

Seletiva UnB 2024

Caderno de Problemas

22 de junho de 2024



(Este caderno contém 15 problemas)

Comissão Organizadora:

Alberto Tavares Duarte Neto (UnB)
Bruno Ribas (UnB/FGA)
Daniel Saad Nogueira Nunes (IFB)
Daniel Porto (UnB)
Edson Alves da Costa Júnior (UnB/FGA)
Eduardo Quirino (UnB)
Guilherme Novaes Ramos (UnB)
José Leite (UnB)
Tiago de Souza Fernandes (UnB)
Vinicius Ruela Pereira Borges (UnB)
Wilson Guimarães (UnB)

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 – Ambição a cafeína

Limite de tempo: 3s
Limite de memória: 256MB

Autor: José Leite

Feiras de café são um evento divertido onde podemos encontrar entusiastas e também desfrutar de um bom café. Nestes eventos, é costumeiro haver estandes onde lojas se promovem e também servem uma porção de café em tamanho padrão de graça.

Seu grupo de amigos gosta de desafios e está determinado a tomar o maior número de porções possível. O i -ésimo amigo consegue consumir até a_i porções de café. O j -ésimo estande tem um estoque de b_i porções e também tem um limite de c_i de porções servidas a uma mesma pessoa.

Com sua grande expertise estratégica, você está responsável por bolar o plano que maximize o café consumido. Entretanto não sabemos ao certo todos os valores de b_i e c_i , então vamos processar Q atualizações p, x, y onde vamos atualizar $b_p := x$ e $c_p := y$.

Cabe a você, com suas excelentes habilidades em algoritmos e em café, fazer um programa que calcule o máximo de porções que o grupo consegue consumir após cada atualização.

Entrada

A primeira linha da entrada contém um inteiro t ($1 \leq t \leq 1000$), o número de casos de teste.

A primeira linha de cada caso de teste contém três inteiros N ($3 \leq N \leq 5 \cdot 10^4$), Q ($3 \leq Q \leq 5 \cdot 10^4$) e M ($3 \leq M \leq 5 \cdot 10^4$), respectivamente o tamanho do grupo de amigos e o número de estandes.

A próxima linha de cada caso de teste contém N inteiros a_i ($1 \leq a_i \leq 10^5$) — o número de porções que o i -ésimo amigo consegue tomar.

A próxima linha de cada caso de teste contém M inteiros b_i ($1 \leq b_i \leq 10^5$) — o número de porções que o i -ésimo estande consegue fornecer.

A próxima linha de cada caso de teste contém M inteiros c_i ($1 \leq c_i \leq 10^5$) — o limite de porções por pessoa que o i -ésimo estande fornece.

As próximas Q linhas de cada caso de teste contém três inteiros p ($1 \leq p \leq M$), x, y ($1 \leq x, y \leq 10^5$), indicando novos limites para o p -ésimo estande.

É garantido que, dentre todos os casos de teste, a soma de N não excede $5 \cdot 10^4$, a soma de M não excede $5 \cdot 10^4$ e a soma de Q não excede $5 \cdot 10^4$.

Saída

Para cada caso de teste, imprima Q linhas — o consumo máximo após cada atualização.

Exemplo

Entrada	Saída
1	14
5 5 5	15
1 5 2 4 3	15
3 3 3 3 3	14
1 1 1 1 1	15
2 2 1	
5 4 1	
2 2 2	
1 2 2	
4 4 3	

Problema B – Baguete

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Guilherme Ramos

A epítome da brazucagem é o bom e velho vira-lata caramelo. Este arauto do caos que domina as ruas é uma manifestação cultural imaterial do Brasil (talvez, reconhecido legalmente no PL 1897/2023). A verdade é que esses notórios caninos também podem ser muito amigáveis, mas os efeitos de um encontro dependem do humor (do cachorro) e da abordagem (do motobói).

Baguete é um exemplar típico de vira-lata caramelo. Adotada na rua quando filhote, ela superou a ausência de seus 8 irmãos (todos bem encaminhados a tutores responsáveis) para tornar-se a representante da classe numa quadra da Asa Norte. Por ser extremamente inteligente, ela desenvolveu uma técnica simples para indicar como se sente na hora de um encontro: solta um sonoro “au-au” quando está serelepe e um gutural “rrrrr” quando não quer papo.



Crie um programa que apresenta a resposta de um transeunte a interlocução da Baguete.

Entrada

A entrada consiste de um string indicando a opinião de Baguete, conforme o enunciado.

Saída

Apresente a mensagem “TMJ!” em caso de uma abordagem favorável, “Esta repreendida!” caso contrário.

Exemplo

Entrada	Saída
rrrrr	Esta repreendida!
au-au	TMJ!

