ToPAS'26
Torneio de Programação para Alunos do Secundário

Departamento de Ciência de Computadores

`https://topas.dcc.fc.up.pt`

Conjunto de Problemas

Faculdade de Ciências da Universidade do Porto

8 de maio de 2026

Este conjunto de problemas deverá conter oito (8) problemas e dezoito (18) páginas.
Se faltar algum problema, por favor avise a organização.

Edição realizada em colaboração com:

Departamento de Informática, Faculdade de Ciências e Tecnologia | NOVA FCT, Universidade NOVA de Lisboa

Departamento de Engenharia Eletrónica e Informática, Faculdade de Ciências e Tecnologia, Universidade do Algarve

ToPAS'26
Torneio de Programação para Alunos do Secundário

Departamento de Ciência de Computadores – FCUP
8 de maio de 2026

Conteúdo

Que Idades Têm?

A Ana tem dois irmãos. Curiosamente, os três nasceram nos primeiros dias de janeiro e têm a mesma diferença de idades. Primeiro, nasceu a Ana; passados D anos, nasceu a Bela; D anos após o nascimento da Bela, nasceu o Chico.

Sabendo a diferença de idades e a soma das idades dos três irmãos, consegue descobrir as idades deles?

Por exemplo:

- Se a diferença de idades for 3 e a soma das idades for 21, a Ana tem 10 anos, a Bela tem 7 e o Chico tem 4.
- Se a diferença de idades for 2 e a soma das idades for 6, a Ana tem 4 anos, a Bela tem 2 e o Chico tem 0 (nasceu este ano).

Tarefa

Escreva um programa que, dadas a diferença de idades e a soma das idades dos três irmãos, calcula as idades deles. Pode assumir que os exemplos testados correspondem a casos reais.

Input

O input tem duas linhas, cada uma com um número inteiro. A primeira linha tem a diferença D de idades. A segunda linha tem a soma S das idades dos três irmãos.

Restrições

$1 \leq D \leq 12$ Diferença de idades

$3 \leq S \leq 99$ Soma das três idades

Output

O output tem três linhas, cada uma com um número inteiro. A primeira linha tem a idade da Ana, a segunda linha tem a idade da Bela e a terceira linha tem a idade do Chico.

Exemplo 1

Input

```
3
21
```

Output

```
10
7
4
```

Exemplo 2

Input

```
2
6
```

Output

```
4
2
0
```

Exemplo 3

Input

```
4
30
```

Output

```
14
10
6
```

Exemplo 4

Input

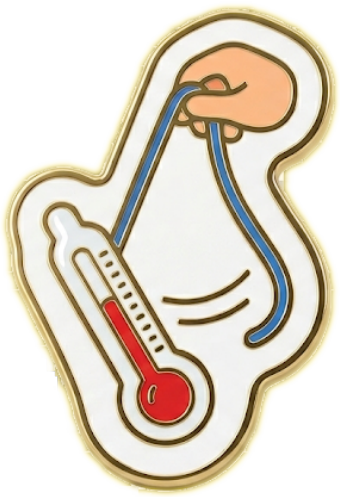
```
9
48
```

Output

```
25
16
7
```

A Dinâmica do Termóstato

Um termóstato é um dispositivo que controla a temperatura e a mantém (quase) constante. Etimologicamente, "termóstato" provém das palavras gregas "thermos", que significa calor, e "statós", que significa fixo ou estático; ou seja, significaria calor estático. No entanto, um termóstato mantém a temperatura num determinado intervalo, entre um máximo e um mínimo, e não propriamente num valor estático.



Por exemplo, o termóstato dum frigorífico liga a refrigeração quando a temperatura interna atinge um valor máximo, mas só a desliga quando a temperatura atinge um valor mínimo. Se estes valores fossem os mesmos, a refrigeração estaria constantemente a ser ligada e desligada. Assim, ligando e desligando mais espaçadamente a refrigeração, aumenta-se o tempo de vida do aparelho. Ou seja, o termóstato, cujo nome sugere calor estático, imprime-lhe alguma dinâmica!

Tarefa

Escreva um programa para funcionar como termóstato de um frigorífico. Assuma que inicialmente a refrigeração está desligada. Quando o programa inicia, começa por ler os valores mínimo e máximo do intervalo pretendido. Seguidamente, vai recebendo a leitura da temperatura do termómetro e liga ou desliga a refrigeração, conforme necessário, até receber indicação para terminar.

