MARATONA DE PROGRAMAÇÃO

InterFatecs Ourinhos -2019

Fase Final - 24/08/2019

Caderno de Problemas

IBM

FUNDAÇÃO FAT

WEBBY Internet

Gemmap GESTÃO MUNICIPAL

mf Soluções em Tecnologia

Fatec Ourinhos

CPS Centro Paula Souza

KPlay fábrica de software

CENGAGE

M4

GRUPO CEDNET

FOMENTAVALE

www.interfatecs.com.br

1 Instruções

Este caderno contém 09 problemas – identificados por letras de A até I, com páginas numeradas de 3 até 22. Verifique se seu caderno está completo.

Informações gerais

1. Sobre a competição
   (a) A competição possui duração de 5 horas (início as 12:30h término as 17:30h);
   (b) NÃO é permitido acesso a conteúdo da Internet ou qualquer outro meio eletrônico digital;
   (c) É permitido somente acesso a conteúdo impresso (cadernos, apostilas, livros);
   (d) É vedada a comunicação entre as equipes durante a competição, bem como a troca de material de consulta entre elas;
   (e) Cada equipe terá acesso a 1 computador dotado do ambiente de submissão de programas (BOCA), dos compiladores, link-editores e IDEs requeridos pelas linguagens de programação permitidas;
   (f) NÃO é permitido o uso de notebooks ou outro tipo de computador ou assistente pessoal;
   (g) Os problemas têm o mesmo valor na correção.
2. Sobre o arquivo de solução e submissão:
   (a) O arquivo de solução (o programa fonte) deve ter o mesmo nome que o especificado no enunciado (logo após o título do problema);
   (b) confirme se você escolheu a linguagem correta e está com o nome de arquivo correto antes de submeter a sua solução;
   (c) NÃO insira acentos no arquivo-fonte.
3. Sobre a entrada
   (a) A entrada de seu programa deve ser lida da entrada padrão (não use interface gráfica);
   (b) Seu programa será testado em vários casos de teste válidos além daqueles apresentados nos exemplos. Considere que seu programa será executado uma vez para cada caso de teste.
4. Sobre a saída
   (a) A saída do seu programa deve ser escrita na saída padrão;
   (b) Não exiba qualquer outra mensagem além do especificado no enunciado.

MARATONA DE PROGRAMAÇÃO
InterFatecs

Problema A

Caixa Eletrônico

Arquivo fonte: caixaeletronico.{ c | cpp | java | py }

Autor: Sérgio Luiz Banin (Fatec São Paulo e Fatec São Caetano do Sul)

Você foi contratado pelo Banco Regional para desenvolver um pacote de software para ser usado na análise e auditoria do volume de saques que ocorrem nos terminais de autoatendimento, o famoso Caixa Eletrônico, também conhecido pela sigla em inglês: ATM.

O Banco Regional usa um tipo de terminal que contém três bandejas para as notas de Real e que são denominadas pelas letras maiúsculas A, B e C. Por ser um terminal de baixo custo ele tem certas limitações. A mais relevante para você neste momento é que ele sempre oferece uma única opção de saque aos correntistas. Quando o valor a ser sacado é fornecido pelo usuário, o software do terminal verifica se o saque é possível e, em caso positivo, ele usa dois critério para definir a decomposição do valor:

a) menor número de notas necessário

b) usa primeiro notas de maior valor.

