devolver um recurso é irrisório, mas logo após o recurso ser alocado ou liberado, o processo gasta 1ms de *overhead*.

Todos os processos serão executados ao mesmo tempo. Caso vários estejam tentando pegar recursos ao mesmo tempo, o processo com menor índice terá o recurso alocado.

Um *deadlock* acontece quando dois ou mais processos ficam impedidos de continuar suas execuções, pois existe uma espera circular por recursos alocados por eles.

Dada uma lista de processos, identifique se há algum *deadlock*. Caso haja, informe o índice do processo e instrução a inserir o *deadlock* pela primeira vez. Caso vários *deadlocks* se formem ao mesmo tempo, informe o processo com menor índice.

Entrada

A primeira linha da entrada contém um inteiro T ($1 \leq T \leq 10^4$), indicando o número de casos de teste.

A primeira linha de cada caso de teste contém dois inteiros P ($1 \leq P \leq 10^5$) e R ($1 \leq R \leq 10^5$), o número de processos e de recursos.

Após, temos a descrição de cada processo. A primeira linha tem o número de instruções I ($1 \leq I \leq 10^5$). As próximas I linhas contém dois inteiros t_i ($1 \leq t_i \leq 3$) e v_i , as instruções do processo. Para $t_i = 1$, então $1 \leq v_i \leq 10^5$. Para $t_i = 2$ ou $t_i = 3$, então $0 \leq v_i < R$. É garantido que um processo só vai pedir um recurso que ainda não é mantido por ele e que só libera um recurso previamente alocado por ele. Ademais, todo processo sempre terá instruções para devolver todos os recursos alocados.

É garantido que a soma do número de processos não excede 5×10^5 .

É garantido que a soma do número de recursos não excede 5×10^5 .

É garantido que a soma do número de instruções de todos os processos não excede 10^6 .

Saída

A saída deve conter T linhas, em que cada uma é uma resposta para um caso de teste.

Para o i -ésimo caso de teste, caso haja não haja um deadlock imprima “Sem deadlock”, sem aspas. Caso haja deadlock “Deadlock em p i ”, sem aspas, onde p é o índice do processo e i é o índice da instrução que introduziu o deadlock pela primeira vez.

Exemplo

Entrada	Saída
2	Deadlock em 1 1
4 4	Sem deadlock
4	
2 0	
2 1	
3 0	
3 1	
4	
2 1	
2 0	
3 0	
3 1	
4	
2 2	
2 3	
3 2	
3 3	
4	
2 3	
2 2	
3 2	
3 3	
2 5	
5	
2 1	
2 2	
1 10	
3 1	
3 2	
5	
2 2	
1 5	
3 2	
2 1	
3 1	

Problema G

G de StrinG

Limite de tempo: 2s

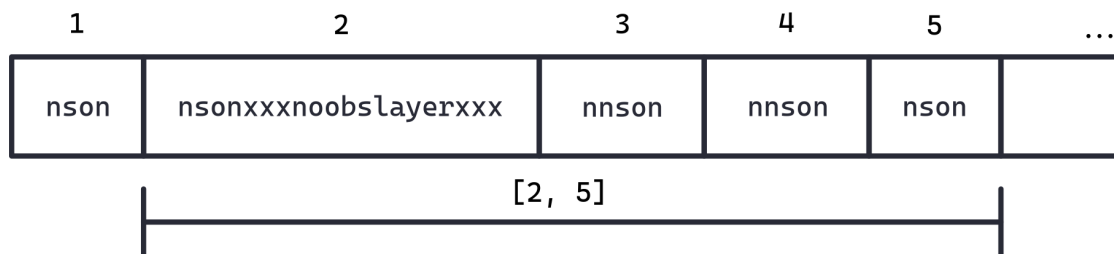
Autor: Tiago Fernandes

Existem N companhias especializadas em venda de strings na sua cidade, e cada uma vende apenas uma string diferente, S_i . A companhia i possui k_i lojas, localizadas nas posições x_j . Todas as posições de lojas são distintas.