Input

A primeira linha do input contém dois inteiros, A e B, que são os valores mínimo e máximo do intervalo pretendido. Seguem-se S linhas, que definem a sequência de valores da temperatura recebidos do termómetro. Cada uma dessas linhas tem um número inteiro, C, que corresponde ao valor da temperatura em décimas de grau Celsius. Depois, há uma linha com o número `999`.

De notar que a sequência poderá ser vazia, caso em que a linha inicial será imediatamente seguida da linha com o número `999`.

Restrições

$-500 \leq A \leq B \leq 500$	Valores mínimo e máximo do intervalo pretendido
$0 \leq S \leq 25$	Número de valores da temperatura
$-1000 < C < 999$	Valor de uma temperatura

Output

Por cada valor da temperatura recebido do termómetro, deve ser escrita uma das seguintes palavras:

"`ON`"	Para ligar a refrigeração, se esta estiver desligada e a temperatura lida for superior ao valor máximo;
"`OFF`"	Para desligar a refrigeração, se esta estiver ligada e a temperatura lida for inferior ao valor mínimo;
"`KEEP`"	Se for desnecessário alterar o estado da refrigeração.

Para reagir ao primeiro valor da temperatura, assume-se que a refrigeração está inicialmente desligada.

Exemplo 1

Input	Output
60 60	ON
61	OFF
59	ON
61	OFF
59	ON
61	
999	

Exemplo 2

Input	Output
55 65	KEEP
64	KEEP
62	KEEP
61	KEEP
58	KEEP
57	KEEP
56	
999	

Exemplo 3

Input	Output
55 65	ON
70	KEEP
66	KEEP
61	KEEP
55	OFF
54	KEEP
53	KEEP
53	KEEP
57	KEEP
58	KEEP
62	KEEP
65	ON
66	
999	

Explicação

No exemplo 1, os valores mínimo e máximo são iguais. A refrigeração é alternadamente ligada e desligada.

No exemplo 2, os valores estão sempre dentre os limites e a refrigeração nunca é ligada, já que inicialmente está desligada.

No exemplo 3, a refrigeração é ligada no início até a temperatura ser inferior ao valor mínimo, quando é desligada, voltando a ser ligada quando é ultrapassado o valor máximo.

Associação de Estudantes

Na escola secundária ToPAS vai realizar-se uma eleição para a associação de estudantes, com duas listas a concurso, a lista A e a lista B. Cada aluno pode votar em apenas uma lista, mas nem todos os votos são iguais. Há um peso associado a cada voto, que depende do ano que o aluno frequenta: o voto de cada estudante conta N vezes, onde N é o ano que ele frequenta.

Terminada a votação, a comissão eleitoral gostaria de uma ajuda para indicar qual a lista vencedora, se houver, ou se há um empate entre as listas, o que levará a nova votação. Consegue ajudar?

Tarefa

Escreva um programa que, dado um conjunto de votos, cada um contendo informação sobre o ano que o aluno frequenta e a lista em que votou, indique a lista vencedora, caso haja, ou reporte a existência de um empate, caso contrário.

Input

A primeira linha do input tem um número inteiro V, que especifica o número de votantes. Seguem-se V linhas, uma para cada um dos votantes. Cada uma dessas linhas contém um número inteiro N e uma letra L, onde N indica o ano que o aluno frequenta e L é o nome da lista em que votou. A letra L é `A` ou `B`.

Restrições

$1 \leq V \leq 50$ Número de votantes

$10 \leq N \leq 12$ Ano que um aluno frequenta

Output

O output tem uma linha com o nome da lista vencedora, se houver, ou a palavra "`EMPATE`", caso contrário.

Exemplo 1

Input

```
4
10 A
10 B
12 B
10 A
```

Output

```
B
```

Exemplo 2

Input

```
4
10 A
11 B
11 B
12 A
```

Output

```
EMPATE
```

Exemplo 3

Input

```
5
12 B
12 A
11 B
11 A
10 A
```

Output

```
A
```

Exemplo 4

Input

```
11
12 B
10 A
10 A
12 B
10 A
12 B
12 B
10 A
12 B
10 A
10 A
```

Output

```
EMPATE
```

Exemplo 5

Input

```
3
10 A
11 A
12 A
```

Output

```
A
```

Volta dos BitAminados

A Volta dos BitAminados em bicicleta chegou ao fim! Depois de subidas exigentes, sprints calculados ao milissegundo e decisões tomadas com precisão... digital, é tempo de atribuir as tão desejadas camisolas aos melhores BitAminados. No final de cada etapa há três camisolas em jogo:

- **Amarela**: para o ciclista com o menor tempo total.
- **Verde**: para o ciclista com mais pontos.
- **Azul**: para o ciclista com mais pontos de montanha.