A título de exemplo, se as notas disponíveis forem R$ 100,00, R$ 50,00 e R$ 10,00 e o usuário quiser sacar R$ 170,00 o terminal, sem qualquer outra opção, fornecerá:

R$ 170,00, com 4 notas: 1 nota de R$ 100,00 + 1 nota de R$ 50,00 + 2 notas de R$ 10,00

Em um segundo exemplo, se as notas disponíveis forem R$ 5,00, R$ 20,00 e R$ 50,00 e o usuário quiser sacar R$ 65,00 haverá dois casos empatados com o mesmo número mínimo de notas:

R$ 65,00, com 4 notas: 1 nota de R$ 50,00 + 3 notas de R$ 5,00 - esta será a opção apresentada

R$ 65,00, com 4 notas: 3 notas de R$ 20,00 + 1 nota de R$ 5,00

Em casos assim o algoritmo usará o critério **b** para desempate e a opção de saque será a primeira apresentada

Uma segunda característica importante para você neste momento é que as bandejas A, B e C, **sempre** são carregadas com diferentes valores de notas. O terceiro aspecto a ser levado em conta é que ao recarregar as bandejas não há uma ordem específica para os valores das notas. Desde que seja respeitada a regra de cada bandeja ter um valor diferente a ordem pode ser qualquer uma.

Por fim, como o terminal é de baixo custo, quando uma bandeja fica vazia os saques não são mais permitidos, uma vez que seu algoritmo não foi projetado para decompor o valor a ser sacado em combinações diversas, valendo sempre a regra já descrita do menor número de notas.

Neste novo projeto sua primeira tarefa é escrever um programa que informe quantas notas restaram em cada bandeja do ATM após um certo período no qual saques foram realizados pelos correntistas. Para isso, você recebe um arquivo de log do terminal. Este arquivo de log contém a situação inicial do ATM, com o valor e a quantidade de notas de cada bandeja, bem como a quantidade e todos os valores sacados. Seu programa deve ler o log e produzir uma saída informando quantas notas restaram em cada bandeja após os saques terem sido realizados.

Entrada

A entrada contém um único caso de teste no qual as três primeiras linhas representam as bandejas A, B e C respectivamente. Para cada bandeja a linha contém dois números inteiros: VN que é o valor de face das notas e $QTDE$ $(100 \leqslant QTDE \leqslant 5000)$ que é a quantidade de notas contidas na bandeja.

Os valores VN nas três bandejas não estão necessariamente em uma ordem específica. Cada bandeja tem obrigatoriamente um valor diferente de nota e são usados exclusivamente valores existentes na moeda Real (2, 5, 10, 20, 50, 100).

Na quarta linha há um número inteiro $NSAQUES$ $(2 \leqslant NSAQUES \leqslant 1000)$ que representa a quantidade de saques realizados no terminal de autoatendimento. Por fim há $NSAQUES$ linhas cada uma contendo o valor sacado. Todos os saques presentes no caso de teste podem ser realizados, ou seja, são valores possíveis de serem decompostos com as notas presentes nas bandejas A, B e C. Além disso, garante-se que há notas suficientes para a realização de todos os saques.

Saída

A saída do programa contém três linhas informando quantas notas restaram em cada bandeja na forma de uma frase escrita em letras minúsculas: **na bandeja X restaram NN notas**, onde **X** é substituído pela letra (maiúscula) da bandeja e **NN** é substituído pela quantidade de notas que sobraram. Não esqueça da quebra de linha ao final de cada frase.

Exemplo de Entrada 1

```
50 100
100 100
20 100
3
100
120
150
```

Exemplo de Saída 1

```
na bandeja A restaram 99 notas
na bandeja B restaram 97 notas
na bandeja C restaram 99 notas
```

Exemplo de Entrada 2

```
20 300
50 300
100 300
6
180
70
140
280
330
1130
```

Exemplo de Saída 2

```
na bandeja A restaram 281 notas
na bandeja B restaram 297 notas
na bandeja C restaram 284 notas
```

Exemplo de Entrada 3	Exemplo de Saída 3
5 100 20 100 50 100 6 15 20 25 50 55 65	na bandeja A restaram 92 notas na bandeja B restaram 98 notas na bandeja C restaram 97 notas

Esta página foi propositadamente deixada em branco.