Seus amigos querem comprar strings com suas respectivas handles no CodeForces, contudo é difícil encontrar lojas que vendam exatamente a string desejada. Para cada amigo i , é suficiente que ele compre uma string S que contenha sua handle H_i em seu **prefixo**, pois eles podem simplesmente cortar do final da string S e obter a handle.

Como seus amigos moram em locais diferentes, eles não querem andar muito para comprar a handle, logo, cada amigo i possui um intervalo fechado de posições $[l_i, r_i]$ que ele consegue visitar.

Query 1 = nson [2, 5]



Ajude-os calculando, para cada amigo, em quantas lojas no intervalo especificado ele vai encontrar uma string S que possua H_i no seu prefixo.

Entrada

A primeira linha da entrada consiste de um valor inteiro N ($1 \leq N \leq 2 \cdot 10^5$), representando a quantidade de companhias na cidade.

Seguem $3 \cdot N$ linhas. A primeira linha possui uma string S_i , representando a string vendida na i -ésima companhia, a segunda um inteiro k_i , a quantidade de lojas da companhia i , e a terceira, k_i inteiros x_j ($1 \leq x_j \leq 5 \cdot 10^5$), as posições das lojas. É garantido que $\sum k_i \leq 5 \cdot 10^5$, e que as lojas não ocupam uma mesma posição x .

A próxima linha possui um inteiro Q ($1 \leq Q \leq 2 \cdot 10^5$), representando a quantidade de amigos.

As próximas Q linhas possuem uma string H_i e dois inteiros l_i e r_i ($1 \leq l_i \leq r_i \leq 5 \cdot 10^5$), representando, respectivamente, a handle e o range de lojas onde o i -ésimo amigo deseja comprar.

Todas as strings possuem apenas caracteres minúsculos de a a z . É garantido que o somatório do tamanho das strings do input não ultrapassa $5 \cdot 10^5$.

Saída

Imprima Q inteiros, representando para cada amigo i , na mesma ordem da entrada, a quantidade de lojas diferentes dentro do range $[l_i, r_i]$ onde ele consegue comprar uma string S que possui H_i como prefixo.

Exemplo

Entrada	Saída
3	2
nson	
2	
1 5	
nnson	
2	
3 4	
nsonxxxnoobslayerxxx	
1	
2	
1	
nson 2 5	
3	5
cebolinha	5
3	10
123 9876 500000	2
calebmartim	0
5	0
2 4 6 8 10	
cachorro	
5	
1 3 5 7 9	
6	
caleb 1 10	
cachorro 1 10	
ca 1 10	
cebolinha 1000 500000	
liaoli 1 500000	
sorvetinho 1 1	

Notas

No primeiro caso de teste, como é possível visualizar na imagem do enunciado, o range $[2, 5]$ possui quatro lojas, duas com a string “nson”, uma “nson” e uma “nsonxxxnoobs-layerxxx”. A string “nson” é prefixo de duas dessas lojas, logo, a resposta para este amigo é 2.

Problema H

Hora do Lanche

Limite de tempo: 1s

Autor: Author name

Chegou a hora do tão sonhado lanche da Escola Classe 3 da Nlogônia! Inicialmente, todos os N estudantes estão brincando em um pátio em formato quadrático no prédio desta escola. Cada estudante da escola é numerado por um inteiro de 1 a N . O lugar ocupado pelo i -ésimo estudante é dado pela posição (x_i, y_i) . Quando o sinal toca, todos os estudantes correm em direção à cantina da escola, que está localizada na posição (x_c, y_c) do pátio.

Apesar dos esforços da escola para lidar com as limitações orçamentárias, nem sempre há lanches para todos os estudantes. A cantina é capaz de oferecer K lanches, sendo fornecido exatamente 1 lanche para cada estudante. Logo, apenas os primeiros K estudantes que chegarem à fila da cantina receberão seus lanches.

Supondo que os estudantes correm na mesma velocidade e que eles sempre percorrem a menor distância possível entre sua posição até a cantina, indique o primeiro estudante e o último estudante que receberão lanches conforme suas ordens de chegada. No caso de dois ou mais estudantes chegarem juntos à cantina, o estudante com o maior índice inteiro terá prioridade. Como o prédio é grande, é possível que dois ou mais estudantes ocupem uma mesma posição.

