acm International Collegiate Programming Contest 2017 IBM event sponsor

Maratona de Programação da SBC 2017

Sub-Regional Brasil do ACM ICPC

9 de Setembro de 2017

Caderno de Problemas

Informações Gerais

Este caderno contém 13 problemas; as páginas estão numeradas de 1 a 18, não contando esta página de rosto. Verifique se o caderno está completo.

A) Sobre os nomes dos programas
1) Sua solução deve ser chamada *codigo_de_problema*`.c`, *codigo_de_problema*`.cpp`, *codigo_de_problema*`.pas`, *codigo_de_problema*`.java` ou *codigo_de_problema*`.py`, 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.

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:

SBC Sociedade Brasileira de Computação

v1.0

Problema A

Acordes intergaláticos

A maratona de composição de sonatas para piano intergalático está tentando dificultar a vida dos competidores, pois cada vez mais seres de inteligência superior estão participando. O piano é composto de N teclas, numeradas de 0 a $N-1$. O sistema tonal intergalático possui 9 notas musicais, com valores de 0 a 8. Inicialmente todas as teclas do piano estão associadas à mesma nota 1. O competidor vai tocar uma sequência de acordes. Cada acorde intergalático é composto por duas teclas distintas, a e b, $0 \leq a < b < N$. Quando o acorde é tocado, o piano vai emitir a nota mais frequente, f, entre todas as teclas do intervalo $[a, b]$. Se houver mais de uma nota mais frequente, ele emite a maior delas. Imediatamente após emitir a nota, o piano muda a nota associada a todas as teclas do intervalo $[a, b]$. A nova nota associada à tecla k, $a \leq k \leq b$, será a anterior mais f, módulo 9.

Por exemplo, se em determinado momento as notas associadas a um piano de $N = 15$ teclas são

teclas	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
notas	2	2	1	4	5	4	3	4	8	0	1	6	2	0	1

e o acorde $[3, 9]$ é tocado, então a nota mais frequente será 4 e as novas notas após o acorde serão:

teclas	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
notas	2	2	1	8	0	8	7	8	3	4	1	6	2	0	1

Dada a sequência de Q acordes, seu programa deve imprimir as notas que estarão associadas às teclas do piano após todos os acordes da sequência terem sido tocados.

Entrada

A primeira linha da entrada contém dois inteiros, N ($2 \leq N \leq 100000$), e Q ($1 \leq Q \leq 100000$), respectivamente o número de teclas do piano intergalático e a quantidade de acordes. As Q linhas seguintes contêm, cada uma, dois inteiros A e B, ($0 \leq A < B < N$), representando um acorde.

Saída

Seu programa deve imprimir N inteiros, um por linha, representando as notas associadas às teclas do piano, após todos os acordes terem sido tocados.

Exemplos

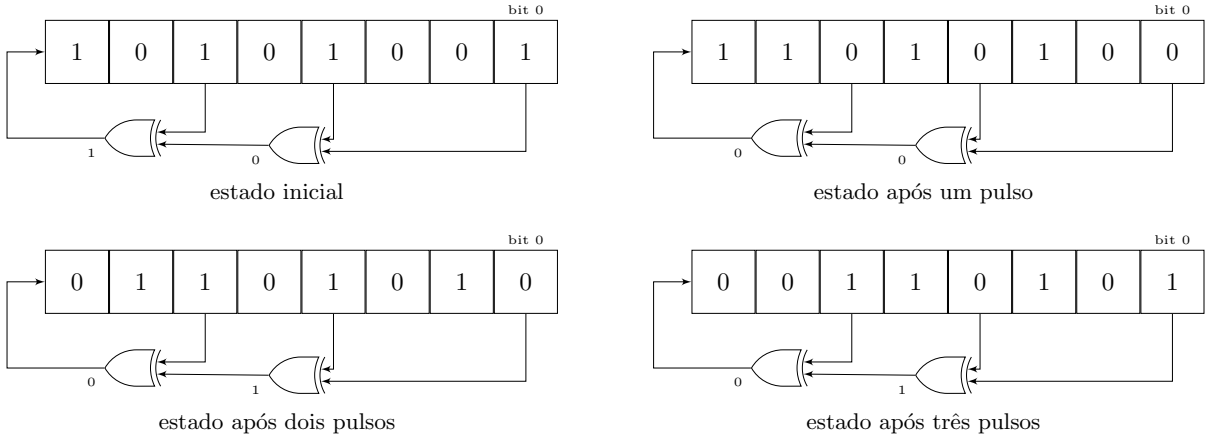
Exemplo de entrada 1	**Exemplo de saída 1**
5 3 1 2 0 4 0 2	5 6 6 2 2