Problema C – Coffee-Break Fit

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Vinicius Borges

Sabe-se que os coffee-breaks das competições de programação são sempre os melhores! Entretanto, nem todos podem saborear um enroladinho, uma coxinha ou um bolo de cenoura. Esse é o caso de N competidores que seguem uma dieta rigorosa e solicitaram um coffee-break *fit*, com frutas, castanhas, sucos verdes, ovos cozidos, batatas doces e etc.

Os competidores são identificados por inteiros de 1 a N . Sabe-se que cada competidor i solicitou a_i gramas de lanche *fit* à organização, que comprou inicialmente b_i gramas com base em uma estimativa aleatória. Além disso, cada competidor i informou que seu lanche *fit* custa c_i dinheiros por grama. O problema é que como pode faltar lanche para alguns competidores, a coordenação do evento quer complementar **integralmente** o lanche *fit* de cada competidor que se encaixe nessa situação.

Como a organização possui um orçamento limitado a M dinheiros, nem todos os competidores serão atendidos. Ela então escolhe uma permutação de X competidores distintos (possivelmente zero) p_1, \dots, p_X ($p_j \in \{1, 2, \dots, N\}$) e verifica com a fornecedora do coffee-break que o custo final (com a taxa de serviço) **por grama** do lanche *fit* para o competidor p_j será então $c_{p_j} \times (X - j + 1)$.

Ajude a organização da competição a determinar a maior quantidade possível de competidores que tiveram seus lanches **integralmente** complementados, sem exceder o orçamento de M dinheiros.

Entrada

A primeira linha da entrada contém dois números inteiros N e M ($1 \leq N \leq 2 \cdot 10^5, 0 \leq M \leq 10^9$) separados por espaço em branco, indicando a quantidade de competidores e o orçamento que a organização possui para gastar com o lanche *fit*, respectivamente.

A segunda linha da entrada contém N números inteiros a_1, a_2, \dots, a_N ($1 \leq a_i \leq 10^5$) separados por espaço, indicando a quantidade de lanche *fit* (em gramas) que cada competidor solicitou para a organizadora.

A terceira linha da entrada contém N números inteiros b_1, b_2, \dots, b_N ($1 \leq b_i \leq 10^5$) separados por espaço, em que b_i indica a quantidade de lanche *fit* (em gramas) comprada pela organização para o i -ésimo competidor.

A última linha da entrada contém N números inteiros c_1, c_2, \dots, c_N ($1 \leq c_i \leq 10^5$) separados por espaço, representando o custo por grama de lanche *fit* para o i -ésimo competidor.

Saída

Imprima um número inteiro com a resposta para o problema – a maior quantidade possível de competidores que terão seus lanches *fit* complementados pela organização sem que esta exceda o orçamento de M dinheiros.

Exemplo

Entrada	Saída
3 5	2
2 2 2	
1 1 1	
1 1 1	
3 6	1
3 3 4	
2 2 2	
5 1 3	
2 2	0
2 3	
3 4	
1 2	

Notas

No primeiro exemplo de teste, cada competidor necessita de 1g para complementar seu lanche *fit*. Para um orçamento igual a 5, pode-se custear integralmente os competidores 1 e 2 a um custo de 3, pois o custo final do lanche *fit* do competidor 1 será igual a 2 e do competidor 2 igual a 1.

No segundo exemplo de teste, os competidores 1 e 2 necessitam de 1 grama de complementação de lanche *fit*, enquanto que o competidor 3 requer 2 gramas. Apenas o competidor 2 será atendido pela organização, resultando em um custo final para complementar seu lanche *fit* igual a 1, pois a consideração de quaisquer outros competidores excede o orçamento $M = 6$ da organização.

No terceiro exemplo de teste, nenhum competidor precisará ter seu lanche *fit* complementado.

Problema D – dogsay

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Guilherme Ramos

Programadores de verdade usam a linha de comando. Mas a interação com esta ferramenta pode ser transformada numa experiência única para os maratonistas com o apoio de animais. E melhor que uma simpática vaquinha, só uma vira-lata caramelo!

dogsay é basicamente um filtro de texto: mande alguma mensagem e a receba de um caramelo.

```
_____  
< Baguete! >  
=====
```



Defina um programa que lê um string da entrada e o apresente em **dogsay**.



Entrada

A entrada consiste de uma linha com pelo menos 5 e não mais que 70 caracteres. É garantido que não há espaços no início ou ao final da mensagem.

Saída

A saída deve ser a mensagem lida, no formato de **dogsay**, ou seja, dentro de um balão de diálogo de um vira-lata caramelo.

Exemplo

Entrada	Saída
Au-au	<pre>----- < Au-au > =====</pre> 
A 5a Escola de Inverno vai bombar!	<pre>----- < A 5a Escola de Inverno vai bombar! > =====</pre> 

Notas

A saída pode ser simplificada aproveitando um dos exemplos abaixo para gerar a formatação.