Entrada

A primeira linha da entrada contém dois inteiros separados por espaço N e K ($1 \leq N \leq 2 \cdot 10^5, 1 \leq K \leq N$), denotando a quantidade de estudantes e a quantidade de lanches oferecidos pela cantina, respectivamente.

A segunda linha contém dois inteiros separados por espaço em branco x_c e y_c ($1 \leq x_c, y_c \leq 10^9$), representando as coordenada da cantina em relação ao pátio da escola.

Em seguida, existem N linhas, em que a i -ésima linha apresenta dois inteiros separados por espaço x_i e y_i ($1 \leq x_i, y_i \leq 10^9, x_i \neq x_c, y_i \neq y_c$), indicando a posição do i -ésimo estudante no pátio.

Saída

Imprima os identificadores do primeiro e do último estudantes que receberão lanches de acordo com a ordem de chegada à cantina. Caso existam dois ou mais estudantes chegando na cantina ao mesmo tempo, imprima os estudantes em ordem decrescente em relação aos seus índices inteiros.

Exemplo

Entrada	Saída
3 2	2 1
2 2	
1 1	
2 3	
4 4	
4 3	4 2
2 2	
3 1	
1 1	
1 3	
3 3	
6 4	3 5
5 7	
5 1	
9 4	
1 4	
8 2	
3 2	
3 2	

Notas

No primeiro exemplo, a ordem das distâncias dos estudantes em relação à cantina é dada pelos estudantes 2, 1 e 3. Como a cantina oferece apenas $K = 2$ lanches, a resposta é 2 e 1.

No segundo exemplo, todos os quatro estudantes estão à mesma distância da cantina, logo os estudantes 4 e 2 são os primeiro e último a receberem seus lanches.

Problema I

Invertendo e Ganhando

Limite de tempo: 1s

Autor: Daniel Saad

Juliano, grande fã de programas de auditório, ficou rico ao participar do programa “*Apagando e ganhando*”, contudo, perdeu todo o seu dinheiro fazendo apostas esportivas. Para recuperar a grana, resolveu participar de outro programa, chamado “*Invertendo e ganhando*”.

Neste programa o apresentador escreve n números, (v_1, \dots, v_n) em uma lousa e o participante deve obter o maior número após realizar, **no máximo**, k trocas. Uma troca consiste em trocar dois elementos v_i e v_j de lugar, com $1 \leq i, j \leq n$ e $i \neq j$. Caso Juliano consiga obter o maior número, ele ganha o prêmio de um milhão de reais em barras de ouro.

Ajude Juliano ao ganhar o prêmio ao calcular o maior número possível que pode ser obtido.

Entrada

A primeira linha da entrada possui dois números, n e k , indicando, respectivamente, a quantidade de dígitos e o número máximo de trocas que podem ser realizadas. A próxima linha possui n dígitos, descrevendo o número inicial.

Restrições

- $1 \leq n \leq 25$
- $0 \leq k \leq 5$

Saída

Imprima, em uma linha, o maior número que Juliano pode obter após realizar, no máximo, k trocas no número original.

Exemplo

Entrada	Saída
5 3 98765	98765
5 3 12345	54321
5 2 91523	95321

Problema J

Jogo do Pulo

Limite de tempo: 1s

Autor: Edson Alves

As crianças do jardim de infância estão aprendendo a contar. Para auxiliá-los e tornar o aprendizado mais lúdico, a professora criou o Jogo do Pulo, que funciona da seguinte maneira: primeiramente, a professora desenha com o giz, no chão, um círculo com N posições, numeradas de 0 a $N - 1$. Então cada criança se dirige ao círculo, escolhe uma posição, fica de pé sobre a marcação e escreve, em uma folha, o número da sua posição. A cada turno, na ordem dos números escritos nas folhas, cada criança avança no círculo p posições, onde p é o número escrito na sua folha. Vence a criança que chegar primeiro à posição 1 em um ou mais turnos.

Por exemplo, se $N = 5$, a criança que escolheu a posição 0 ficará parada toda a brincadeira; a que escolheu a posição 1 levaria 5 turnos para chegar à posição 1; a da posição 2 levaria 2 turnos; a criança da posição 3 levaria 1 turno e, por fim, a criança da posição 4 levaria 3 turnos. Portanto seria vencedora a criança que escolheu a posição 3.