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

Problema B
Brincadeira

Um Registrador de Deslocamento é um circuito que desloca de uma posição os elementos de um vetor de bits. O registrador de deslocamento tem uma entrada (um bit) e uma saída (também um bit), e é comandado por um pulso de relógio. Quando o pulso ocorre, o bit de entrada se transforma no bit mais significativo do vetor, o bit menos significativo é jogado na saída do registrador, e todos os outros bits são deslocados de uma posição em direção ao bit menos significativo do vetor (em direção à saída).

Um Registrador de Deslocamento com Retroalimentação Linear (em inglês, LFSR) é um registrador de deslocamento no qual o bit de entrada é determinado pelo valor do ou-exclusivo de alguns dos bits do registrador antes do pulso de relógio. Os bits que são utilizados na retroalimentação do registrador são chamados de torneiras. A figura abaixo mostra um LFSR de 8 bits, com três torneiras (bits 0, 3 e 5).



Durante uma competição de programação, enquanto aguardam a divulgação do resultado final, Ricardo e Cláudio se divertem com um LFSR que encontraram no local.

Eles usam o LFSR para gerar uma sequência infinita de números. Para gerar tal sequência, antes de cada pulso do relógio, os bits do registrador são convertidos para decimal. Assim, para um LFSR como o da figura os primeiros elementos da sequência são: $A_0 = 169$ (10101001), $A_1 = 212$ (11010100), $A_2 = 106$ (01101010), $A_3 = 53$ (00110101) e $A_4 = 26$ (00011010). Note que o valor dos bits antes do primeiro pulso é o primeiro elemento da sequência.

Em cada rodada da brincadeira um deles fala dois números inteiros, X e Y. Daí em diante o outro deve encontrar uma subsequência contígua, de tamanho maior ou igual a Y, dos elementos da sequência gerada pelo LFSR, de modo que a soma dos elementos da subsequência contígua seja divisível por X.

De alguma forma os dois são capazes de se divertir com isso e encontrar as respostas mesmo sem a ajuda de um computador. E você, dada a descrição de um LSFR e dois inteiros X e Y, é capaz de encontrar uma subsequência válida (ou informar caso não exista uma)?

Entrada

A primeira linha contém cinco números inteiros N, T, A_0, X e Y. O inteiro N representa o número de bits ($2 \leq N \leq 30$), T é o número de torneiras ($1 \leq T \leq N$), A_0 é a representação decimal do estado inicial do LFSR, X o valor pelo qual a soma da subsequência contígua deve ser divisível ($1 \leq X \leq 10^6$) e Y é a quantidade mínima de elementos na subsequência contígua desejada ($1 \leq Y \leq 10^6$). Os bits são identificados por inteiros de 0 (bit menos significativo) a $N-1$ (bit mais significativo). A segunda linha

contém T inteiros, separados por espaços, representando os identificadores dos bits que são torneiras, em ordem crescente. O bit 0 sempre é uma torneira.