Problema E – Estruturando o coffee-break

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Daniel Saad Nogueira Nunes

Vinicius é sempre o responsável pelo coffee-break e pela fotografia dos eventos de programação competitiva no Distrito Federal. Após várias Maratonas, ele desenvolveu TEPT (transtorno especial de pães e tortas). Se ele não organizar os alimentos de uma forma bem específica, a foto não fica do seu gosto.

No coffee-break, os alimentos ficam dispostos em uma sequência S , em que S_i representa o tipo do i -ésimo alimento. Se $S_i = S_j$ com $i \neq j$, temos uma repetição do mesmo alimento na mesa. Para evitar o estresse causado por seu TEPT, a técnica que Vinicius usa é a seguinte: organizar a sequência de alimentos em partes de forma que a sequência formada pelas partes seja não-crescente considerando a ordem lexicográfica. Em outras palavras, definir as partes X_i tais que $S = X_1, \dots, X_k$ e $X_1 \geq X_2 \geq \dots X_k$. Além disso, cada parte X_i tem que ser menor, lexicograficamente falando, do que qualquer subparte formada a partir de X_i retirando 1 ou mais alimentos do seu início.

Ajude Vinicius a organizar a mesa de modo que as fotos fiquem “jóia”.

Entrada

A entrada possui uma única linha com a sequência S de alimentos.

Restrições:

- $1 \leq |S| \leq 10^5$
- S é formada por símbolos sobre o alfabeto $\{a, \dots, z\}$

Saída

Dê como saída, em uma única linha, a estruturação do coffee-break por Vinicius. As partes devem estar separadas pelo símbolo `|`, que representa uma divisória na mesa de alimentos.

Exemplo

Entrada	Saída
banana	b an an a
ababaa	ab ab a a
abacaba	abac ab a

Problema F – Fenótipos

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Guilherme Ramos

Seu Gregório João Mendel gosta de biologia e de totós. Ele tem uma grande chácara onde cuida dos cães que resgata da rua. Para facilitar o processo de adoção responsável, cada animal é brevemente descrito por uma sigla, determinada pelas iniciais das características, considerando:

1. tamanho: $\{A, P, M, G, X\}$ para raças anã, pequena, média, grande ou gigante;
2. pelagem: $\{B, C, M, P\}$ para animais brancos, caramelos, malhados ou pretos;
3. forma da cabeça: $\{B, D, M\}$ para braquicefálico (focinho achatado e a cabeça grande), dolicocefálico (focinho alongado e a cabeça pequena) ou mesocefálico (proporções harmônicas);
4. forma do corpo: $\{L, B, M\}$ longilíneo (corpo mais comprido que largo), brevilíneo (corpo mais largo que comprido) ou mediolíneo (corpo harmonioso);
5. temperamento: $\{A, C, E, I, M, P\}$ agressivo, companheiro, energético, inseguro, manso, ou perseverante.

Toda sigla é determinada na ordem das características acima. Por exemplo, o melhor amigo do Seu Gregório é o Ervilha, um MMMMM (médio, malhado, mesocefálico, mediolíneo e manso). Já o Cassulinha, buldogue do vizinho, é um GBBBA (grande, branco, braquicefálico, brevilíneo e agressivo).

Seu Gregório aproveita a cachorrada para analisar as características por seu prisma favorito, o biológico! Ele acompanha os fenótipos dos filhotes em relação aos dos pais e sempre fica maravilhado em ver como eles passam entre as gerações. Ele também usa essa informação para tentar rastrear como cresce a população de cães, vendo se algum novo cachorro pode estar relacionado a uma matilha conhecida. Ajude-o nesta tarefa!

Entrada

A entrada consiste de uma linha com três siglas, separadas por espaço. As duas primeiras descrevem os pais do cão descrito pela terceira.

Saída

Caso o filhote tenha as características de um dos pais, apresente a mensagem “Herdou!”. Caso haja alguma inconsistência, apresente a mensagem “Orra meu...”.

Exemplo

Entrada	Saída
MMMBE GCDMM MMMMM	Herdou!
MMMBE GCDMM MCMME	Herdou!
GMBBC XBMM PBBBA	Orra meu...

Notas

No primeiro caso, o Ervilha obteve seu porte médio e cor malhada do primeiro parente, e o corpo curto e a mansidão do segundo – ambos os pais são mesocefálicos. No segundo caso, a Baguete (irmã do Ervilha) herdou tamanho, forma da cabeça e temperamento do primeiro parente, pelagem e forma do corpo do segundo. No terceiro exemplo, parece que nenhum dos parentes do Cassulinha tem o tamanho certo...