No entanto, o regulamento é claro: cada ciclista só pode vestir uma camisola. Se um ciclista liderar mais do que uma classificação, veste a camisola mais importante, sendo as restantes atribuídas aos ciclistas seguintes ainda sem camisola. Em caso de empate em qualquer classificação, a camisola é atribuída ao ciclista com menor número de dorsal (critério de desempate mais justo em qualquer competição séria)! Todos os números de dorsal são distintos.

Tarefa

Dada a lista de ciclistas da Volta dos BitAminados, determine quem veste cada uma das três camisolas, respeitando as seguintes regras:

1. A Camisola Amarela é atribuída ao ciclista com o menor tempo total.
2. A Camisola Verde é atribuída ao ciclista com o maior número de pontos, excluindo o ciclista que já veste a camisola amarela.
3. A Camisola Azul é atribuída ao ciclista com o maior número de pontos de montanha, excluindo os ciclistas que já vestem a camisola amarela e a camisola verde.
4. Em caso de empate em qualquer camisola, vence o ciclista com dorsal mais baixo.
5. Há sempre pelo menos três ciclistas, garantindo que todas as camisolas são atribuídas.

Input

A primeira linha contém um inteiro N, que corresponde ao número de ciclistas. Seguem-se N linhas, cada uma com 5 valores separados por um espaço:

$$nome\ \ dorsal\ \ tempoTotal\ \ pontos\ \ pontosMontanha$$

onde *nome* é uma palavra sem espaços, com entre 1 e 10 letras maiúsculas ou minúsculas, sem acentos nem cedilhas, *dorsal*, *pontos* e *pontosMontanha* são números inteiros e *tempoTotal* está no formato $hh:mm$. O número de horas (hh) varia entre `00` e `99` e o número de minutos (mm) entre `00` e `59`. Assim, o *tempoTotal* varia entre `00:00` e `99:59`. Ciclistas diferentes têm números de dorsal diferentes.

Restrições

$3 \leq N \leq 100$	Número de ciclistas
$1 \leq dorsal \leq 100$	Número de dorsal de um ciclista
$0 \leq pontos \leq 999$	Número de pontos de um ciclista
$0 \leq pontosMontanha \leq 999$	Número de pontos de montanha de um ciclista

Output

O output tem três linhas, com o seguinte formato e ordem, onde *nome* e *dorsal* são o nome e o número de dorsal dos ciclistas que vestem as respetivas camisolas:

`AMARELA` *nome dorsal*
`VERDE` *nome dorsal*
`AZUL` *nome dorsal*

Exemplo

Input

```
5
Rita 12 02:10 80 10
Tiago 7 02:08 60 25
Sofia 20 02:09 90 5
Nuno 3 02:30 85 30
Lia 15 03:00 70 30
```

Output

```
AMARELA Tiago 7
VERDE Sofia 20
AZUL Nuno 3
```

Planeamento de Tarefas Aero-Espaciais

Como programador da empresa TOPAS (*Tecnologias de Organização de Projetos Aeroespaciais e Satélites*), pretende-se que desenvolva um programa para auxiliar o planeamento dos lançamentos espaciais.

Cada projeto será dividido em N tarefas, numeradas de 1 a N. Algumas tarefas só podem ser iniciadas após a conclusão de outras, devido a dependências técnicas e de segurança.

Uma *proposta* para a ordem de execução das tarefas é uma sequência com N números, onde cada número de 1 a N ocorre exatamente uma vez:

$$t_1\ t_2\ \cdots\ t_N.$$

Uma proposta *respeita todas as dependências* se, executando as tarefas pela ordem indicada (primeiro t_1, depois t_2, etc.), quando se vai executar a tarefa t_i, todas as tarefas que tinham de estar concluídas antes de t_i já foram executadas (para todo o $i = 1, 2, \ldots, N$).

Tarefa

Escreva um programa que leia as dependências entre as tarefas, bem como várias propostas para a ordem de execução das mesmas e indique se cada uma dessas propostas respeita todas as dependências ou não.

Input

A primeira linha do input tem um inteiro N, que representa o número de tarefas do projeto. Seguem-se N linhas com a informação sobre as dependências. A i-ésima dessas linhas diz respeito à tarefa i (com $i = 1, 2, \ldots, N$) e tem o formato:

$$k\ X_1\ X_2\ \ldots\ X_k$$

significando que a tarefa i só pode ser iniciada depois das tarefas $X_1, X_2, \ldots, X_k$ estarem concluídas. Note que o primeiro número (k) não identifica uma tarefa; especifica quantas tarefas (diferentes) vêm a seguir. É possível que uma tarefa não dependa de nenhuma outra e nesses casos a linha contém um único 0 (o valor de k).