Saída

Seu programa deve imprimir, em uma única linha, dois inteiros I e F, representando os índices do primeiro e do último elementos da subsequência contígua escolhida. Caso não exista uma solução imprima a palavra `impossivel`. Caso exista mais de uma solução possível escolha aquela que minimiza o valor de F. Se mesmo assim houver mais de uma possibilidade opte por aquela que minimiza o valor de I.

Exemplo de entrada 1	**Exemplo de saída 1**
`8 3 169 169 1` `0 3 5`	`0 0`

Exemplo de entrada 2	**Exemplo de saída 2**
`8 3 169 238 2` `0 3 5`	`13 25`

Problema C

Cigarras periódicas

As "cigarras periódicas" americanas têm o ciclo de vida mais longo de todos os insetos conhecidos. A cada 17 anos, estas cigarras periódicas amadurecem, se acasalam, depositam ovos e morrem. Suas crias se refugiam debaixo da terra, a 20 centímetros de profundidade, onde elas se alimentarão da seiva de raízes por 17 anos, até que chegue seu dia de buscar um lugar ao sol.

Acredita-se que esse número não aconteceu por acaso, outras espécies de cigarras da região tem ciclos de 13 anos, assim essas duas espécies emergem ao mesmo tempo apenas a cada 221 anos. Isso é desejável pois dessa forma a chance de que as duas espécies se misturem diminui consideravelmente e características indesejáveis de uma população não são introduzidas na outra.

Inspirado por esse fenômeno, uma nova variação de algoritmo evolutivo foi criada. Na última etapa desse algoritmo as melhores possíveis soluções são divididas em populações de modo que cada população i tem um ciclo de vida C_i. Além disso uma população extra também é adicionada, de modo que a quantidade de iterações até que o ciclo de vida de todas as populações coincida seja a maior possível. Essas populações são então avaliadas até que o ciclo de vida de todas coincida e a melhor solução ao final do processo é escolhida. Como não é interessante esperar demais até que o algoritmo gere uma resposta, um limite superior L no número de iterações também deve ser respeitado.

Dados os ciclos de vida das populações criadas e o limite na quantidade de iterações L, sua tarefa é computar qual o período ótimo para a população extra que será adicionada.

Entrada

A primeira linha da entrada contém dois inteiros N e L, respectivamente, a quantidade de populações geradas pelas etapas anteriores do algoritmo e o limite da quantidade de iterações, $2 \leq N \leq 10^4$, $1 \leq L \leq 10^6$. A linha seguinte contém os N valores C_i representando a quantidade de iterações no ciclo de vida de cada população, onde $1 \leq C_i$. Você pode assumir que os ciclos de vida das populações atuais coincidem em menos de L iterações.

Saída

Seu programa deve produzir uma única linha com um inteiro representando o período da população extra que maximiza a quantidade T de iterações até que os ciclos de vida de todas as populações coincidam, respeitando a restrição de que $T \leq L$. Caso exista mais de um valor possível imprima o menor deles.

Exemplo de entrada 1	**Exemplo de saída 1**
2 5000 105 55	4

Exemplo de entrada 2	**Exemplo de saída 2**
2 512 3 14	72

Exemplo de entrada 3	**Exemplo de saída 3**
3 80 6 10 15	4

Exemplo de entrada 4	**Exemplo de saída 4**
3 60 12 10 15	1

Problema D

Despojados

Todo inteiro positivo pode ser escrito como um produto de potências de primos. Por exemplo, $252 = 2^2 \times 3^2 \times 7$. Um inteiro é *despojado* se pode ser escrito como um produto de dois ou mais primos distintos, sem repetição. Por exemplo, $6 = 2 \times 3$ e $14 = 2 \times 7$ são despojados, mas $28 = 2^2 \times 7$, 1, 17 não são despojados.

Entrada

A entrada consiste de uma única linha que contém um inteiro N ($1 \leq N \leq 10^{12}$).

Saída

Seu programa deve produzir uma única linha com um inteiro representando o número de divisores despojados de N.

