1 Cometa (+)

 (+)

O cometa Halley é um dos cometas de menor período do Sistema Solar, completando uma volta em torno do Sol a cada 76 anos. Na última ocasião em que ele se tornou visível do planeta Terra, em 1986, várias agências espaciais enviaram sondas para coletar amostras de sua cauda e assim confirmar teorias sobre sua composição química.
Escreva um programa que, dado o ano atual, determina qual o próximo ano em que o cometa Halley será visível novamente no planeta Terra. Se o ano atual é um ano de passagem do cometa, considere que o cometa já passou nesse ano, ou seja, considere sempre o próximo ano de passagem após o atual.
Observação: Não se esqueça de considerar os anos bissextos.

Entrada

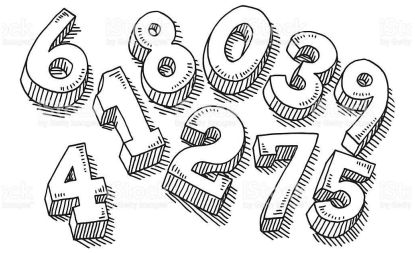
A única linha da entrada do programa contém um único inteiro A, indicando o ano atual.

Saída

Seu programa deve imprimir uma única linha, contendo um número inteiro, indicando o próximo ano em que o cometa Halley será visível novamente do planeta Terra.

Exemplo

Entrada	Saída
2010	2062



1 Jose (+)

 (+)

João tem um irmão mais novo, José, que começou a ir à escola e já está tendo problemas com números. Para ajudá-lo a "*pegar o jeito*" com a escala numérica, sua professora escreve dois números de três dígitos e pede a José para comparar esses números.
Entretanto, ao invés de interpretá-los com o dígito mais significativo à esquerda, ele deve interpretá-lo com o dígito mais significativo à direita. Ele tem que dizer à professora qual o maior dos dois números.
Escreva um programa que irá verificar as respostas de José.

Entrada

A entrada conterá uma única linha com dois números de três dígitos, A e B, os quais não serão iguais e não conterão zeros.

Saída

A saída deve conter, uma linha com o maior dos números na entrada, comparados como descrito no enunciado da tarefa. O número deve ser escrito invertido, para mostrar a José como ele deve lê-lo.

Exemplo

Entrada	Saída
734 893	437

1 overflow (++)

 (++)

Os computadores foram inventados para realizar cálculos muito rapidamente, e atendem a esse requisito de maneira extraordinária. Porém, nem toda "*conta*" pode ser feita num computador, pois ele não consegue representar todos os números dentro de sua memória.
Em um computador pessoal atual, por exemplo, o maior inteiro que é possível representar em sua memória é 4.294.967.295 ($2^{32} - 1$). Caso alguma "*conta*" executada pelo computador dê um resultado acima desse número, ocorrerá o que chamamos de *overflow*, que é quando o computador faz uma "*conta*" e o resultado não pode ser representado, por ser maior do que o valor máximo permitido (em inglês *overflow* significa *trasbordar*).
Por exemplo, se um computador só pode representar números menores do que 1023 ($2^10 - 1$) e mandarmos ele executar a conta 1022 + 5, vai ocorrer um *overflow*.
Dados o maior número que um computador consegue representar e uma expressão de soma ou multiplicação entre dois inteiros positivos, determine se ocorrerá, ou não, *overflow*.

Entrada

A primeira linha da entrada contém um inteiro *N* representando o maior número que o computador consegue representar.
A segunda linha contém um inteiro *P*, seguido de um espaço em branco, de um caractere *C* (que pode ser '+' ou '*', representando os operadores de *adição* e de *multiplicação*, respectivamente), de outro espaço em branco, e, finalmente, de outro número inteiro *Q*.
Assim, a segunda linha da entrada representa a expressão P + Q , se o caractere C for '+', ou P x Q , se o caractere C for '*'.

Saída

Seu programa deve imprimir a palavra 'OVERFLOW ' se o resultado da expressão causar um *overflow* no computador, ou a palavra 'OK ' caso contrário.
Ambas as palavras devem ser escritas com todas as letras maiúsculas.

Exemplo

Entrada	Saída
10 5 + 5	OK

Entrada	Saída
44 23 * 2	OVERFLOW

1 capicua (++)

 (++)

Alan Turing está aprendendo a decompor um número em unidades, dezenas, centenas, etc., e está com grandes dificuldades. Sua professora, preocupada com o rendimento de Alan Turing decidiu ensiná-lo uma brincadeira:
Alan Turing deve pegar um número com quatro algarismos, e verificar se o reverso deste número é ele próprio. Se for, Alan Turing deve responder *sim*, do contrário deve responder *não*.
Em verdade, a professora de Alan Turing está ensinando quando um número é chamado de *capicua* (ou também conhecido por *palíndromo*). Um número é capicua quando seu reverso é ele próprio.
Implemente este jogo divertido.