MARATONA DE PROGRAMAÇÃO
InterFatecs

Problema B

Balas Dumbinho

Arquivo fonte: dumbinho.{ c | cpp | java | py }
Autor: Leandro Luque (Fatec Mogi das Cruzes)

Xoéslei é filho de um grande produtor de balas, o famoso Dumbo. Sua marca de balas ficou bastante conhecida nas últimas décadas e tem sido consumida em todo o mundo. Recentemente, a Dumbo Balas, empresa do pai, fundiu-se com a 8-Bonito, outra marca famosa de balas. Como os embrulhos destas duas balas são muitíssimo parecidos, Xoéslei procurou uma forma de ajudar a nova empresa na organização do estoque. Durante sua pesquisa, ele percebeu que o código utilizado nas balas dumbinho e 8-bonito diferenciavam-se de forma definitiva. A soma dos números do código de uma bala dumbinho era sempre par, enquanto a soma dos números do código de uma bala 8-bonito era sempre ímpar. Como exemplo, a bala com código 2342 é da marca 8-bonito, pois a soma 2+3+4+2 é um número ímpar (= 11).

Xoéslei pediu sua ajuda para escrever um programa de computador que, dado o código de uma bala, diz se ela é da marca dumbinho ou 8-bonito.

Entrada

A entrada contém um número inteiro C $(1 \leqslant C \leqslant 1000000)$, representando o código da bala.

Saída

A saída contém o texto $dumbinho$ (em minúsculas) ou $8 - bonito$ (em minúsculas) indicando qual é o fabricante da bala com o código C. Finalize com uma quebra de linha.

Exemplo de Entrada 1	Exemplo de Saída 1
23582	dumbinho

Exemplo de Entrada 2	Exemplo de Saída 2
1000000	8-bonito

Exemplo de Entrada 3	Exemplo de Saída 3
1112	8-bonito

Esta página foi propositadamente deixada em branco.

MARATONA DE PROGRAMAÇÃO
InterFatecs

Problema C

Password Strength

Arquivo fonte: forcasenha.{ c | cpp | java | py }
Autor: Erico de Souza Veriscimo (Fatec Mogi das Cruzes)

Elenilson and Erinilson are developing the new InterFatecs Programming Contest portal. They are currently having difficulty writing code that validates user registration passwords. They found out that you always participate in programming contests and asked for your help. To do so, you must meet the following criteria:

- A password cannot have punctuation characters (.,!?;:), accent (áéíóúâêôãàçÁÉíóÚÃÔÔÕÀÇ), or spaces;
- A password must contain at least one uppercase letter, one lowercase letter, and one number;
- A password must be between 6 and 15;
- And cannot yet contain values in sequence, such as "01", "ab"or "AB".

Your task is: given a password, determine if it is valid or invalid considering the criteria presented.

Input

An entry contains a string S $(1 \leqslant length(S) \leqslant 20)$ corresponding to the password entered by the user.

Output

The output contains a single line with the phrase $True.$ (with the first letter capitalized and the point at the end) if the password meets the criteria, or $False.$ (with the first letter capitalized and the point at the end) if contrary. Finish with a line break.

Example of Input 1	Example of Output 1
`abCE13245`	`False.`

Example of Input 2	Example of Output 2
`aceF13592`	`True.`

Esta página foi propositadamente deixada em branco.

MARATONA DE PROGRAMAÇÃO
InterFatecs

Problema D

Números Amigos

Arquivo fonte: numerosamigos.{ c | cpp | java | py }
Autor: Anderson V. de Araujo (UFMS)

Maria, que está no quinto ano, acha a segunda-feira o melhor dia da semana porque tem aula de matemática. Hoje na aula a professora decidiu introduzir o conceito de números amigos para os alunos. Ela explicou: “Dois números são considerados amigos se a soma de seus divisores próprios resulta no outro número. Por exemplo, a soma dos divisores próprios de 220 é 284, e a soma dos divisores de 284 é 220. Então 220 e 284 são números amigos”. A professora complementou: "Os divisores próprios de um número positivo N são todos os divisores inteiros positivos de N exceto o próprio N".