Dado o valor de N , determine a posição inicial da criança que vencerá a brincadeira.

Entrada

A entrada consiste em uma única linha, contendo o inteiro N ($2 \leq N \leq 10^9$).

Saída

Imprima, em uma linha, a posição inicial da criança que vencerá a brincadeira.

Exemplo

Entrada	Saída
5	3
31	16
3141592	2094395

Notas

O primeiro caso foi detalhado no texto do problema.

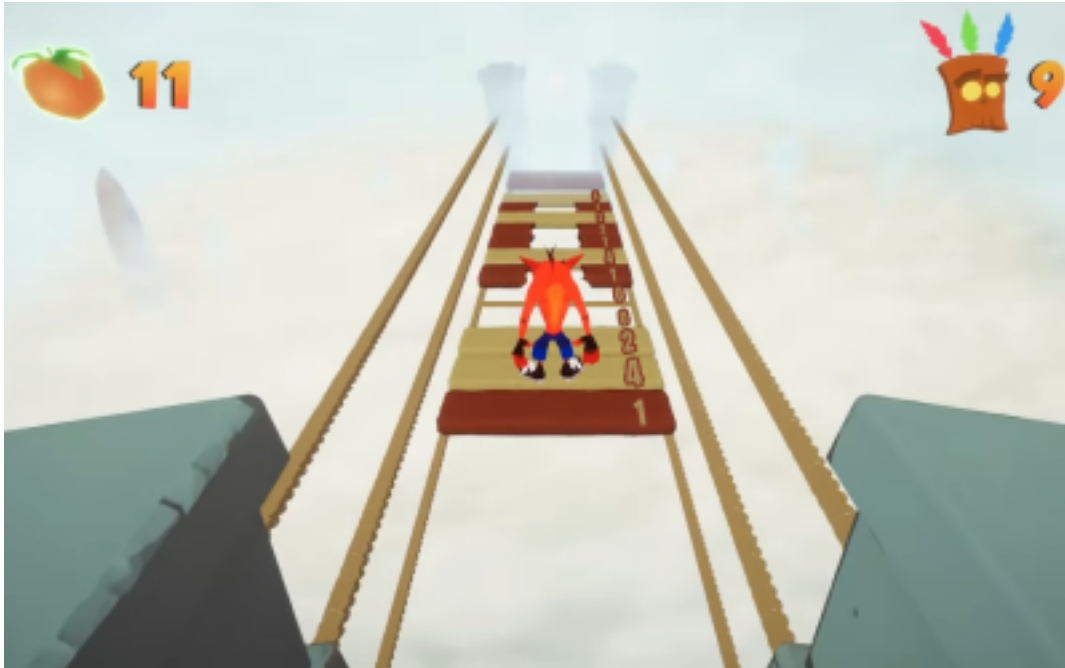
No segundo caso, a criança que ocupa, inicialmente, a posição 16 levaria apenas um turno para chegar à posição 1.

Problema K

Krash Bondjump

Limite de tempo: 1s

Autor: Jeremias Gomes



Krash Bondjump é um jogo de plataforma onde os jogadores assumem o papel de Krash, um marsupial em uma missão para derrotar o maligno Dr. Mentex. Nessa aventura, um dos mundos o qual Krash precisa sobreviver é composto por fases onde é necessário atravessar pontes com problemas de sustentação, e Krash vai pulando pavimentos nessas pontes.

Cada ponte é um agregado contíguo de lacunas ou pavimentos de madeira, que ocupam uma posição cada, e Krash começa a fase no primeiro pavimento dessa ponte. Cada pavimento possui o seu valor de sustentação e, uma vez que Krash está em cima de um pavimento, ele sabe que somente consegue pular no máximo o valor de sustentação daquele pavimento para a próxima posição. Dessa forma, Krash salta diferentes distâncias de pavimentos tentando chegar até a última posição dessa ponte.

Assim, sua tarefa é, dada as descrições de sustentação de uma ponte, informar se é possível para Krash ir pulando por essa ponte até o final ou não.