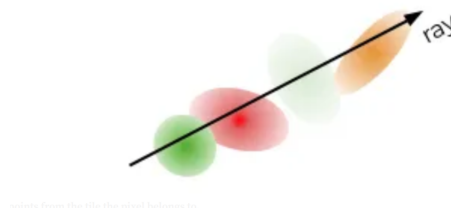
Problema G – Gaussian Splatting

Limite de tempo: 4s
Limite de memória: 256MB

Autor: Tiago de Souza Fernandes

Gaussian Splatting é a técnica mais quente do momento para renderização 3D, capaz de criar cenas realistas a partir de fotos ou vídeos. A ideia é simples: uma cena é gerada utilizando milhares de funções gaussianas, cada uma com posição, forma, cor e opacidade otimizados para representar fielmente a geometria e a aparência dos objetos reais. Pronto! Agora você tem uma versão tridimensional da cena capturada em fotos ou vídeo. Para renderizar a cena, ou seja, transformar em uma imagem exibida na tela, é necessário sobrepor as funções gaussianas daquele campo de visão.

Gaussian Splatting



Pedro Gallo, fascinado com essa técnica, começou a se questionar: qual será a maior quantidade de funções gaussianas que serão sobrepostas durante a renderização de um campo de visão qualquer? Para simplificar, ele decidiu resolver uma versão 2D desse problema e, como engenheiro, aproximou as funções gaussianas como segmentos de reta 2D. Dadas n funções gaussianas representadas por segmentos de reta, encontre a maior quantidade de sobreposições para uma reta qualquer do campo de visão.

Entrada

A primeira de entrada contém um inteiro n ($1 \leq n \leq 2 \cdot 10^3$). Cada uma das próximas n linhas da entrada contém 4 inteiros x_1, y_1, x_2 e y_2 ($0 \leq x_1, y_1, x_2, y_2 \leq 10^7$), representando as coordenadas dos dois pontos extremos do segmento de reta.

Saída

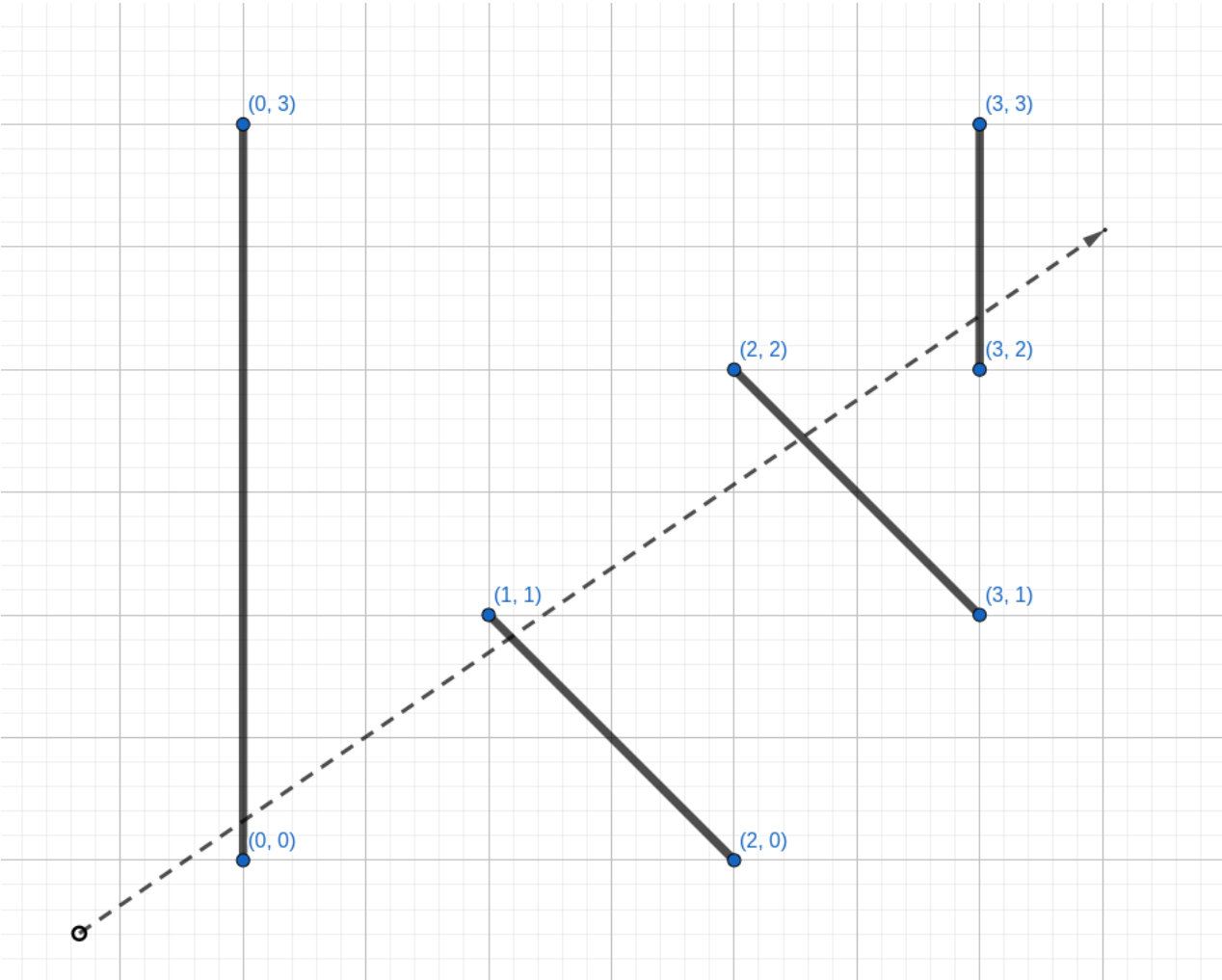
Imprima um único inteiro, a quantidade máxima de funções gaussianas sobrepostas por uma reta do campo de visão.

