VII Simulado de Ingressantes

Instituto de Matemática e Estatística

Universidade de São Paulo

Caderno de Problemas

Departamento de Ciência da Computação IME-USP

Sexta, 6 de maio de 2022.

Instruções

- A competição tem duração de 5 horas;
- Os times são compostos de no máximo 3 integrantes;
- Os times podem utilizar somente um computador;
- Não é permitida a consulta de materiais durante a prova, exceto documentação das linguagens;
- O ambiente da prova será o CodeForces (www.codeforces.com).

- A entrada de cada problema deve ser lida da entrada padrão (teclado);
- A saída de cada problema deve ser escrita na saída padrão (tela);
- Siga o formato apresentado na descrição da saída, caso contrário não é garantido que seu código será aceito.

Vereditos das submissões	
In queue	Paciência
Accepted	Código aceito. Parabéns!
Wrong answer	Errado. Pode tentar novamente.
Time limit exceded	Seu programa demora muito para dar a resposta (certa ou errada). Pode tentar novamente.
Runtime error	Erro em tempo de execução (ex.: *segmentation fault*). Pode tentar novamente.
Compilation Error	Erro de compilação. Pode tentar novamente.

Problem A. Ajude o Nathan!

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Durante a Final Brasileira em Gramado, Thiago sequestrou Nathan para impedí-lo de participar da competição! Para ganhar a sua liberdade das mãos desse monstro sem escrúpulos, ele pode pagar o seu resgate (aproximadamente R$104.00) ou ele pode ganhar uma partida de Tênis de Mesa.

Na regra oficial, os jogadores alternam o saque a cada 2 pontos. Infelizmente, ambos estavam tão embriagados de adrenalina que eles se esqueceram dessa regra! Eles estavam alternando o saque a cada k pontos.

Dado que Thiago começou sacando e que o placar está atualmente T x N, escreva um programa para dizer quem será o próximo a sacar!

Input

Cada instância contém vários casos de teste. A primeira linha contém um inteiro t ($1 \leq t \leq 10^4$) - o número de casos de teste. A descrição de cada um dos t casos de teste seguem.

Cada caso de teste contém uma linha com 3 inteiros $k(1 \leq k \leq 10^9)$, T e $N(1 \leq T, N \leq 10^9)$ - o número de pontos para alternar o sacador, o número de pontos de Thiago e o número de pontos de Nathan, respectivamente.

Output

Para cada um dos t casos de teste, imprima apenas uma linha contendo "Thiago"(sem aspas) se Thiago é o próximo a sacar ou imprima "Nathan", caso contrário.

Examples

standard input	standard output
7 2 7 1 2 2 8 5 0 0 5 3 1 5 3 2 5 3 3 10 11 11	Thiago Nathan Thiago Thiago Nathan Nathan Thiago
1 2 3 4	Nathan

Problem B. Buscando Primos

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

É início de ano e a Universidade de São Paulo está recebendo seus ingressantes. Todo ingressante recebe um número, o número USP, que é mais um número para memorizar junto com Telefone, RG, CPF, etc. Jiang "*Fábio*" Zhi, um matemático nato, está curioso para entender a natureza da distribuição dos números USP. De certo não são sequenciais e também não estão em ordem alfabética. Como todo problema difícil de matemática Jiang pensou consigo "teoria dos números, é claro!". Contudo, ele não estava em dia com a lista de teoria dos números, e precisou pedir ajuda para Enrique "*Byakko*" Junchaya, aquele que tudo sabe.

Depois de pensar muito, eles perceberam alguns padrões envolvendo a decomposição em primos do número USP de cada aluno. Porém, para testar a teoria, eles precisam de dados. Ambos são muito bons em matemática, porém, não sabem programar. Eles vieram até você e pediram sua ajuda para resolver esse mistério.

Dada uma lista de nomes e números USP, eles precisam que você encontre o aluno com o maior fator primo no número USP. Em caso de empate, retorne o que aparece primeiro na lista.