Maria ficou muito impressionada com esse conceito e perguntou à professora: "Professora, será que é possível estender esse conceito para mais de dois números? Por exemplo, eu pego um número e faço a soma de seus divisores. Com este número eu faço a mesma coisa, mas não volta ao primeiro, entretanto eu continuo fazendo a soma dos divisores e acabo chegando no primeiro número, fechando assim como se fosse um "círculo de números amigos". A professora ficou sem reação, pois não sabia responder a pergunta de Maria, mas prometeu a ela que iria procurar a resposta e dizer a ela na próxima aula. Maria, muito impaciente, decidiu correr atrás da resposta ela mesma, mas ela sabia que existiam muitos números e que não tinha tanto tempo para testar todos. Foi quando ela decidiu pedir ajuda a um colega de classe, no caso você, para ajudá-la. Ela quer que você faça um programa que, dado um número inicial e o tamanho N da sequência, dizer se, a partir deste número inicial, é possível formar um círculo de amigos de tamanho N.

Entrada

Cada caso de teste contém o número N do tamanho da sequência, tal que $(2 \leqslant N \leqslant 100)$, em seguida o número inicial da sequência L, tal que $(0 \leqslant L \leqslant 10^6)$, separados por um espaço em branco.

Saída

A saída deve dizer se, a partir do número inicial, é possível formar um circulo de números amigos de acordo com os exemplos de entrada e saída. Finalize com uma quebra de linha.

Exemplo de Entrada 1	Exemplo de Saída 1
5 12496	sim

Exemplo de Entrada 2	Exemplo de Saída 2
3 220	nao

Esta página foi propositadamente deixada em branco.

MARATONA DE PROGRAMAÇÃO
InterFatecs

Problema E

Primorials

Arquivo fonte: primorial.{ c | cpp | java | py }
Autor: Antonio Cesar de Barros Munari (Fatec Sorocaba)

Um número que é o produto dos N primeiros números primos é chamado de primorial. Por exemplo, se tomarmos os 4 primeiros números primos (2, 3, 5 e 7) e os multiplicarmos, teremos como resultado 210, que é então um número primorial. Observe que no cálculo de um primorial cada primo é empregado uma única vez, o que explica porque, por exemplo, 630 não é primorial: precisaríamos utilizar o 3 na multiplicação duas vezes. Seu objetivo neste problema é bem simples: dado um inteiro informado, determinar se ele é ou não um número primorial.

Entrada

A entrada é composta por um único caso de teste, expresso em uma linha contendo um inteiro X ($0 \leqslant X \leqslant 2^{63} - 1$).

Saída

O programa deve imprimir uma letra maiúscula "S" se o valor informado é um número primorial e "N" em caso contrário. Finalize com uma quebra de linha.

Exemplo de Entrada 1	Exemplo de Saída 1
210	S

Exemplo de Entrada 2	Exemplo de Saída 2
510510	S

Exemplo de Entrada 3	Exemplo de Saída 3
2	S

Exemplo de Entrada 4	Exemplo de Saída 4
630	N

Exemplo de Entrada 5	Exemplo de Saída 5
9223372036854775807	N

Esta página foi propositadamente deixada em branco.

MARATONA DE PROGRAMAÇÃO
InterFatecs

Problema F

Regras

Arquivo fonte: regra.{ c | cpp | java | py }
Autor: Antonio Cesar de Barros Munari (Fatec-Sorocaba)