Exemplo

Entrada	Saída
4	4
0 0 0 3	
1 1 2 0	
2 2 3 1	
3 2 3 3	
4	3
1 1 2 2	
3 3 4 4	
5 5 6 6	
1 2 2 3	
1	1
0 0 10000000 10000000	

Notas

No primeiro caso de teste, é possível escolher um raio do campo de visão que sobreponha todos os quatro segmentos.



No segundo caso de teste, é impossível escolher uma reta que passe por mais de três segmentos.

Problema H – Hurricane!

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Daniel Porto

No universo dos jogos de cartas, existe o lendário *Hurricane!* em que dois jogadores duelam. Cada um tem seu próprio baralho de cartas mágicas e de monstros, e vence quem montar o exército de monstros mais forte!

Inicialmente, as cartas são embaralhadas e inicia quem tiver mais sorte no lançamento de uma moeda. Os jogadores se alternam acrescentando, uma de cada vez, cartas ao seus respectivos exércitos. Cada carta contém um monstro com o valor de seu ataque, e vence quem tiver a maior soma de ataques considerando todos os monstros no seu campo de batalha.

No entanto, existe a carta mágica que pode mudar o rumo da partida: a *Hurricane!*. Essa carta não aumenta o poder de seu exército, pelo contrário, seu efeito é remover do exército oponente o último monstro invocado, se isso for possível.

Determine quem será o vencedor da partida considerando as cartas sacadas por cada jogador.

Entrada

A entrada consiste em uma sequência de 10 números separados por espaços: $J1_1 J2_1 J1_2 J2_2 J1_3 J2_3 J1_4 J2_4 J1_5 J2_5$. $J1_n$ representa a n -ésima carta sacada do baralho pelo jogador que inicia o jogo (Jogador 1), enquanto $J2_n$ representa a n -ésima carta sacada do baralho pelo oponente (Jogador 2).

Se a carta for um monstro, o valor informado na entrada é um inteiro entre 1 e 20. Se a carta for do tipo *Hurricane!*, seu valor é -1 .

Saída

Apresente o número do jogador vencedor da partida, ou **0** em caso de empate.

Exemplo

Entrada	Saída
5 9 16 20 -1 2 18 14 7 10	1
9 18 19 20 -1 -1 -1 7 2 4	0

Notas

No caso do Exemplo 1, temos que a soma do ataque dos monstros do Jogador 1 é 46 ($5 + 16 + 18 + 7$). A soma do ataque dos monstros do Jogador 2 é 35 ($9 + 2 + 14 + 10$) pois o monstro de ataque 20 foi removida do jogo pela carta *Hurricane!* do Jogador 1.

Problema I – Inseticida

Limite de tempo: 2s
Limite de memória: 256MB

Autor: Edson Alves

O professor I. Novaes foi contratado para desenvolver o inseticida perfeito. Usando toda a sua genialidade e rigor matemático, o professor chegou em um inseticida com um comportamento eficaz, porém bastante curioso.

Quando aplicado em um único inseto, ele o elimina com 100% de eficácia. Se aplicado em um exame de N insetos, ele reduz em k vezes a quantidade de insetos, onde k é um divisor de N escolhido aleatoriamente. Por exemplo, se $N = 6$, em 25% das aplicações ele reduz o exame para $6/2 = 3$ insetos; em 25% ele reduz para $6/3 = 2$ insetos; em 25% ele reduz para $6/6 = 1$ inseto e, nas demais, ele não elimina nenhum inseto, pois $6/1 = 6$.