Input

A primeira linha contém um inteiro positivo n ($n \leq 10^5$), o número de alunos. As próximas n linhas contém uma string s ($1 \leq |s| \leq 20$) consistindo de letras minúsculas do alfabeto latino e um número x ($2 \leq x \leq 10^8$). Todos os nomes são distintos.

Output

Imprima uma string, o nome do aluno com maior fator primo em seu número USP. Em caso de empate, imprima o que aparece primeiro na lista.

Examples

standard input	standard output
5 arrais 2 cassio 3 joao 5 wang 7 mori 11	mori
4 caiaffa 97 antonio 223 matheus 21631 gustavo 384	antonio
5 jiang 99999931 nathan 99999941 germano 99999959 noronha 99999971 byakko 99999989	byakko

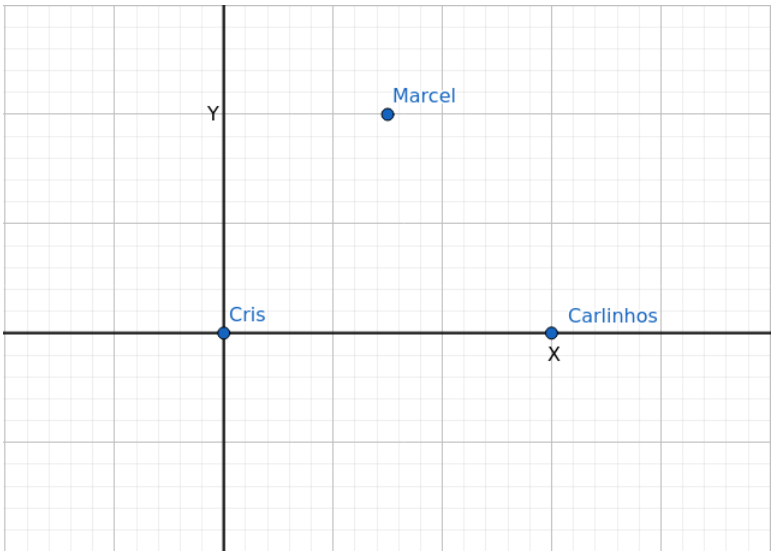
Note

No segundo caso teste temos que $21631 = 97 \times 223$ e $384 = 2^7 \times 3$. Logo, o aluno com o maior número primo na decomposição que aparece mais cedo na lista é o antonio.

Problem C. CCM

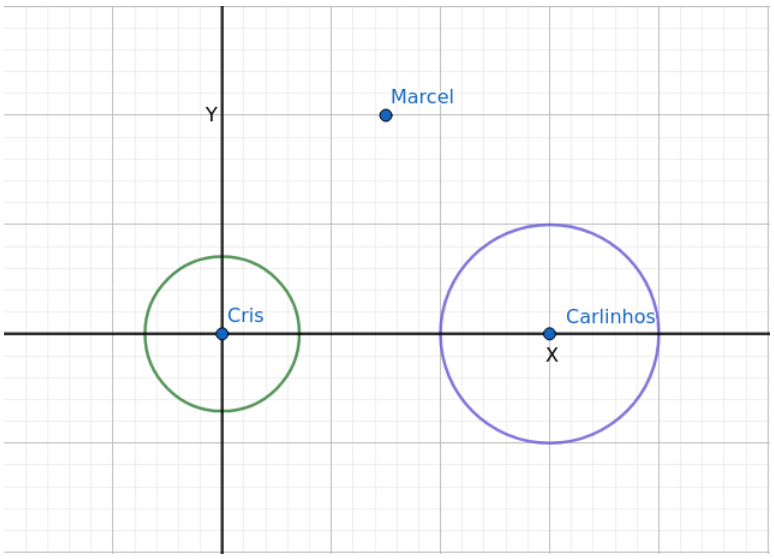
Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

O time CCM formado pelos professores Cris, Carlinhos e Marcel acabou de ganhar a maratona de programação! E para celebrar este feito eles vão comemorar enchendo balões. Cada professor pegou um balão e foi para o salão de eventos de hotel. O salão pode ser visto como um plano cartesiano onde Cris está na posição $(0, 0)$, Carlinhos na posição $(X, 0)$ e Marcel na posição $(X/2, Y)$.