Você foi contratado como estagiário em uma empresa de tecnologia que trabalha com detecção de padrões nos dados dos clientes. Mas o caminho para o maravilhoso mundo do Big Data é espinhoso e você precisará se mostrar à altura do desafio. Seu primeiro teste, que sempre é o mais simples, pra não desanimar, consiste em determinar, para um conjunto de strings, quais os pares de letras que são considerados frequentes, ou seja, que costumam aparecer juntas na mesma palavra. O critério para se determinar se um par de letras é frequente é um clássico desse novo mundinho $fashion$: precisa atender às especificações de suporte e confiança requeridas pelo usuário. Para entender melhor esses dois conceitos, vamos usar um exemplo. Considere as palavras BRASIL, FATEC, SP, MARTELO e PARQUES e o par de letras 'A' e 'R', que será expresso como sendo o par ordenado (A, R), que é diferente do par (R, A). Temos um total de 5 palavras e o suporte de (A, R) indica o percentual de palavras em que encontramos tanto a letra 'A' como a letra 'R'. No caso, temos 3 palavras (BRASIL, MARTELO e PARQUES) em 5, ou seja, 3/5 = 0.6. Portanto, dizemos que o suporte do par (A, R) é 0.6 ou 60 por cento. Observe que o suporte do par (A, R) é o mesmo do par (R, A). A confiança, por sua vez, indica qual a probabilidade de a palavra que contém a primeira letra do par ter também a segunda. Em nosso caso, qual a probabilidade de uma palavra que tem a letra 'A' ter também a letra 'R'? O cálculo é bastante simples: basta dividir a quantidade de palavras que possuem as duas letras pela quantidade de palavras que possuem a primeira delas. Como vimos, temos 3 palavras com 'A' e 'R', e no conjunto são 4 palavras que possuem a letra 'A' (BRASIL, FATEC, MARTELO e PARQUES). Assim, a confiança do par (A, R) é dada por 3/4 = 0,75, ou seja, 75 por cento. Perceba que a confiança do par (R, A) pode ter um valor diferente do par (A, R): temos apenas 3 palavras com a letra 'R' (BRASIL, MARTELO e PARQUES), então a confiança é dada pela proporção 3/3, que corresponde a 1.0 ou 100 por cento. O seu desafio consiste em, para um conjunto de palavras fornecido e um valor de suporte e confiança informados, indicar todos os pares que podem ser considerados como frequentes.

Entrada

A entrada é composta por um único caso de teste, composto por várias linhas. Na primeira linha será encontrado um inteiro T $(1 \leqslant T \leqslant 10000)$ que representa a quantidade de palavras do conjunto. Na segunda linha é informado um real de precisão dupla S $(0.0 < S \leqslant 1.0)$ que indica o suporte mínimo que o par deve atender para ser considerado frequente. Na terceira linha temos um real de precisão simples C $(0.0 < C \leqslant 1.0)$ que expressa a confiança mínima esperada de um par frequente. Seguem-se então T linhas cada uma contendo uma $string$ de até 26 caracteres alfabéticos maiúsculos que são as palavras do conjunto de dados a ser processado pelo seu programa.

Saída

O programa deve imprimir todos os pares que atendem aos requisitos de suporte e confiança indicados, respeitando o formato apresentado nos exemplos. Os valores apresentados entre colchetes são, respectivamente, a confiança e o suporte, sempre com 3 casas após o ponto decimal e calculados na forma de números reais de precisão dupla. Os pares devem ser apresentados na ordem lexicográfica de seus termos. Caso não existam pares de letras que atendam aos requisitos, imprimir a mensagem "nada foi encontrado", em

minúsculas. Finalize com uma quebra de linha.

Exemplo de Entrada 1

```
5
0.600
0.400
BRASIL
FATEC
SP
MARTELO
PARQUES
```

Exemplo de Saída 1

```
A -> E [0.750 0.600]
A -> R [0.750 0.600]
E -> A [1.000 0.600]
R -> A [1.000 0.600]
```

Exemplo de Entrada 2

```
8
0.400
0.200
SAOPUL
SORCAB
FATEC
BRASIL
SAOBENT
ATLEICO
ITU
JAU
```

Exemplo de Saída 2

```
A -> O [0.571 0.500]
A -> S [0.571 0.500]
O -> A [1.000 0.500]
S -> A [1.000 0.500]
```