Entrada

A primeira linha da entrada contém um inteiro N ($N \geq 1$) representando a quantidade de números inteiros que Alan Turing deve responder *sim* (capicua) ou *não*. Cada uma das N linhas seguintes será composta por um inteiro de quatro algarismos.

Observação: O seu programa não poderá decompor o número na entrada, ou seja, não poderá ler o número N como quatro caracteres individuais que formam o número: deve lê-lo como número.

Saída

A saída consiste de N linhas, cada uma com ou o caracter 'S' se o número for capicua, ou o caracter 'N' caso o número não seja capicua.

Exemplo

Entrada	Saída
2 4569 5775	N S

Entrada	Saída
3 1458 1228 9779	N N S

1 Computação (++)

 (++)

A capacidade do ser humano em calcular quantidades nos mais variados modos foi um dos fatores que possibilitaram o desenvolvimento da Matemática, da Lógica e da Computação. Nos primórdios da Matemática e da Álgebra, utilizavam-se os dedos das mãos para efetuar cálculos.
Por volta do século III a.C., o matemático indiano Pingala inventou o sistema de numeração binário, que ainda hoje é utilizado no processamento de todos computadores digitais: o sistema estabelece que sequências específicas de uns e zeros pode representar qualquer número, letra, imagem, etc. Entretanto, a Computação está evoluindo rapidamente e recentemente a SBC (Sociedade Brasileira de Computação) inventou um computador com a base 4 (tetrade).
A SBC contratou você para fazer um programa que receba um número inteiro positivo, na base decimal, e converta-o para a base 4 utilizando divisões sucessivas. Você deve escrever um programa que, a partir de uma lista de números, calcule o valor correspondente de cada número na base 4.

Observação: Considere que os símbolos utilizados para representar as quantidades ZERO, UM, DOIS e TRÊS, na base 4 são, respectivamente, 0, 1, 2 e 3.

Entrada

A entrada contém um único conjunto de testes, que deve ser lido do dispositivo de entrada padrão (o teclado).
A primeira linha contém o número de inteiros N ($N \geq 1$) que será digitada.
A segunda linha contém N números inteiros n_i, cada um representando um número decimal.

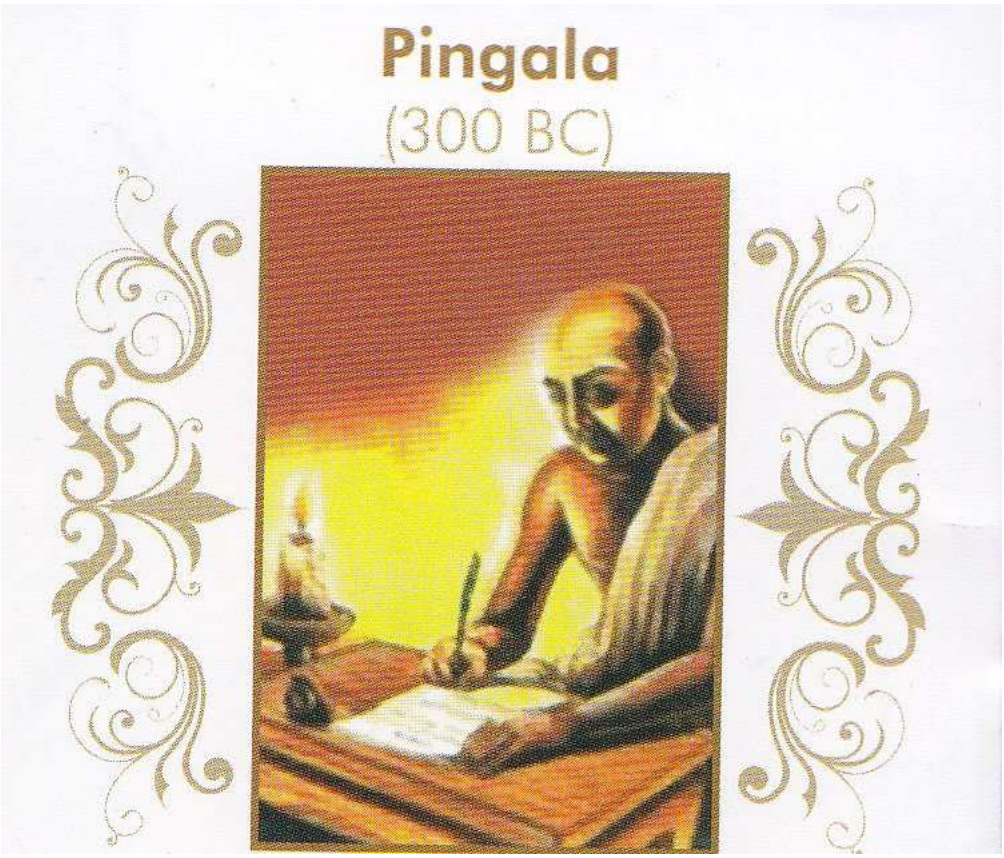
Saída

Seu programa deve imprimir, na saída padrão, os valores correspondentes na base 4 para cada número decimal digitado.