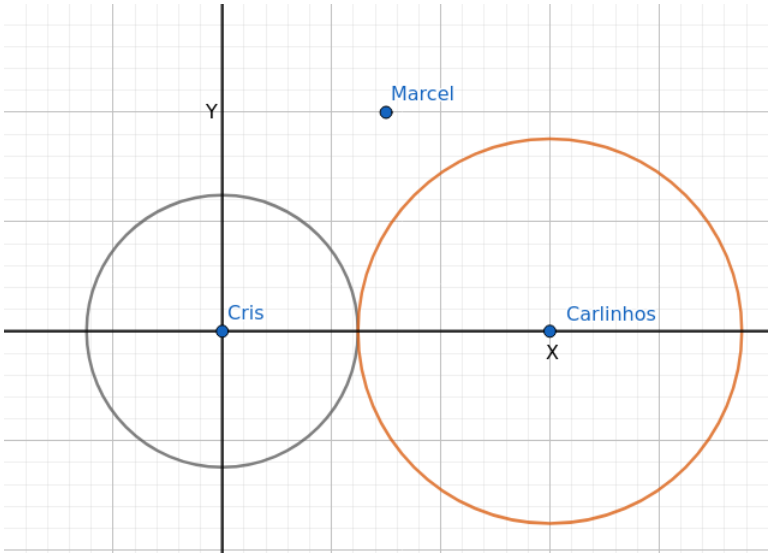


Posição inicial dos professores.

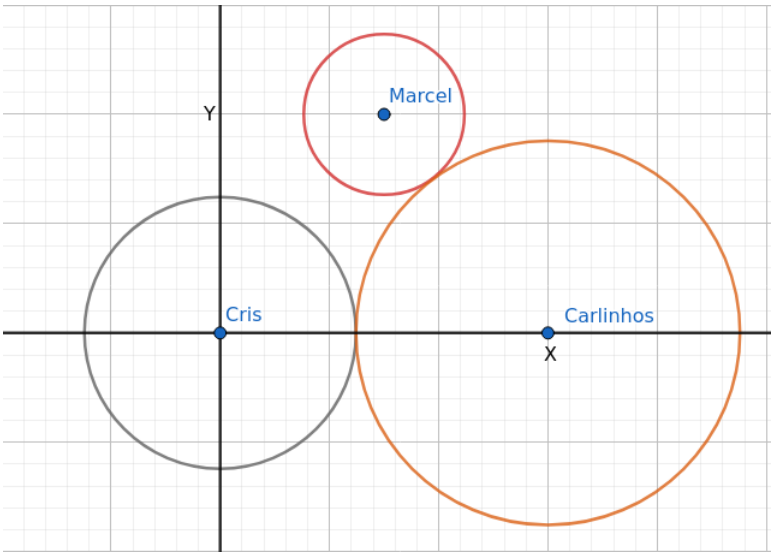
Cris e Carlinhos começam a encher seus balões, geometricamente isto pode ser visto como círculos centrados nos professores que aumentam de raio continuamente. Cris enche seu balão com velocidade V_{cr} m/s, isto é, o raio de seu balão cresce V_{cr} metros a cada segundo. Já Carlinhos enche seu balão com velocidade V_{ca} m/s. Quando o balão de Cris e Carlinhos se encontram eles param de encher seus balões. Neste instante Marcel começa a encher o seu com velocidade V_{ma} até que seu balão se encontre com um dos seus companheiros. As figuras a seguir ilustram o processo.



Cris e Carlinhos começam a encher seus balões.



Cris e Carlinhos param quando seus balões se encontram.