Depois há uma linha com um inteiro M, que representa o número de propostas para a ordem de execução das tarefas. Cada uma das restantes M linhas tem N números, que definem uma proposta, garantindo-se que cada número de 1 a N ocorre exatamente uma vez.

Restrições

$3 \leq N < 50$ Número de tarefas

$1 \leq M < 50$ Número de propostas

$0 \leq k < N$ Número de tarefas numa linha com informação sobre dependências

Output

O output deve ser constituído por M linhas, uma por proposta, com a palavra "`YES`" ou "`NO`", conforme a respetiva proposta respeite todas as dependências ou não.

Exemplo 1

Input

```
4
0
2 1 3
1 1
1 3
3
1 3 2 4
1 2 3 4
1 3 4 2
```

Output

```
YES
NO
YES
```

Explicação

Há quatro tarefas com as seguintes dependências:

- A tarefa 1 não depende de qualquer outra;
- A tarefa 2 depende das tarefas 1 e 3 (i.e. as tarefas 1 e 3 têm de estar concluídas antes de se iniciar a tarefa 2);
- A tarefa 3 depende da tarefa 1;
- A tarefa 4 depende da tarefa 3.

A primeira e a terceira propostas respeitam todas as dependências. A segunda proposta não respeita porque a tarefa 2 só pode começar depois das tarefas 1 e 3 terem sido executadas.

Exemplo 2

Input

```
7
1 3
2 6 1
0
1 5
0
2 1 4
0
3
3 1 5 4 6 2 7
5 4 3 6 7 1 2
7 5 3 4 2 1 6
```

Output

```
YES
NO
NO
```

A Festa de Aniversário

O João faz anos na próxima sexta-feira e quer convidar os seus amigos para a festa de aniversário. Os avós e os pais do João querem que tudo corra na perfeição.

A festa vai ser em casa, onde há espaço para 6 adultos e 6 jovens: tirando um adulto, ganha-se espaço para dois jovens. Por exemplo, se houver 4 adultos, é possível ter 10 jovens. Os pais do João têm de estar na festa. Os avós estão dispostos a não ir à festa se o João quiser convidar mais amigos. Considere que, neste problema, os avós do João são sempre dois adultos. Nunca poderá haver mais de 6 adultos na festa.

O João quer convidar alguns dos seus amigos, mas faz questão que na festa haja um número igual de raparigas e de rapazes ou, em alternativa, apenas rapazes. Estas restrições não se aplicam aos adultos; só se aplicam aos amigos.

Alguns amigos só vão à festa se outras pessoas também forem. Por exemplo, há quem só vá com a sua mãe (se a sua mãe também for) e há quem só vá com outro amigo (se esse amigo do João também for).

Assuma que o João tem os seguintes 17 amigos (onde F representa rapariga e M representa rapaz), cujas exigências são:

- Ana (F, só com os seus pais)
- Pedro (M, só com o seu pai)
- Mario (M, só com o Miguel)
- Maria (F, só com a sua mãe)
- Ines (F)
- Miguel (M)
- Natalia (F, só com os avós do João)
- Antonio (M)
- Fatima (F)
- Marco (M, só com a Ana)
- Leonor (F)
- Duarte (M)
- Luis (M, só com a Ana)
- Tomas (M, só com a sua mãe)
- Rui (M, só com o Pedro)
- Afonso (M)
- Beatriz (F)

Tarefa

Escreva um programa que indique se um conjunto de jovens pode ser convidado para ir à festa (porque satisfaz todas as regras enunciadas acima), no caso dos avós irem à festa e no caso de não irem.

Input

A primeira linha do input tem um inteiro N, que especifica o número de jovens do conjunto. Cada uma das restantes N linhas tem o nome de um jovem, que é uma palavra sem espaços, com entre 1 e 10 letras maiúsculas ou minúsculas, sem acentos nem cedilhas. Linhas diferentes têm nomes diferentes.

Restrições

$1 \leq N \leq 20$ Número de jovens do conjunto

Output

O output é constituído por duas linhas, cada uma com a palavra "`YES`" ou "`NO`", consoante o conjunto de jovens possa ser convidado para ir à festa, ou não possa. Na primeira linha, assume-se que os avós vão à festa. Na segunda linha, assume-se que os avós não vão à festa.