A equipe de marketing está preocupada com a aceitação do produto por parte do público, que pode entender a necessidade de múltiplas aplicações como desperdício. Para isso, ela o contratou para escrever um programa que compute o valor esperado do número de aplicações para a eliminação de um exame de N insetos.

Entrada

A entrada contém um único número N ($1 \leq N \leq 10^{14}$), que indica o número de insetos no exame.

Saída

Imprima, em uma linha, o valor esperado do número de aplicações do inseticida para a completa eliminação do enxame. Caso sua resposta seja x e a resposta do juiz seja y , ela será considerada certa se $\frac{|x-y|}{\max\{1, |y|\}} < 10^{-6}$.

Exemplo

Entrada	Saída
2	3.0000000000
9	3.5000000000
12	4.0333333333
26962228293000	6.2549896957

Notas

No primeiro caso, há 50% de chances de eliminar todos os insetos com duas aplicações: na primeira, o enxame é reduzido para $2/2 = 1$ inseto, e a aplicação seguinte o elimina. Nos casos onde a primeira eliminação não reduz nenhum inseto, há 50% de chances de eliminá-los com 3 aplicações, ou seja, ele elimina com três aplicações em $(50\%) \times (50\%) = 25\%$ das vezes. Assim, o valor esperado do número de aplicações é dado por

$$2 \times \frac{1}{2} + 3 \times \frac{1}{4} + 4 \times \frac{1}{8} + \dots = 3$$

Problema J – Jornada à Fortaleza

Limite de tempo: 3s
Limite de memória: 256MB

Autor: José Leite

Fernanda tinha a tradição de viajar de carro de Brasília à Fortaleza em todas as férias quando criança.

Agora que cresceu, quer fazer novamente esta jornada, mas ela vai dirigir e pode escolher o caminho. Para tornar a viagem mais interessante, Fernanda teve a ideia de passar por cidades e estradas totalmente diferentes na ida e na volta, exceto por Brasília e Fortaleza. Assim, sempre vai descobrir novos lugares.

Para uma cidade c que não seja Brasília ou Fortaleza, temos três possibilidades:

- c não é visitada;
- c é visitada somente na viagem de ida de Brasília a Fortaleza;
- c é visitada somente na viagem de volta de Fortaleza a Brasília.

Dentro destas restrições, ela ainda quer passar o menor tempo possível em trânsito entre as cidades para sobrar mais tempo para curtir a viagem. Ajude-a a calcular as rotas de ida e volta de forma a minimizar o tempo total em deslocamento.

Entrada

A primeira linha da entrada contém um inteiro t ($1 \leq t \leq 1000$), o número de casos de teste.

A primeira linha de cada caso de teste contém dois inteiros N ($3 \leq N \leq 10^5$) e M ($N \leq M \leq 3 \cdot 10^5$), respectivamente os números de cidades e de estradas. Brasília é a cidade 1 e Fortaleza a cidade N .

As próximas M linhas de cada caso de teste contém três inteiros u, v ($1 \leq u, v \leq N, u \neq v$) e w ($1 \leq w \leq 10^9$), indicando uma estrada de mão dupla entre as cidades u e v cuja distância é percorrida em w unidades de tempo. É garantido que não haverá duas estradas distintas entre as mesmas duas cidades.

Também é garantido que, entre todos os casos de teste, a soma de N não excede $2 \cdot 10^5$ e a soma de M não excede $5 \cdot 10^5$.

Saída

Para cada caso de teste, caso não seja possível criar um plano de viagem seguindo as restrições, imprima -1.

Caso contrário, apresente três informações, uma por linha. Na primeira linha, mostre três inteiros d, a e b , separados por espaço, indicando respectivamente o tempo total de viagem, o número de cidades visitadas na ida e o número de cidades visitadas na volta.

Na próxima linha, imprima a números distintos, representando a rota de ida à Fortaleza.

Na próxima linha, imprima b números distintos, representando a rota de volta a Brasília.

Caso hajam vários pares de rotas que minimizam o tempo total em deslocamento, imprima qualquer um.

Exemplo

Entrada	Saída
2	13 5 2
5 5	1 2 3 4 5
1 2 1	1 5
1 5 2	12 4 4
2 3 2	1 2 6 7
3 4 2	1 3 5 7
4 5 6	
7 8	
1 2 1	
2 5 2	
5 7 1	
1 3 3	
3 6 3	
6 7 3	
2 6 2	
3 5 2	

Problema K – K-ésimo Kara Kickado

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Alberto Neto

José e seus n amigos estão jogando Sounter Ctrike, um jogo de computador de vários jogadores. Para moderar os servidores, o jogo conta com um sistema de banimento. José, querendo pregar uma peça em seus amigos, decide banir os outros n jogadores conforme as seguintes regras:

- José lê a lista de jogadores (que não o inclui) da esquerda para a direita e vai banindo um a um. Quando chega ao final, ele volta para o início da lista e continua os banimentos.
- Ao ser banido, o jogador é removido da lista de jogadores do servidor imediatamente, e a ordem dos jogadores restantes não muda.
- Após m jogadores serem banidos consecutivamente, o próximo jogador não é banido. Os próximos m jogadores serão banidos, e o próximo não, e assim por diante.
- O banimento finaliza após exatamente k jogadores serem banidos.