Marcel enche seu balão até que se encontre com o balão de um dos seus companheiros.

Carlinhos gostaria que seu balão tivesse área maior ou igual ao balão de seus companheiros, porém ele não quer gastar toda sua energia nessa atividade. Por isso, ele deseja saber qual o menor valor de V_{ca} tal que ao final do processo seu balão tem área maior ou igual ao balão de Cris e ao balão de Marcel.

Input

O input consiste de duas linhas. A primeira contém dois inteiros, $1 \leq X, Y \leq 10^9$ que compõe as posições dos professores. A segunda linha contém dois números de ponto-flutuante $1.0 \leq V_{cr}, V_{ma} \leq 10.0$ que representam a velocidade com que Cris e Marcel enchem seus balões, respectivamente.

Output

O output deve conter um único número de ponto-flutuante: V_{ca}, a menor velocidade que Carlinhos deve encher seu balão tal que seu balão é maior ou igual ao balão de Cris e Marcel. Sua resposta será considerara correta se o erro absoluto ou o erro relativo entre sua resposta e a do juiz for menor que 10^{-4}. Isto significa que se sua resposta for B e a do juiz for A, então a solução será considerara correta caso $|A - B| \leq 10^{-4}$ ou $|A - B| \leq B \times 10^{-4}$. É garantido que existe uma maneira de Carlinhos terminar com o balão maior ou igual o de seus companheiros. É garantido que existe uma resposta com $V_{ca} \leq 10^{18}$.

Recomenda-se imprimir a resposta com ao menos 15 casas decimais. Para isso use:

- `printf(".15%lf\n", resposta);` em C;
- `cout << setprecision(15) << resposta << endl;` em C++;
- `format(resposta, '.15f')` em Python;

Examples

standard input	standard output
2 1 1.000000 1.000000	1
10 12 7.000000 2.000000	13
183457 213874 3.141592 2.718281	5.44783769599496

Note

No segundo caso teste se Carlinhos encher seu balão com velocidade 13, então após meio segundo seu balão e o de Cris irão se encontrar na posição $\left(\frac{7}{2}, 0\right)$. Marcel então irá encher seu balão até ele se encontrar com o de Carlinhos. Ao final do processo, Cris terá um balão de raio $\frac{7}{2}$ enquanto Carlinhos e Marcel terão um balão de raio $\frac{13}{2}$. Pode se mostrar que se $V_{ca} < 13$ então Carlinhos terminaria com um balão menor que um de seus companheiros.

Problem D. Dias Felizes

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Gustavo de Medeiros é um rapaz que se preocupa muito com sua saúde. Sabendo que saúde mental é um tópico muito importante, Gustavo decidiu monitorar seus dias por meio de um diário. Cada dia recebe uma nota inteira no intervalo $[-10, 10]$ que reflete como Gustavo se sentiu em relação ao mesmo. Então, dias bons são dias com notas positivas, dias ruins são dias com notas negativas e dias normais recebem zero.

Gustavo registrou a nota dos últimos n dias e precisa de sua ajuda para determinar qual é a média das notas.

Input

A primeira linha contém um número inteiro positivo $n \leq 10^5$, o número de dias que o Gustavo registrou as notas. A linha seguinte contém uma lista a de n inteiros separados por espaços tal que $a_i \in [-10, 10]$ para todo $1 \leq i \leq n$.

Output

Imprima a string ":)" (sem aspas), se a média das notas é positiva, ":|" se a média é zero e ":(" se a média é negativa.

Examples

standard input	standard output
5 10 10 10 10 10	:)
2 -1 1	:|
3 1 2 -4	:(

Problem E. El Classificador

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

Lucas "*El Classificador*" Harada, um integrante do MaratonUSP, criou o famoso algoritmo conhecido como *Classificador*. Dada uma lista a de tamanho m, e um inteiro x, o algoritmo acha o primeiro inteiro nesta lista maior ou igual a x e remove dela.