Exemplo

Entrada	Saída
5 1 2 3 4 10	1 2 3 10 22

Entrada	Saída
2 16 8	100 20


Pingala
(300 BC)

1 Envelopes (++)

 (++)

Kurt é um garoto muito esperto que adora promoções e sorteios. Como já participou de muitas promoções da forma "para participar, envie n rótulos de produtos ...", Kurt tem o hábito de guardar o rótulo de todos os produtos que compra. Dessa forma, sempre que uma empresa faz uma promoção ele já tem "*um monte*" de rótulos para mandar.

A SBC (Super Balas e Caramelos) está fazendo uma nova promoção, e, como era de se esperar, Kurt quer participar desta promoção. Nela é preciso enviar um envelope contendo um rótulo de cada tipo de bala que a SBC produz.

Por exemplo, se a SBC produz 3 tipos de balas (A, B, C) e uma pessoa tem 3 rótulos de A, 3 de B e 2 de C, ela pode enviar no máximo 2 envelopes, já que falta um rótulo de C para compor o terceiro envelope. Não há limite para o número de envelopes que uma pessoa pode enviar. Balas são a segunda coisa de que Kurt mais gosta (a primeira, como você, sabe são promoções). Por causa disso a quantidade de rótulos de balas que ele tem é muito grande, e ele não está conseguindo determinar a quantidade máxima de envelopes que ele pode enviar.

Como você é o melhor amigo de Kurt, ele pediu sua ajuda para fazer o cálculo, de modo que ele compre o número exato de envelopes.

Você deve escrever um programa que, a partir da lista de rótulos de Kurt, calcula o número máximo de envelopes válidos que ele pode enviar.

Entrada

A entrada contém um único conjunto de testes, que deve ser lido do dispositivo de entrada padrão (o teclado).

A primeira linha contém dois números inteiros N e K representando, respectivamente, a quantidade de rótulos de balas que Kurt possui e o número de tipos diferentes de bala que a SBC produz. Os tipos de balas são identificados por inteiros de 1 a K.

A segunda linha contém N números inteiros n_i, cada um representando um rótulo de bala que Kurt possui.

Saída

Seu programa deve imprimir, na saída padrão, o número máximo de envelopes válidos que Kurt pode enviar.

Exemplo

Entrada	Saída
10 2 1 1 1 1 1 2 2 2 2 2	5

Entrada	Saída
20 5 1 2 3 4 1 2 3 4 1 2 3 4 5 1 2 3 4 5 4 4	2

1 Sapo (+++)

 (+++)

Sebastião Bueno Coelho, apelidado de SBC pelos familiares e amigos, passou as férias de janeiro de 2011 no sítio de seus avós. Durante sua estadia, uma das atividades prediletas do SBC era nadar no rio que havia no *fundo* da casa onde morava. Uma das características do rio que mais impressionava SBC era um belo caminho, feito inteiramente com pedras brancas.

Há muito tempo, o avô de SBC notara que os habitantes do sítio atravessavam o rio com grande frequência e, por isso, construiu um caminho no rio com pedras posicionadas em linha reta; ao fazê-lo, tomou muito cuidado para que o espaçamento entre as pedras fosse de exatamente um metro. Hoje em dia, a única utilidade do caminho é servir de diversão para os sapos que vivem no rio, que pulam de uma pedra a outra agitadamente.

Um certo dia, enquanto descansava e nadava nas águas, SBC assistiu atentamente às acrobacias dos bichos e notou que cada sapo sempre pulava (zero, uma ou mais vezes) uma quantidade fixa de metros.

SBC sabe que você participa da OBI (Olimpíada Brasileira de Informática) todos os anos e, chegando na escola, resolveu desafiar-lhe com o seguinte problema:

"*Dado o número de pedras no rio, o número de sapos, a pedra inicial sobre a qual cada sapo está (cada pedra é identificada por sua posição na sequência de pedras – 1, 2, 3, ...) e a distância que cada sapo pula, determinar as posições onde pode existir um sapo depois que SBC chega no rio*".

Entrada

A primeira linha da entrada contém dois inteiros N ($N \geq 1$) e M ($M \leq 100$) representando, respectivamente, o número de pedras no rio e o número de sapos.

Cada uma das M linhas seguintes possui dois inteiros P e D representando, respectivamente, a posição inicial de um sapo e a distância fixa de pulo.