Exemplo de Entrada 3

```
8
0.600
0.300
SAOPUL
SORCAB
FATEC
BRASIL
SAOBENT
ATLEICO
ITU
JAU
```

Exemplo de Saída 3

```
nada foi encontrado
```

MARATONA DE PROGRAMAÇÃO
InterFatecs

Problema G

Robô

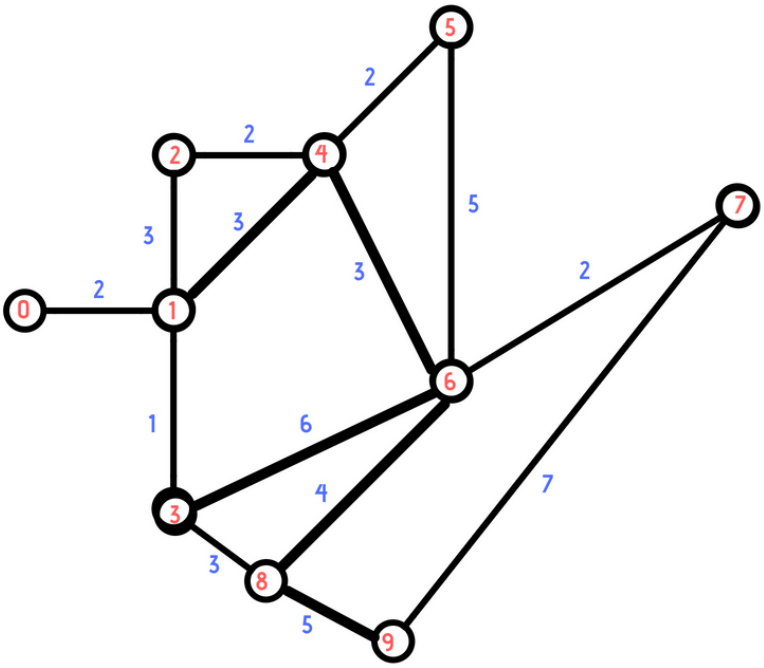
Arquivo fonte: robo.{ c | cpp | java | py }
Autor: Érico de Souza Veriscimo (ETEC de Guaianazes)

A ETEC de Guaianazes é um Polo de Robótica do Centro Paula Souza e, por isso, é sede de algumas competições, inclusive o Desafio de Robótica, que consiste em criar um robô autônomo que segue linhas e supera obstáculos.

Para evitar falhas na hora do julgamento, os professores responsáveis pela robótica estão pensando em ter uma solução computacional para o julgamento. E como você é um excelente programador(a), os professores solicitarão sua ajuda.

Na nova etapa da competição, o robô deve seguir a linha entre um ponto e outro, escolhendo sempre o menor caminho possível.

Toda pista é formada por linhas com cruzamentos, curvas e encruzilhadas. O Robô deve ser autônomo e sempre seguir pelo menor caminho possível. A figura seguinte mostra uma pista que contém encruzilhadas, curvas e cruzamentos numerados dentro de círculos e sobre as linhas as distâncias de um ponto a outro.



O robô sempre iniciará do valor 0 (zero) e deve chegar a um ponto determinado pelo juiz.

Sua tarefa é mostrar qual é o menor caminho pelo qual o robô deve seguir. Para o exemplo da pista representada na imagem, tal que o robô inicia no marco 0 (zero) e vai até o marco 7, o caminho que o mesmo deve seguir é 0 > 1 > 4 > 6 > 7.