Exemplo de entrada 1	**Exemplo de saída 1**
252	4

Exemplo de entrada 2	**Exemplo de saída 2**
6469693230	1013

Exemplo de entrada 3	**Exemplo de saída 3**
8	0

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

Exemplo de entrada 5	**Exemplo de saída 5**
88290298627	0

Problema E

Escala musical

As notas musicais são as unidades mais básicas da composição musical no ocidente. Muitas pessoas acreditam que existem apenas 7 notas musicais:

dó ré mi fá sol lá si

Chamaremos essas notas de notas *elementares*. Na verdade, existem notas além destas acima, normalmente identificadas pelo nome de uma das notas acima seguido do símbolo sustenido (#):

dó dó# ré ré# mi fá fá# sol sol# lá lá# si

Assim, existem 12 notas musicais básicas distintas. Entretanto, a rigor, esta sequência é infinita e periódica: após um "si" existe um outro "dó", e a sequência se repete novamente.

As notas elementares são mais conhecidas, por estarem em um tom musical conhecido como "dó maior". Em qualquer tom "maior", as distâncias entre as possíveis notas seguem um padrão. No tom "dó maior", por exemplo:

Nota	dó	ré	mi	fá	sol	lá	si
Intervalo para a próxima nota	2	2	1	2	2	2	1

Note que eu poderia usar qualquer "dó" na escala de "dó maior", pois a nota seguinte ao "si" será, novamente, um "dó". O mesmo vale para as demais notas. Um outro exemplo de notas em um determinado tom maior seria a escala de "dó# maior":

Nota	dó#	ré#	fá	fá#	sol#	lá#	dó
Intervalo para a próxima nota	2	2	1	2	2	2	1

Guilherme está aprendendo a tocar um teclado com 61 teclas, numeradas de 1 a 61. Assim, a nota 1 corresponde a um "dó", a nota 2 corresponde a um "dó #" e assim por diante, até chegar nas notas 60 (um "si") e 61 (um "dó").

Acredita-se que as músicas com as melhores melodias são aquelas que estão em algum tom maior, ou seja, músicas em que todas as notas pertencem à escala de algum tom maior. Enquanto pratica no teclado, Guilherme usa um aparelho que grava todas as notas tocadas durante a música. Para ajudá-lo a melhorar sua técnica você decidiu criar um programa capaz de avaliar as músicas gravadas por ele e determinar se elas estão em algum tom maior ou não.

Entrada

A primeira linha da entrada terá um número inteiro N, com $1 \leq N \leq 10^5$, correspondente ao número de notas musicais da música. Em seguida, serão fornecidos N números, um por linha, todos entre 1 e 61, inclusive, correspondendo às notas musicais.

Saída

Seu programa deve verificar se a música está em algum tom maior. Em caso afirmativo, seu programa deve imprimir uma única linha com o tom maior **(sem acentos)** em que a música está. Caso contrário, seu programa deve imprimir uma linha contendo a palavra `desafinado`. Caso a música possa estar em mais de um tom maior imprima aquele relativo a menor nota musical básica, sendo que "`do`" < "`do#`" < "`re`", ...

Exemplo de entrada 1	**Exemplo de saída 1**
8 1 3 5 6 8 10 12 13	do

Exemplo de entrada 2	**Exemplo de saída 2**
10 8 11 21 16 11 8 27 57 27 21	re#

Exemplo de entrada 3	**Exemplo de saída 3**
7 2 2 4 3 12 12 3	desafinado

Problema F

Fase

Em diversas competições acadêmicas, como a Olimpíada Brasileira de Informática (OBI), uma certa quantidade de competidores se classifica de uma fase para a fase seguinte, garantindo uma das vagas disponíveis. Entretanto, normalmente essa quantidade é variável, pois dada uma certa quantidade mínima de classificados, é frequente que haja empate na última vaga de classificação. Neste caso, é comum que todos os competidores empatados na última colocação se classifiquem.