O MaratonUSP fez uma encomenda de n tênis, todos do mesmo modelo mas de tamanhos variados. Ao todo, q clientes irão comprar o produto. Cada cliente tem um tamanho mínimo do tênis que gostaria de levar e Lucas deve utilizar o critério do *Classificador* para gerenciar o estoque. Como o estoque pode ser bem grande, você deve ajudar o Lucas e fazer um programa eficiente que diz qual tamanho do tênis que cada cliente irá levar.

Input

A primeira linha contém 2 inteiros n, q ($1 \leq n, q \leq 2 \cdot 10^5$) — o número de tênis comprados e o número de clientes, respectivamente.

A próxima linha tem n inteiros $l_1, l_2, \ldots, l_n$ ($1 \leq l_i \leq 10^9$) — os tamanhos dos tênis inicialmente no estoque.

As próximas q linhas terão um inteiro x ($1 \leq x \leq 10^9$) representando o tamanho mínimo do tênis que o cliente quer.

Output

Imprima q linhas, sendo a i-ésima linha o tamanho do tênis que o i-ésimo cliente irá levar. Caso não tenha nenhum tênis no estoque que agrade ele, imprima -1.

Examples

standard input	standard output
2 3 4 4 3 4 4	4 4 -1
3 5 1 2 3 2 1 3 5 1	2 1 3 -1 -1
5 6 10 5 7 1 7 6 2 15 8 1 1	7 5 -1 10 1 7

Note

No caso teste 1, temos inicialmente dois tênis de tamanho 4. O primeiro cliente quer um tênis de tamanho pelo menos 3, e leva o de tamanho 4. O segundo cliente leva o último tênis de tamanho 4. Como não tem mais tênis, o terceiro cliente não leva nada :(.

Problem F. Fila do Bandeco

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Novo ano, novo ciclo, nova rotina, novos amigos: ingressar em uma Universidade como a USP é extremamente gratificante e traz muitas novidades!

Uma delas é o Bandejão!

A possibilidade de fazer uma refeição completa por apenas R$2,00 é uma oportunidade imperdível. Exatamente por isso, quase todos os que transitam pela Cidade Universitária tentam bandejar em algum dos 4 bandejões: Central, Física, Prefeitura e Química. Digamos que temos N pessoas querendo bandejar, sendo que a i-ésima pessoa foi para o bandejão b_i(que pode ser C se ela foi para o Central, F se foi para a Física, P se foi para a Prefeitura e Q se foi para a Química) e ela demora v_i minutos para comer.

Como N pode ser bem grande e algumas pessoas comem devagar (o autor do problema come beeeem devagar) as filas acabam ficando imensas! Para piorar, cada bandejão somente pode acomodar UMA pessoa por vez por causa do distanciamento social.

Dado que os bandejões ficam abertos por somente mais T minutos, descubra qual o número máximo de estudantes que ainda podem bandejar (veja que a ordem da fila não precisa ser respeitada).

Input

A primeira linha da entrada contém dois inteiros: N ($1 \leq N \leq 4 \cdot 10^5$) - o número de estudantes - e T ($1 \leq T \leq 10^9$) - o tempo que os 4 bandejões ainda ficarão abertos.

Depois, N linhas seguem. A i-ésima linha contém um caracter b_i - que pode ser C, F, P ou Q representando um bandejão - e um inteiro v_i ($1 \leq v_i \leq 10^9$) representando o tempo que essa pessoa demora para comer.

Output

Imprima somente um inteiro contendo o número máximo de estudantes que podem bandejar.

Examples

standard input	standard output
6 10 C 5 F 3 P 2 Q 4 C 1 C 6	5
1 1 C 3	0
4 10 C 6 C 3 C 4 C 5	2

Problem G. Grande Encontro

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Cássio e Morete – coach e co-coaches do MaratonUSP, respectivamente – precisam se encontrar para realizar uma reunião ultrassecreta onde eles vão decidir o futuro do grupo de extensão, assim como o plano de dominação do resto dos grupos. Como ambos moram perto da linha 4 amarela – uma infame linha de metrô da cidade de São Paulo – eles decidiram usá-la como meio de locomoção até o local do encontro.