Entrada

A entrada consiste de um único caso de teste. A primeira linha contém três inteiros, sendo um inteiro E $(2 \leqslant E \leqslant 1500)$ representando a quantidade de pontos (encruzilhadas, curvas e cruzamentos, um inteiro Q $(1 \leqslant Q \leqslant 10^5)$ representando a quantidade de ligações (fita que liga um ponto ao outro) e, em seguida, um inteiro F $(1 \leqslant F \leqslant E)$ representando o ponto final do circuito. As E linhas seguintes serão compostas por três inteiros A $(0 \leqslant A \leqslant E)$, B $(0 \leqslant B \leqslant E)$, D $(0 \leqslant D \leqslant 10^6)$ representando que o ponto A tem ligação com o ponto B e distância D.

Saída

Para cada caso de teste imprima a ordem no pontos que o robô deve seguir para chegar no ponto final utilizando o menor percurso. Finalize com uma quebra de linha.

Exemplo de Entrada 1

```
10 14 7
0 1 2
1 2 3
1 3 1
1 4 3
2 4 2
3 6 6
3 8 3
4 6 3
4 5 2
5 6 5
6 8 4
6 7 2
7 9 7
9 8 5
```

Exemplo de Saída 1

```
0 1 4 6 7
```

Exemplo de Entrada 2

```
6 8 5
0 1 1
0 2 1
1 4 2
1 3 2
2 3 3
2 4 3
4 5 1
3 5 4
```

Exemplo de Saída 2

```
0 1 4 5
```

MARATONA DE PROGRAMAÇÃO
InterFatecs

Problema H

Pronto Atendimento

Arquivo fonte: prontoatendimento.{ c | cpp | java | py }
Autor: Anderson V. de Araujo (UFMS)

Dione Bigud, professor de programação de uma escola de programação e robótica, teve que levar a sua filha no pronto atendimento de um hospital particular da região. Ao chegar no local ele ficou impressionado com a maneira com que os pacientes eram ordenados e chamados para atendimento de acordo com o risco definido no pré-atendimento. Enquanto ele esperava, ficou pensando nisso. Durante a espera, Dione entendeu o funcionamento:

- Pacientes com o risco VERMELHO passam todos os outros pacientes e vão para o final da lista de pacientes com o risco VERMELHO;
- Pacientes com o risco LARANJA passam todos os outros pacientes menos os do risco VERMELHO e vão para o final da lista de pacientes com o risco LARANJA;
- Pacientes com o risco AMARELO passam todos os outros pacientes menos os do risco VERMELHO e LARANJA e vão para o final da lista de pacientes com o risco AMARELO;
- Pacientes com o risco VERDE passam os outros pacientes do risco AZUL e vão para o final da lista de pacientes com o risco VERDE;
- Pacientes com o risco AZUL não passam nenhum outro nível de risco e vão para o final da lista de pacientes.

A cada vez que uma campainha tocava "beep" o nome do primeiro paciente de acordo com os critérios de risco aparecia no painel do hospital.

Ao chegar na escola em que dá aulas Dione entrou e já pediu um exercício para os alunos valendo nota. A tarefa é implementar um sistema que faz a classificação de acordo com a prioridade e define qual será o próximo paciente a ser atendido. Você que é aluno de Dione tem que entregar a tarefa até amanhã!

Entrada

A entrada contém 2*N linhas, onde N corresponde ao número de pacientes a serem atendidos, tal que $(2 \leqslant N \leqslant 10^5)$, ordenados por ordem de chegada. Cada paciente vem em uma linha com a senha (K), tal que $(0 < K \leqslant 10^5)$ e o risco (R) que pode ser uma das opções: VERMELHO, LARANJA, AMARELO, VERDE e AZUL. O risco vem separado da senha por um espaço em branco. Não existem senhas duplicadas. Entre as senhas estarão misturadas palavras "beep" indicando que um paciente foi chamado naquele momento. Não pode ser chamado um paciente ("beep") quando não existir nenhum paciente esperando. A entrada termina com o fim do arquivo.

Saída

A saída deve imprimir a lista ordenada de senhas dos pacientes de acordo com a prioridade definida pelo hospital e ordem de chegada. Finalize com uma quebra de linha.