Sua tarefa é ajuda a calcular o número de competidores classificados para a próxima fase. Você receberá uma lista de pontuações obtidas pelos competidores e o número mínimo de vagas para a fase seguinte e você deve decidir quantos competidores de fato vão se classificar.

Entrada

A primeira linha da entrada contém um número inteiro N, $1 \leq N \leq 1000$, representando o número de competidores. A segunda linha conterá um inteiro K, $1 \leq K \leq N$, indicando o número mínimo de competidores que devem se classificar para a próxima fase. Em seguida, N linhas conterão, cada uma um número entre 1 e 1000, inclusive, correspondente à pontuação de um competidor.

Saída

Seu programa deve imprimir uma linha, contendo o número de classificados para a próxima fase.

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

Exemplo de entrada 2	**Exemplo de saída 2**
5 2 500 500 500 500 500	5

Problema G

Ginástica

Vinícius gosta muito de se exercitar na academia de ginástica. Ele fez um acordo com o seu treinador para ter programas de exercícios diferentes a cada vez que usar a bicicleta ergométrica. Um programa, na linguagem das academias, é uma sequência de níveis de dificuldade do exercício. Os programas de Vinícius para a bicicleta ergométrica devem ter a mesma duração em minutos e os níveis de dificuldade devem mudar a cada minuto, para um nível imediatamente acima ou um nível imediatamente abaixo. Os níveis de dificuldade não podem estar abaixo de um mínimo e nem acima de um máximo previamente estipulados.

Seu problema é calcular o número de programas diferentes que o treinador pode construir, obedecidas as restrições acima.

Entrada

A entrada consiste de uma única linha que contém três inteiros, T, M, N ($1 \leq T \leq 50, 1 \leq M < N \leq 10^5$) em que T é o número de minutos do exercício, M é o valor mínimo de dificuldade permitido e N é o valor máximo de dificuldade permitido.

Saída

Seu programa deve produzir uma única linha com um inteiro representando o número de programas diferentes que o treinador pode construir. Como esse número pode ser grande, a resposta deve ser esse número módulo $10^9 + 7$.

Exemplo de entrada 1	**Exemplo de saída 1**
`3 2 5`	`10`

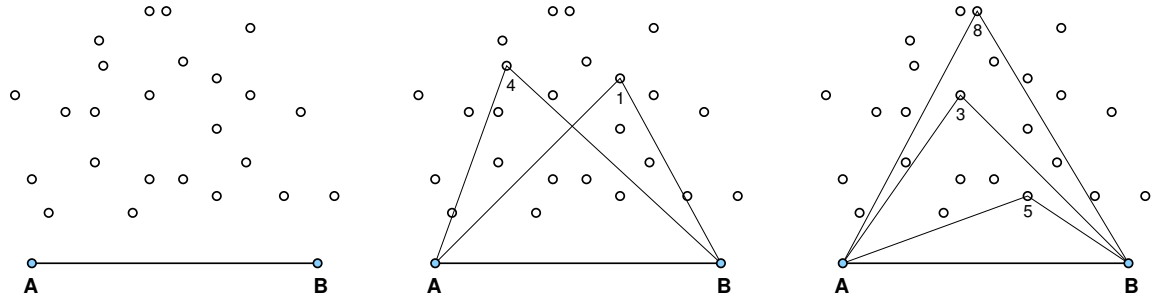
Exemplo de entrada 2	**Exemplo de saída 2**
`30 2 5`	`4356618`

Exemplo de entrada 3	**Exemplo de saída 3**
`50 1 100000`	`738072143`

Problema H

Hipercampo

São dadas duas âncoras, dois pontos $A = (X_A, 0)$ e $B = (X_B, 0)$, formando um segmento horizontal, tal que $0 < X_A < X_B$, e um conjunto P de N pontos da forma (X, Y), tal que $X > 0$ e $Y > 0$. A figura mais à esquerda exemplifica uma possível entrada.