Saída

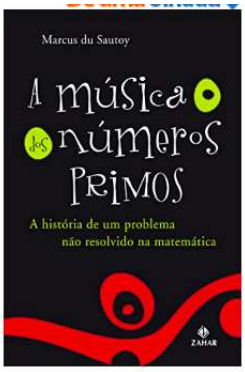
A saída contém N linhas. A i-ésima linha indica a possibilidade, ou não, de ter um sapo na i-ésima pedra.

Para as pedras que podem ter um sapo você deve imprimir 1, e para as pedras que, com certeza, não podem ter nenhum sapo você deve imprimir 0.

Exemplo

Entrada	Saída
5 2 3 2 4 4	1 0 1 1 1

Entrada	Saída
8 3 3 3 2 2 6 2	0 1 1 1 0 1 0 1



1 primos (++)

 (++)

No livro *A música dos números primos*, de Marcus du Saboy (2007, Editora Zahar, 471 páginas), o autor mostra que o mistério dos *números primos* passou a ser considerado o maior problema matemático de todos os tempos. Em meados do século XIX, o alemão Bernhard Riemann (1826 – 1866) formulou uma hipótese: é possível uma harmonia entre esses números primos, à semelhança da harmonia musical.
A partir de então, as mentes mais ambiciosas da Matemática embarcaram nessa procura que parece não ter fim. Atualmente, estipulou-se o prêmio de um milhão de dólares para quem provar a hipótese. O livro relata esse verdadeiro *Santo Graal* da Matemática, com casos interessantes e retratos pitorescos dos personagens que, desde Euclides, se envolveram nesse estranho mistério.
Você deverá, assim, pequisar e implementar um algoritmo que seja capaz de identificar se um dado número inteiro positivo é, ou não, um *número primo*.

Entrada

A primeira linha da entrada contém um inteiro N ($N \geq 1$) representando a quantidade de números inteiros positivos para os quais seu programa deve responder *sim* (o número é primo) ou *não* (o número é composto).
Cada uma das N linhas seguintes será composta por um inteiro positivo.

Observação: O seu programa deve estar preparado para receber números no intervalo de 2 a $2^64 - 1$.

Saída

A saída consiste de N linhas, cada uma com ou o caracter 'S' se o número for primo, ou o caracter 'N' caso o número não seja primo (número composto).

Exemplo

Entrada	Saída
2 3 11 16 60	S S S N N

1 vetores (+++)

 (+++)

Uma operação comum em diversas áreas da computação científica é a multiplicação de números inteiros positivos com grande número de dígitos. Por exemplo, multiplicar um número de 24 dígitos por outro de 16 dígitos, o que pode gerar um número de até 40 dígitos.
Você está participando de uma equipe de desenvolvimento de uma aplicação científica que deve implementar, utilizando um *vetor* para representar cada um dos números envolvidos, a operação de multiplicação mencionada.

Entrada

A primeira linha da entrada contém dois inteiros N e M ($1 \leq N \leq 40$ e $1 \leq M \leq 40$) representando a quantidade de dígitos dos números a serem multiplicados.
Cada uma das duas linhas seguintes conterá um dos números a ser multiplicado.

Saída

A saída consiste de uma única linha: o resultado da operação de multiplicação dos números fornecidos.

Exemplo

Entrada	Saída
5 8 17546 92130935	1616529385510

1 áreas (++)

 (++)

Um grande amigo(as) seu(sua), dos tempos de colégio, está cursando Arquitetura e pediu auxílio para você para resolver o seguinte problema:
Ele precisa calcular a área, em metros quadrados, de diversas figuras planas:

C círculo – cujo raio é dado por R;

E elipse – cujos raios maior e menor são, respectivamente, R e r;

T triângulo – cujos lados são a, b e c (nesta ordem);

Z trapézio – cujas bases maior e menor são, respectivamente, B e b, e a altura é H (nesta ordem).

Você já imaginou uma solução: elaborar um programa de computador que seja capaz de receber as informações necessárias e retornar a área da figura que, de acordo com seu(sua) amigo(a), precisa ter somente quatro casas decimais de precisão.
Observação: Utilize $\pi = 3,1415$.

Entrada

A primeira linha da entrada contém um inteiro N ($N \geq 1$) representando a quantidade de figuras planas para os quais seu programa deve calcular a área.
Cada uma das N linhas seguintes será composta por, primeiramente, um caractere que identifica qual é a figura e, em seguida, os parâmetros necessários para calcular sua área.

Saída

A saída consiste de N linhas, cada uma contendo a área da respectiva figura plana, com quatro casas decimais de precisão.
Observação: Considere que a resposta (área de cada figura plana) deverá ser *truncada* para quatro casas decimais de precisão.

Exemplo

Entrada	Saída
C 2.3748 E 2.6572 4.7428 T 8.0000 8.0000 8.0000 Z 7.3285 3.7523 4.6500	17.7175 39.5921 27.7128 25.7628