Exemplo de Entrada 1

```
1 VERDE
2 AZUL
3 AMARELO
4 LARANJA
5 VERMELHO
beep
beep
beep
beep
beep
```

Exemplo de Saída 1

```
5
4
3
1
2
```

Exemplo de Entrada 2

```
10 AMARELO
12 VERMELHO
beep
33 AZUL
34 AMARELO
beep
beep
51 VERDE
beep
beep
```

Exemplo de Saída 2

```
12
10
34
51
33
```

MARATONA DE PROGRAMAÇÃO
InterFatecs

Problema I

Universos Paralelos

Arquivo fonte: universos.{ c | cpp | java | py }
Autor: Antonio Cesar de Barros Munari (Fatec-Sorocaba)

"The God Themselves" é um livro de ficção científica muito interessante escrito por Isaac Asimov em 1972. Aqui no Brasil teve pelo menos duas traduções, uma mais antiga, que recebeu o título de "O despertar dos deuses" e outra, mais recente, intitulada "Os próprios deuses". A trama descreve a existência de um universo paralelo, que acaba se comunicando com o nosso de uma forma meio precária, mas que permite que ocorra uma espécie de troca de energia entre eles, naquilo que na obra é chamado pomposamente de Bomba de Elétrons Entre Universos. Frederik Hallam é o controvertido personagem a que é atribuído o título de Pai da Bomba de Elétrons e que goza do prestígio terráqueo de ter resolvido o problema da geração de energia em nosso planeta por meio do processo intitulado para-fusão, que é uma forma de materialização entre os dois universos, e que teve início por volta do ano 2070. Bem, acontece que temos fortes indícios de que a obra do grande mago da ficção científica era, na verdade, uma premonição. Descobrimos na Fatec que há não apenas dois universos, mas uma infinidade deles! E há alguma forma de ligação entre eles, de maneira um pouco semelhante à retratada no livro. Inclusive, recebemos de certos para-homens (habitantes de universos paralelos) alguns diagramas sigilosos que mostram a forma como algumas ligações entre universos já conhecidas estão definidas. A principal característica dessas ligações é que elas possuem apenas um sentido de direção (uma passagem do Universo A para o Universo B, por exemplo, onde o trajeto em direção contrária é impossível). Percebemos também que nem todos os universos apresentados no esquema que recebemos dos para-homens estão ligados de forma direta. Na verdade, percebemos que a maioria das ligações entre universos é indireta. Estamos então no momento estudando todo esse material e pretendemos publicar nossas descobertas em breve, mas como estamos muito ocupados, precisaremos de uma ajudinha da sua parte. Dada uma representação indicando os universos e a forma como eles estão interligados, queremos saber se, a partir de um universo é possível atingir qualquer um dos universos restantes. Só isso e você poderá fazer parte da História, sendo apontado como um dos co-autores do nosso revolucionário trabalho!

Entrada

A entrada consiste de vários casos de teste. A primeira linha de um caso de teste contém dois inteiros U e L, representando respectivamente a quantidade de universos $(1 \leqslant U \leqslant 3000)$ e de ligações contidas no esquema recebido $(1 \leqslant L \leqslant 90000)$. Os universos são identificados por números de 1 a U. Cada uma das L linhas seguintes contém dois inteiros X e Y, separados por um espaço em branco, representando a existência de uma ligação conhecida que permite ir do universo X para o universo Y $(1 \leqslant X \leqslant U, 1 \leqslant Y \leqslant U$ e $X \neq Y)$. O final da entrada é indicado por $U = N = 0$.

Saída

Para cada caso de teste seu programa deve imprimir uma letra maiúscula "S" se for possível viajar de qualquer universo para todos os outros e "N" em caso contrário. Finalize com uma quebra de linha.

Exemplo de Entrada 1	Exemplo de Saída 1
3 4 2 3 3 2 1 2 3 1 3 4 1 2 3 2 1 3 2 3 0 0	S N