Para "ligar" um ponto $v \in P$ precisamos desenhar os dois segmentos de reta (v, A) e (v, B). Queremos ligar vários pontos, mas de modo que os segmentos se interceptem apenas nas âncoras. Por exemplo, a figura do meio mostra dois pontos, 1 e 4, que não podem estar ligados ao mesmo tempo, pois haveria interseção dos segmentos fora das âncoras. A figura mais à direita mostra que é possível ligar pelo menos 3 pontos, 8, 5 e 3, com interseção apenas nas âncoras.

Seu programa deve computar o número máximo de pontos que é possível ligar com interseção de segmentos apenas nas âncoras.

Entrada

A primeira linha da entrada contém três inteiros, N ($1 \leq N \leq 100$), X_A e X_B ($0 < X_A < X_B \leq 10^4$), representando, respectivamente, o número de pontos no conjunto P e as abscissas das âncoras A e B. As N linhas seguintes contêm, cada uma, dois inteiros X_i e Y_i ($0 < X_i, Y_i \leq 10^4$), representando as coordenadas dos pontos, para $1 \leq i \leq N$. Não há pontos coincidentes e não há dois pontos u e v distintos tais que $\{A, u, v\}$ ou $\{B, u, v\}$ sejam colineares.

Saída

Seu programa deve imprimir uma linha contendo um inteiro, representando o número máximo de pontos de P que podem ser ligados com interseção de segmentos apenas nas âncoras.

Exemplos

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

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

Problema I

Imposto Real

O reino de Nlogônia é rico, o povo é educado e feliz, mas o Rei é um tirano quando o assunto se refere a impostos. A cada final de ano, cada cidade do país deve pagar uma determinada quantidade de quilos de ouro em impostos. Chegado o momento de coletar os impostos, o Rei envia sua carruagem real para recolher o ouro devido, usando as estradas do reino.

Cada estrada liga duas cidades diferentes e pode ser percorrida nas duas direções. A rede de estradas é tal que é possível ir de qualquer cidade para qualquer outra cidade, possivelmente passando por cidades intermediárias, mas há apenas um caminho entre duas cidades diferentes.

Em cada cidade há um cofre real, utilizado para armazenamento de ouro de impostos. Os cofres reais são imensos, de forma que cada cofre tem capacidade de armazenar todo o ouro devido por todo o reino. A carruagem sai da capital, percorrendo as estradas do reino, visitando as cidades para recolher o ouro devido, podendo usar qualquer cofre real para armazenar temporariamente uma parte do imposto recolhido, se necessário. Ao final da coleta, todo o ouro devido por todas as cidades deve estar armazenado no cofre real da capital.

Ávaro como é o Rei, ele contratou o seu time para, dados a quantidade de ouro a ser recolhido em cada cidade (em kg), a lista das estradas do reino, com os respectivos comprimentos (em km) e a capacidade de carga da carruagem real (em kg), determine qual é a mínima distância que a carruagem deve percorrer para recolher todo o ouro devido.

Entrada

A primeira linha contém dois inteiros N e C indicando respectivamente o número de cidades e a capacidade de carga da carruagem ($2 \leq N \leq 10^4$ e $1 \leq C \leq 100$). A capital do reino é identificada pelo número 1, as outras cidades são identificadas por inteiros de 2 a N. A segunda linha contém N inteiros E_i representando a quantidade de imposto devido por cada cidade i ($0 \leq E_i \leq 100$ para $1 \leq i \leq N$). Cada uma das M linhas seguintes contém três inteiros A, B e C, indicando que uma estrada liga a cidade A e a cidade B ($1 \leq A, B \leq N$) e tem comprimento C ($1 \leq C \leq 100$).