Como quem ri por último ri melhor, diga o índice do último jogador a ser banido, para que todos possam rir dele. Como José é o dono do servidor, seu índice é 0; os outros jogadores são indexados de 1 até n .

Entrada

A primeira e única linha de entrada contém três inteiros n , m e k ($1 \leq n, m \leq 10^{14}$, $1 \leq k \leq n$) — a quantidade de amigos de José, quantos jogadores são banidos até um não ser banido, e a quantidade de jogadores banidos, respectivamente.

Saída

Imprima um único inteiro — o índice do último amigo banido.

Exemplo

Entrada	Saída
8 1 5	2
5 2 3	4
100 1000000 89	89
31415926535897 144 12345678912345	12431412793680

Notas

No primeiro caso de teste, os jogadores banidos são os de índice: 1, 3, 5, 7, 2. Logo, a resposta é 2.

No segundo caso de teste, os jogadores banidos são os de índice: 1, 2, 4. Logo, a resposta é 4.

Problema L – Letra aleatória

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Daniel Saad Nogueira Nunes

Uma fonte de ruído desconhecida atrapalhou a comunicação por e-mail dos professores do *campus* Darcy Ribeiro com os da Faculdade do Gama. Essa fonte embaralhava a palavra original e inseria uma letra aleatória, em qualquer posição da palavra embaralhada.

Descubra qual a letra extra inserida e ajude os professores a organizarem a Seletiva UnB 2024.

Entrada

A primeira linha da entrada possui um inteiro N , indicando o número de mensagens trocadas entre os professores.

Cada uma das próximas N linhas possui duas strings, S_1 e S_2 , separadas por um espaço. S_1 é a mensagem enviada originalmente e S_2 é S_1 embaralhada e com o símbolo extra.

Restrições:

- $1 \leq |N| \leq 1000$
- $1 \leq |S_1| \leq 30$
- $2 \leq |S_2| \leq 31$
- Os caracteres de S_1 e S_2 estão sobre o alfabeto $\Sigma = \{a, b, \dots, z\}$

Saída

Imprima uma linha com a letra aleatória inserida pela fonte de ruído.

Exemplo

Entrada	Saída
4	o
porta portao	m
problema bolrapmme	a
abracadabra rbaraaadcbaa	e
balao aealbo	

Notas

No primeiro exemplo, a letra o foi inserida ao final da palavra porta. No segundo exemplo, a letra m foi inserida na palavra **problema** embaralhada. No terceiro exemplo, a letra a foi inserida na palavra **abracadabra** embaralhada. No último exemplo, a letra e foi inserida na palavra **balao** embaralhada.

Problema M – Máxima Sintonia

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Edson Alves

Após a análise exaustivas dos dados das participações anteriores da UnB na Maratona de Programação, o professor Ramos chegou a conclusão que, para obter um trio com desempenho ótimo, todos os membros deveriam ser selecionados por meio do coeficiente de sintonia σ entre eles.

Cada participante tem um nível de performance p e o coeficiente de sintonia entre os competidores A, B e C é dado pelo maior divisor comum entre p_A, p_B e p_C , isto é, $\sigma(A, B, C) = (p_A, p_B, p_C)$. Por exemplo, se os estudantes A, B, C, D e E tem níveis de performance iguais a 6, 10, 14, 15 e 21, respectivamente, então $\sigma(A, B, C) = (6, 10, 14) = 2$ e $\sigma(A, D, E) = (6, 15, 21) = 3$.

Sabendo que os competidores são identificados por inteiros distintos entre 1 e N , e dados os coeficiente de performance de cada um deles, auxilie o professor Ramos determinando uma equipe formada por três competidores distintos tal que o coeficiente de sintonia entre eles é o maior possível.

Entrada

A primeira linha da entrada contém o número de competidores N ($3 \leq N \leq 10^5$);

A segunda linha contém N inteiros p_i ($1 \leq p_i \leq 10^6, 1 \leq i \leq N$), separados por um espaço em branco, informando o nível de performance p_i do competidor i .

Saída

Imprima, em uma linha, os três identificadores da equipe que maximiza o nível de sintonia, separados por um espaço em branco. Se existem duas ou mais equipes distintas com coeficiente de sintonia máximo, escolha qualquer uma delas.