Dado que essa reunião é de extrema importância, Cássio e Morete decidiram que não iriam combinar o local final da reunião através da internet, com medo da espionagem de outros grupos de extensão, como o SegurIME. Desta forma, ambos combinaram de embarcar nas estações mais próximas de suas respectivas casas e de se encontrarem o mais rápido possível em alguma estação, para então decidir o local final da reunião.

Como o LaboratorioDeCodigo pode ter sabotado o aplicativo do metrô, eles pediram a sua ajuda para descobrir em quanto tempo eles conseguem se encontrar. Sabendo que cada estação de metro está a exatamente 1 minuto de distância uma da outra e que ambos embarcam nas estações no mesmo instante, ajude os coaches a descobrirem qual o tempo mínimo necessário para eles conseguirem se encontrar.

Input

Uma linha contendo um inteiro n ($1 \leq n \leq 100$) – o número de estações da linha amarela.

Em seguida n linhas, cada linha contento uma string s_i ($1 \leq |s_i| \leq 100$) de letras minusculas – o nome da i-ésima estação da linha amarela, seguindo a orientação oeste-leste. É garantido que nenhum par de estações tem o mesmo nome.

Uma linha contendo duas strings c e m – as estações que Cássio e Morete embarcaram, respectivamente. É garantido que c e m são nomes de estações da linha amarela.

Output

Uma linha contendo um inteiro x – o tempo minimo necessário para os dois se encontrarem em uma estação.

Example

standard input	standard output
7 butanta pinheiros faria fradique oscar paulista luz pinheiros oscar	2

Problem H. Harada Futebol Clube

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.25 seconds
Memory limit:	256 megabytes

Lucas Harada é um dos maiores fãs do SPFC (São Petersburgo Futebol Clube) e também de problemas de combinatória, então porque não juntar os dois em um problema? Uma formação tática de um time de futebol é uma distribuição dos jogadores entre as posições de goleiro, defesa, meio-campo e ataque em que tem exatamente um goleiro e ao menos um jogador na defesa, ao menos um no meio-campo, e ao menos um no ataque.

Por exemplo, em um time de futebol tradicional, com 11 jogadores, alguns exemplos de formação tática são: 1-4-4-2, 1-5-4-1, 1-3-4-3. Note que não importa quais jogadores estão em cada posição, e somente quantos jogadores. Agora Harada está se questionando, se um time de futebol tivesse N jogadores, quantas formações táticas seriam possíveis?

Input

O input consiste de um único inteiro $4 \leq N \leq 10^6$.

Output

O output consiste de um único inteiro: o número de formações táticas possíveis em um time de N jogadores.

Examples

standard input	standard output
4	1
6	6
11	36
1000000	499997500003

Note

No segundo caso teste as formações possíveis são as seguintes: 1-1-1-3; 1-2-2-1; 1-1-3-1; 1-2-1-2; 1-3-1-1; 1-1-2-2.

Problem I. Irritando o Carlinhos

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Thiago é um menino muito travesso, passa o dia pregando peças pelo departamento de computação. Por outro lado, Calinhos, sente-se na missão de exercer seu papel de professor e ensinar uma lição ao Thiago quando surge a oportunidade. Com medo das punições, Thiago foge à menor menção do nome do Carlinhos. Por outro lado, Carlinhos, como um dos maiores maratonistas do Brasil, não se importa em correr atrás dele, em algum momento ele se cansará de fugir e a justiça será feita.