Entrada

A primeira linha da entrada contém um inteiro F ($1 \leq F \leq 100$), que indica o número de fases do jogo.

Em seguida há $2 * F$ linhas, onde cada um dos pares de linhas seguintes representam uma fase. A primeira linha de uma fase contém um inteiro P ($2 \leq P \leq 10000$) que indica o número de pavimentos da ponte naquela fase e a segunda linha de uma fase contém P

inteiros B_i ($0 \leq B_i \leq 100$), separados por um espaço em branco, que indicam o valor de sustentação daquele pavimento B_i . Além disso é garantido que B_1 e B_P , que marcam o início e o fim de cada fase, são diferentes de zero.

Saída

A saída deve conter F linhas, onde para cada fase deve-se responder "Possível passar" se for possível passar de fase, ou "Impossível passar", caso contrário.

Exemplo

Entrada	Saída
1	Possível passar
13	
1 4 2 0 0 1 4 1 1 2 3 1 5	
2	Impossível passar
7	Possível passar
2 2 1 0 3 4 7	
6	
1 4 0 0 0 1	

Notas

No primeiro caso de teste, que possui apenas uma fase (1, 4, 2, 0, 0, 1, 4, 1, 1, 2, 3, 1, 5), Krash inicia na posição 1, que tem sustentação 1. Dada essa sustentação, ele pode pular apenas para o pavimento 2, que possui sustentação 4. A partir do pavimento 2, ele consegue pular para qualquer um dos pavimentos 3, 4, 5 ou 6. Então, escolhendo pular para o pavimento 6, as escolhas seguintes de pavimentos que permite passar de fase podem ser ser 7, 11, 13.

Problema L

Limite de Resistência

Limite de tempo: 1s

Autor: Alberto Tavares

Os maratonistas da Universidade dos Brabos (UnB) estão se preparando para a fase seletiva do ICPC (Incrível Campeonato dos Pés Celeres). Todos sabem que uma maratona é uma corrida de longa distância na qual a resistência é crucial. Por essa razão, os competidores decidiram realizar algumas corridas em diferentes partes de Brasília como parte do treinamento.

Para representar as rotas de corrida em Brasília, os competidores desenharam um mapa em papel, que pode ser visualizado como uma árvore. Essa árvore é um grafo conectado com n vértices e $n - 1$ arestas, em que cada aresta representa uma unidade de distância. Um caminho válido para a corrida é qualquer caminho que **não visite** vértices repetidos.

Ao escolher os pontos de partida e chegada para a corrida, os maratonistas perceberam que existem muitas possibilidades! No entanto, como eles estão interessados em testar seus limites de resistência, eles estão interessados apenas nos **dois** caminhos mais longos.

Considerando que os maratonistas estão ocupados correndo, sua tarefa é determinar o comprimento dos dois caminhos mais longos e quantos caminhos possuem esses mesmos comprimentos.

Entrada

A primeira linha de entrada contém um único inteiro n ($2 \leq n \leq 10^5$) — a quantidade de vértices no mapa de Brasília.

Cada uma das próximas linhas contém dois inteiros a, b ($1 \leq a, b \leq n$) — representando uma aresta entre a e b .

Saída

Imprima duas linhas contendo dois inteiros cada.

A primeira linha de saída deve conter dois inteiros s_1, q_1 — o comprimento do maior caminho na árvore e a quantidade de caminhos com este comprimento, respectivamente.

A segunda linha de saída deve conter dois inteiros s_2, q_2 — o comprimento do **segundo** maior caminho na árvore e a quantidade de caminhos com este comprimento, respectivamente.