Saída

Seu programa deve produzir uma única linha com um inteiro representando a menor distância que a carruagem real deve percorrer para recolher todo o imposto devido, em km.

Exemplo de entrada 1	**Exemplo de saída 1**
6 10 0 10 10 10 10 10 1 4 7 5 1 2 3 5 3 2 5 2 6 5 2	44

Exemplo de entrada 2	**Exemplo de saída 2**
3 10 10 10 12 1 2 5 2 3 7	58

Exemplo de entrada 3	**Exemplo de saída 3**
5 9 5 2 6 3 6 1 2 1 2 3 1 2 4 1 2 5 1	10

Problema J

Jogo de Boca

Um jogo infantil, muito popular, é o *21 de boca*. O jogo é jogado da seguinte forma: o primeiro jogador diz um número, n_0, que pode ser 1 ou 2. O segundo jogador pode então dizer um número n_1 tal que $n_1 \in \{n_0 + 1, n_0 + 2\}$. E assim por diante, os jogadores se alternam, dizendo sempre um número que é um ou dois maior do que o anterior. O jogador que disser 21 ganha o jogo. Por exemplo, a sequência de números poderia ser: $1, 3, 5, 6, 7, 9, 11, 12, 14, 15, 16, 18, 19, 21$. Neste jogo, o primeiro jogador sempre perde, se o segundo souber jogar bem.

A cada nova geração as crianças ficam mais espertas. Atualmente, apesar de acharem o *21 de boca* um jogo interessante, muitas crianças não se sentem desafiadas o bastante e por isso resolveram generalizar o jogo, criando assim o *N de boca*. Dado um inteiro N, no lugar do 21, o primeiro jogador pode escolher 1 ou 2. A partir daí os jogadores se alternam, adicionando 1 ou 2 ao número anterior, até que um deles diga o número N e ganhe o jogo. Sabendo que ambos os jogadores são excelentes e sabem jogar muito bem, seu problema é determinar qual o inteiro inicial que o primeiro jogador deve escolher para ganhar o jogo.

Entrada

A entrada consiste de uma única linha que contém o inteiro N ($3 \leq N \leq 10^{100}$) escolhido para a partida atual do *N de boca*.

Saída

Seu programa deve produzir uma única linha com um inteiro representando o número, em $\{1, 2\}$, que o primeiro jogador deve escolher, para ganhar o jogo. Se não for possível, então o inteiro deve ser zero.

Exemplo de entrada 1	**Exemplo de saída 1**
7	1

Exemplo de entrada 2	**Exemplo de saída 2**
9	0

Exemplo de entrada 3	**Exemplo de saída 3**
123412341234123412341234123412341234	2

Problema K

K-ésimo

Dado um número real X da forma $A + \sqrt{B}$, com A e B inteiros positivos e $-1 < A - \sqrt{B} < 1$, e dois números inteiros N e K, sua tarefa é determinar o K-ésimo dígito menos significativo da parte inteira de X^N. Por exemplo, se $K = 1$, você precisa determinar o algarismo das unidades de $\lfloor X^N \rfloor$.

Entrada

A entrada consiste de uma única linha, que contém quatro números inteiros, A, B, N e K, com $1 \leq A, B \leq 10^4$, $1 \leq N \leq 10^9$ e $1 \leq K \leq 4$.

Saída

Seu programa deve imprimir uma única linha, contendo o K-ésimo dígito menos significativo da parte inteira de X^N.

Exemplo de entrada 1	**Exemplo de saída 1**
`3 10 1 1`	`6`

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

Exemplo de entrada 3	**Exemplo de saída 3**
`3 10 1000000000 1`	`1`

Exemplo de entrada 4	**Exemplo de saída 4**
`10 90 1000000000 2`	`9`

Problema L

Laboratório de biotecnologia

Uma cadeia ponderada é definida sobre um alfabeto Σ e uma função f que atribui um peso a cada caractere do alfabeto. Assim, podemos definir o peso de uma cadeia s como a soma dos pesos de todos os caracteres em s.