Nesse problema vamos considerar que Thiago e Carlinhos são dois pontos no plano 2D com coordenadas (tx, ty) e (cx, cy) respectivamente. Uma sequência de movimentos pode ser representada por uma string contendo as letras U (Up), D (Down), R (Right) e L (Left). De forma que cada uma representa um movimento diferente da seguinte forma:

- U : $(x, y) \to (x, y + 1)$;
- D : $(x, y) \to (x, y - 1)$;
- R : $(x, y) \to (x + 1, y)$;
- L : $(x, y) \to (x - 1, y)$.

Dadas as posições inicias de cada um, distintas, e a sequência de movimentos de cada um, ambas de mesmo tamanho, determine se Thiago foi punido pelo Carlinhos. Carlinhos consegue punir o Thiago se, a qualquer momento, ambos têm as mesmas posições. Os movimentos ocorrem de forma alternada, porém, como Thiago está sempre muito atento, então o mesmo se movimenta antes.

Input

A primeira linha contém quatro inteiros separados por espaços tx, ty, cx, cy ($-10^5 \leq tx, ty, cx, cy \leq 10^5$) as coordenadas iniciais nos eixos x e y de Thiago e Carlinhos respectivamente. É garantido que as coordenadas são distintas.

As próximas duas linhas contém cada uma uma string s, t ($1 \leq |s| = |t| \leq 2 \cdot 10^5$) consistindo somente das letras U, D, L e R, os movimentos de Thiago e Carlinhos respectivamente.

Output

Imprima a string "Rodou!" (sem aspas) se Thiago foi pego pelo Carlinhos ou "Quase!" (sem aspas) caso contrário.

Examples

standard input	standard output
0 1 0 0 UU UU	Quase!
2 0 0 0 L R	Rodou!
2 2 0 0 DLDL RURU	Rodou!

Note

Note que no primeiro caso teste ocorre a seguinte sequência de movimentos

- Thiago: $(0, 1) \to (0, 2)$;
- Carlinhos: $(0, 0) \to (0, 1)$;
- Thiago: $(0, 2) \to (0, 3)$;
- Carlinhos: $(0, 1) \to (0, 2)$.

Portanto, nunca ficaram na mesma posição.

No segundo caso teste temos a seguinte sequência de movimentos

- Thiago: $(2, 0) \to (1, 0)$;
- Carlinhos: $(0, 0) \to (1, 0)$.

Portanto, Thiago foi pego no último movimento.

Problem J. Jogo do Noronha

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Enquanto estava de férias Noronha criou um jogo para melhorar sua memorização. Ele já é um profissional nesse jogo, e decidiu jogar com você! Dado um conjunto inicialmente vazio, você deve fazer uma das seguintes operações:

1. Adicione um elemento x no conjunto;
2. Diga qual é o maior elemento do conjunto logo após a operação t;
3. Diga qual é o menor elemento do conjunto logo após a operação t;
4. Diga qual é a soma dos elementos do conjunto logo após a operação t.

Como você é esperto, decidiu fazer um programa que joga esse jogo por você.

Input

Na primeira linha é dado um inteiro n ($1 \leq n \leq 10^5$) — o número total de operações.

As próximas n linhas descrevem as operações feitas. Cada linha representa as operações como a seguir:

- "1 x" você deve adicionar um elemento x ($1 \leq x \leq 10^9$) ao conjunto;
- "2 t" você deve dizer qual o maior elemento logo após a operação t ($1 \leq t \leq n-1$);
- "3 t" você deve dizer qual o menor elemento logo após a operação t ($1 \leq t \leq n-1$);
- "4 t" você deve dizer qual a soma dos elementos logo após a operação t ($1 \leq t \leq n-1$).

É garantido que t é menor ou igual ao número de operações feitas até o momento e que a primeira operação é do tipo 1.

Output

Para cada operação do tipo 2, 3 ou 4, deve ser impresso um inteiro com a resposta da operação.

Examples

standard input	standard output
7 1 10 2 1 2 2 3 2 4 2 1 5 2 6	10 10 10 10 10
2 1 3 4 1	3
6 1 1 1 10 2 2 3 3 4 4 4 1	10 1 11 1