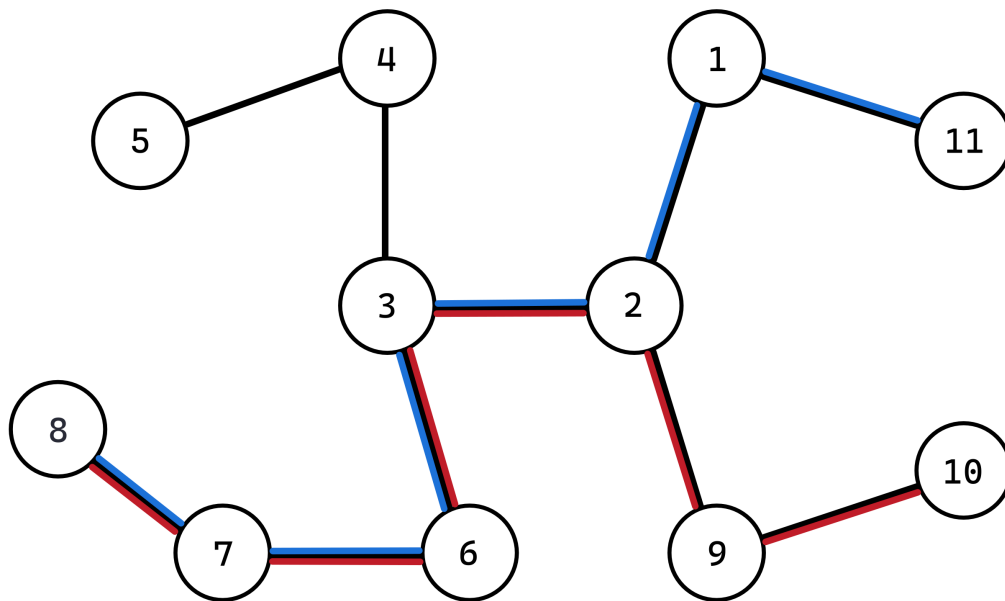
Exemplo

Entrada	Saída
5	3 2
1 2	2 4
1 3	
2 4	
5 2	
11	6 2
1 2	5 7
1 11	
2 3	
2 9	
3 4	
3 6	
4 5	
6 7	
7 8	
9 10	
5	4 1
1 2	3 2
2 3	
1 4	
4 5	

Notas

Um caminho em um grafo é uma sequência de vértices conectados por arestas, onde cada aresta do caminho tem uma extremidade em comum com a aresta seguinte. O comprimento de um caminho é a quantidade de arestas que o compõem. Lembrando que um caminho não pode repetir vértices.

O caso de teste 2 tem dois caminhos de comprimento 6, como visto na imagem abaixo:



Por isso que a camisa do UnBalloon tem que ter tecido bom, quem disse que maratonista não corre?

Problema M

montinhos especiais

Limite de tempo: 2s

Autor: Eduardo Quirino

grug, bravo guerreiro, ter N números, um de cada 1 até N . grug amar montinhos especiais, grug achar eles mágicos.

grug saber montinho especial ser lista de números onde cada número, menos primeiro, menor que número na metade de índice. grug saber que, se índice ímpar, arredondar para baixo. grug ter cérebro pequeno então grug indexar em 1. grug achar regra fácil.

grug querer mostrar regra para bravos guerreiros. grug saber que lista com 5, 3, 4, 1, 2 ser montinho especial porque 3 menor que 5, 4 menor que 5, 1 menor que 3 e 2 menor que 3.

grug olhar lista de números e ficar feliz. grug querer fazer todos montinhos especiais, mas grug não ter tempo. grug lutar com dinossauros e dar comida para família. grug ter vida difícil.

grug pedir ajuda para bravos guerreiros. grug querer achar quantos montinhos especiais fazer com lista de números. grug acreditar que bravos guerreiros inteligentes e sábios. grug achar número muito grande, não caber em tablete de pedra. grug pedir número módulo 998244353.

grug achar que montinhos especiais servir para alguma coisa. grug ser muito ingênuo e sonhador. grug não saber que montinhos especiais só servir para enfeitar. grug gastar tempo e energia com montinhos especiais. grug ser motivo de piada para família e dinossauros.

Entrada

A entrada consiste em um único inteiro N , $1 \leq N \leq 10^6$.

Saída

A saída deve ser um único inteiro: a quantidade de permutações dos elementos $1, 2, 3, \dots, N$ que são montinhos especiais.

A resposta deve ser dada módulo 998244353.

Exemplo

Entrada	Saída
3	2
10	3360
10000	330147944

Notas

Uma permutação $P_1, P_2, P_3, \dots, P_N$ é um **montinho especial** se para qualquer $i \in [2, N]$, temos que $P_{\lfloor i/2 \rfloor} < P_i$.