Vários problemas da bioinformática podem ser formalizados como problemas em cadeias ponderadas. Um exemplo é a espectrometria de massa de proteínas, uma técnica que permite identificar proteínas de forma bastante eficiente. Podemos representar cada aminoácido por um caractere distinto e uma proteína é representada pela cadeia de caracteres relativos aos aminoácidos que a compõe.

Uma das aplicações da espectrometria de massa de proteínas são buscas em bancos de dados. Para isso a cadeia que representa a proteina é dividida em subcadeias, a massa de cada subcadeia é determinada, e a lista de massas é comparada com um banco de dados de proteínas. Um dos desafios para essa técnica é lidar com cadeias muito grandes de caracteres, que podem ter várias possíveis subcadeias. A quantidade de subcadeias selecionadas é fundamental para obter bons resultados.

Em seu primeiro dia de estágio em um renomado laboratório de biotecnologia, Carlos recebeu a tarefa de determinar, para uma cadeia s, a quantidade de pesos distintos encontrada ao avaliar os pesos de todas as subcadeias não vazias de caracteres consecutivos de s.

Carlos não conseguiu pensar em uma solução eficiente para essa tarefa, mas felizmente ele conhece o grupo ideal para auxiliá-lo.

Considerando que s é formada por letras minúsculas e cada letra tem um peso diferente entre 1 e 26: a letra `a` tem peso 1, a letra `b` tem peso 2 e assim por diante. Mostre que seu time é capaz de ajudar Carlos a impressionar seu supervisor logo na primeira semana, com uma solução capaz de lidar facilmente com as maiores cadeias de caracteres existentes.

Entrada

Apenas uma linha, que contém a cadeia s formada por letras minúsculas, cujo comprimento $|s|$ satisfaz $1 \leq |s| \leq 10^5$.

Saída

Seu programa deve produzir uma única linha com um inteiro representando a quantidade de pesos distintos das subcadeias não vazias de caracteres consecutivos de s.

Exemplo de entrada 1	**Exemplo de saída 1**
`abbab`	`8`

Exemplo de entrada 2	**Exemplo de saída 2**
`adbbabdcdbcbacdabbaccdac`	`56`

Problema M

Máquina de café

O novo prédio da Sociedade Brasileira de Computação (SBC) possui 3 andares. Em determinadas épocas do ano, os funcionários da SBC bebem muito café. Por conta disso, a presidência da SBC decidiu presentear os funcionários com uma nova máquina de expresso. Esta máquina deve ser instalada em um dos 3 andares, mas a instalação deve ser feita de forma que as pessoas não percam muito tempo subindo e descendo escadas.

Cada funcionário da SBC bebe 1 café expresso por dia. Ele precisa ir do andar onde trabalha até o andar onde está a máquina e voltar para seu posto de trabalho. Todo funcionário leva 1 minuto para subir ou descer um andar. Como a SBC se importa muito com a eficiência, ela quer posicionar a máquina de forma a minimizar o tempo total gasto subindo e descendo escadas.

Sua tarefa é ajudar a diretoria a posicionar a máquina de forma a minimizar o tempo total gasto pelos funcionários subindo e descendo escadas.

Entrada

A entrada consiste em 3 números, A_1, A_2, A_3 ($0 \leq A_1, A_2, A_3 \leq 1000$), um por linha, onde A_i representa o número de pessoas que trabalham no i-ésimo andar.

Saída

Seu programa deve imprimir uma única linha, contendo o número total de minutos a serem gastos com o melhor posicionamento possível da máquina.

Exemplo de entrada 1	**Exemplo de saída 1**
10 20 30	80

Exemplo de entrada 2	**Exemplo de saída 2**
10 30 20	60

Exemplo de entrada 3	**Exemplo de saída 3**
30 10 20	100