Exemplo

Entrada	Saída
5	1 4 5
6 10 14 15 21	
3	2 3 1
300 100 200	
7	1 2 3
2 3 5 7 11 13 17	

Notas

No primeiro caso, conforme ilustrado no texto, a equipe formada pelos participantes 1, 4 e 5 tem sintonia igual a 3, a maior possível dentre as equipes que podem ser formadas a partir destes 5 competidores.

No segundo caso é possível formar uma única equipe, com coeficiente de sintonia igual a $\sigma(1, 2, 3) = (300, 100, 200) = 100$. Observe que a ordem dos identificadores não importa.

No terceiro caso, qualquer uma das 35 equipes que podem ser formadas tem coeficiente de sintonia igual a 1.

Problema N – Nürburgring

Limite de tempo: 2s
Limite de memória: 256MB

Autor: Daniel Porto

Nürburgring é um autódromo na cidade de Nürburg na Alemanha também conhecido por “Inferno verde”, pela sua dificuldade, número de acidentes e por estar rodeado de uma floresta.

Uma equipe de Fórmula 1 resolveu escolher esse autódromo para fazer alguns testes para aprimorar a versão do carro da próxima temporada. Acontece que o engenheiro chefe percebeu que a evolução do carro seguia uma regra: quanto mais voltas o piloto de testes dava com o carro, mais ajustes eram feitos e mais eram as chances de conseguir um tempo menor na volta seguinte. Ele também percebeu que essa evolução seguia uma fórmula matemática e que ele poderia calcular o tempo da n -ésima volta.

A fórmula encontrada pelo engenheiro chefe é:

$$T_i = \begin{cases} x, & i = 1 \\ x - M_2, & i = 2 \\ \lfloor \frac{T_{i-1} + T_{i-2}}{2} \rfloor - M_i, & i > 2 \end{cases}$$

em que T_i é o tempo para completar a i -ésima volta, x é o tempo da volta inicial e M_i é o tempo economizado pelo ajuste realizado para a volta i . A melhora a cada ajuste M_i é calculada por:

$$M_i = (3 \cdot i) \mod 10$$

Por conta da precisão dos medidores, o tempo de cada volta sempre é contabilizado em milissegundos. Qualquer valor menor que 1ms em uma volta pode ser desconsiderado.

Calcule o tempo que o piloto de testes fará a volta solicitada a partir do tempo da volta inicial informada.

Entrada

A entrada consiste de dois números n e x , separados por espaço, onde n é o número de voltas dadas ($0 < n \leq 1000$), e x é o tempo da volta inicial em milissegundos ($7 * 10^4 \leq x \leq 10^5$).

Saída

Apresente o tempo, em milissegundos, da n -ésima volta solicitada.

Exemplo

Entrada	Saída
3 98969	98957
5 88633	88616

Notas

O operador `mod` no cálculo de M_i é de módulo, que fornece o resultado da divisão inteira de um número por outro.

No caso do Exemplo 1, temos que M_2 é calculada como 6, portanto $T_2 = 98963$. Como M_3 é 9 e a média entre T_1 e T_2 é 98966 tem-se que $T_3 = 98957$.

Problema O – O Teorema do Macaco infinito

Limite de tempo: 1s
Limite de memória: 256MB

Autor: José Leite

Pelo Teorema do Macaco Infinito, um macaco digitando teclas aleatórias em um teclado por um intervalo de tempo infinito irá quase certamente criar um texto qualquer escolhido, como, por exemplo, a obra completa de William Shakespeare.

Gustavo Leal quer estimar quanto tempo levará para o macaco digitar alguns textos. Considerando que o macaco digita uma das 26 letras minúsculas com mesma probabilidade a cada segundo, Ajude Gustavo e calcule o tempo esperado para o macaco digite o texto.

Podemos provar que o tempo esperado pode ser expresso por uma fração $\frac{p}{q}$, onde p e q são inteiros positivos com q coprimo com 998244353. Imprima $p \cdot q^{-1} \bmod 998244353$, onde q^{-1} é o inverso multiplicativo de q módulo 998244353.

Entrada

A primeira linha da entrada contém um único inteiro t ($1 \leq t \leq 500$) — o número de casos de teste.

Cada caso de teste consiste em uma linha com uma string s ($1 \leq |s| \leq 5000$) — o texto desejado. É garantido que a string s contém apenas letras minúsculas.

É garantido que a soma dos tamanhos das strings é no máximo 5000.

Saída

Para cada caso de teste, imprima uma única linha contendo $p \cdot q^{-1} \bmod 998244353$ — o tempo esperado para o macaco digitar o texto.

Exemplo

Entrada	Saída
6	26
z	18278
aaa	17602
ada	45872954
abacaba	52643009
gomonkeys	419129909
afternacasadoquirino	

Notas

Nos três primeiros testes, a resposta é um inteiro ($q = 1$).