A notação $\lfloor x \rfloor$ representa o maior inteiro que é menor ou igual à x . (e.g. $\lfloor 3 \rfloor = 3$, $\lfloor 9/2 \rfloor = 4$).

Problema N

Nara

Limite de tempo: 2s

Autor: Lucas Mattioli

Dada uma string S , uma string T e um inteiro K , dizemos que T **pertence** a S caso exista ao menos uma substring de T de tamanho pelo menos K que é uma substring de S .

Por exemplo, se $K = 3$, $S = 111101010$ e $T = 0101010$, então T **pertence** a S porque a substring $T[1..4] = 1010$ (que tem tamanho 4, que é maior ou igual a K) é uma substring de S ($S[5..8]$). Em contrapartida, se T tivesse o valor 00011 , então ela não pertenceria a S porque ela não possui nenhuma substring de tamanho pelo menos K que é uma substring de S .

Ajude Nara a resolver o seguinte problema: dada uma string binária S , um inteiro K e Q strings binárias T_i , determine, para cada T_i , se ela **pertence** ou não a S .

Entrada

A primeira linha contém a string binária S ($1 \leq S \leq 10^6$).

A segunda linha contém o inteiro K ($1 \leq K \leq 20$).

A terceira linha contém a quantidade Q ($1 \leq Q \leq 10^6$) de strings T_i a serem analisadas.

Cada uma das próximas Q linhas contém uma string T_i ($1 \leq T_i \leq 10^6$).

É garantido que a soma do tamanho de todas as T_i não ultrapassa 10^6 . Isto é, $\sum_{i=1}^Q |T_i| \leq 10^6$.

Todas as strings são formadas apenas por 0s e 1s.

Saída

Para cada uma das Q strings T_i , imprima uma linha com 1 caso ela pertença a S e 0 caso contrário.

Exemplo

Entrada	Saída
111101010	1
3	0
2	
0101010	
00011	
111101	1
2	1
4	1
0101	0
0001	
11111111111111	
0000	
101000101	1
4	1
3	1
101000101	
010101	
0101	

Notas

O primeiro exemplo é o mesmo exemplo do enunciado.

No segundo exemplo, 01 é uma substring que faz T_1 e T_2 pertencerem a S e 111, por exemplo, é uma que faz T_3 pertencer. Como só existe 1 zero em S , não há nenhuma substring de T_4 para fazê-la pertencer a S .

Lembrando que uma substring de uma string S é um subsequência contígua de caracteres de S . Por exemplo, as substrings de $ABCD$ são: A , B , C , D , AB , BC , CD , ABC , BCD e $ABCD$. $S[l..r]$ representa a substring de S que começa na posição l e termina na posição r .

Problema O

O sapo não lava o pé

Limite de tempo: 1s

Autor: Guilherme Ramos

Em tempos de inteligências artificiais avançadas, ajude as máquinas a participarem das brincadeiras! Dado um trecho de música, mostre como cantá-lo usando somente uma vogal.

Entrada

A entrada consiste de uma linha contendo um trecho de música m com não mais de 100 caracteres, seguido de uma quantidade indeterminada de linhas contendo, cada uma, uma vogal específica para ser usada. A entrada termina com EOF.

Saída

Para cada vogal, apresente, em uma linha, o trecho da música alterado para usar apenas esta vogal.

Exemplo

Entrada	Saída
O sapo nao lava o pe	A sapa naa lava a pa
A	E sepe nee leve e pe
E	I sipi nii livi i pi
I	O sopo noo lovo o po
O	U supu nuu luvu u pu
U	
Nao lava porque nao quer.	Naa lava parqaa naa qaar.
a	Nii livi pirqii nii qiir.
i	Nuu luvu purquu nuu quur.
u	
Baby Shark, doo-doo, doo-doo	Baby Shark, daa-daa, daa-daa
A	Buby Shurk, duu-duu, duu-duu
u	

Notas

Pontuação (hifens, exclamações, interrogações, etc.) é válida, mas não há caracteres especiais (@, #, etc.).