Exemplo 1

Input

```
2
Ana
Francisco
```

Output

```
NO
NO
```

Exemplo 2

Input

```
5
Ana
Pedro
Mario
Ines
Miguel
```

Output

```
NO
NO
```

Exemplo 3

Input

```
6
Ana
Pedro
Mario
Ines
Miguel
Fatima
```

Output

```
NO
YES
```

Exemplo 4

Input

```
8
Ana
Antonio
Fatima
Ines
Leonor
Mario
Miguel
Marco
```

Output

```
NO
YES
```

Exemplo 5

Input

```
4
Luis
Mario
Ines
Fatima
```

Output

```
NO
NO
```

Explicação

No exemplo 1, o Francisco não é um amigo do João.

No exemplo 2, não há um número igual de raparigas e de rapazes, nem há apenas rapazes.

No exemplo 3, com os avós, há 7 adultos, mas na casa só há espaço para 6 adultos. Sem os avós, todas as regras são cumpridas.

No exemplo 4, com os avós, só há espaço para 6 jovens, porque há 6 adultos. Sem os avós, há espaço para 10 jovens e todas as regras são cumpridas.

No exemplo 5, o Luis só vai se a Ana também for.

O Paleontólogo Curioso

Um paleontólogo quer explorar um terreno retangular cheio de fósseis. O terreno é representado por uma matriz de inteiros não negativos. Se uma posição da matriz tiver um número positivo, existe um e um único fóssil com esse valor nesse local. Nas posições da matriz que têm zero não existem fósseis.

O objetivo do paleontólogo é efetuar um percurso no terreno, recolhendo os fósseis que se encontram nos locais por onde passa. Mas, como tem de colocar os fósseis numa caixa especial, para não os danificar, não pode recolher mais fósseis do que o número de divisórias da caixa. Caso não possa colocar todos os fósseis que encontra na caixa, o paleontólogo recolherá apenas os fósseis mais valiosos.

O paleontólogo começa sempre o percurso na posição $(0, 0)$ da matriz, que corresponde ao canto superior esquerdo do terreno (o canto mais a Norte e mais a Oeste). Como não tem um sentido de orientação apurado, pediu ajuda ao seu amigo de IA, que definiu o percurso a efetuar no terreno através de uma sequência de direções. Cada uma dessas direções é N, S, E ou O, que indica que o paleontólogo deve mover-se uma posição para Norte, Sul, Este ou Oeste, respetivamente. O seu amigo de IA engana-se com alguma frequência. Por isso, o paleontólogo decidiu estar atento e ignorar qualquer direção que o levasse para fora dos limites do terreno.

Sempre que o paleontólogo se move para uma nova posição onde exista um fóssil, pode optar por uma das seguintes ações:

- Recolher esse fóssil, colocando-o na caixa, podendo retirar algum fóssil da caixa;
- Ignorar esse fóssil, deixando-o no terreno.

Quando retira um fóssil da caixa para o substituir por outro que encontrou, o fóssil retirado deve ser colocado no terreno exatamente na posição onde encontrou o novo.

Note que o paleontólogo escolhe sempre a ação que lhe permite maximizar a soma dos valores dos fósseis recolhidos (aqueles que se encontram na caixa) quando o percurso termina. A posição $(0, 0)$, que é a primeira posição visitada pelo paleontólogo, pode conter um fóssil e o trajeto pode passar várias vezes pela mesma posição do terreno.

Tarefa

Escreva um programa que, dados o número de divisórias da caixa (que é o número máximo de fósseis que podem ser recolhidos), a configuração do terreno e a sequência de direções que descreve o percurso, indica a posição final do paleontólogo, a soma dos valores dos fósseis na caixa e o conjunto de fósseis na caixa, especificando as posições no terreno onde foram encontrados e os seus valores. Os fósseis serão listados pela ordem em que foram recolhidos.

Input

O input tem $L + 2$ linhas, sendo L o número de linhas da matriz que representa o terreno. A primeira linha tem três números inteiros positivos, que representam, respetivamente, o número D de divisórias da caixa, o número L de linhas e o número C de colunas da matriz.

De seguida existem L linhas, cada uma delas com uma sequência de C valores inteiros f_{ij}, separados por um espaço, que indicam se a posição (i, j) do terreno tem um fóssil e, caso tenha, qual é o seu valor (para $i = 0, \ldots, L - 1$ e $j = 0, \ldots, C - 1$). Os valores dos fósseis são todos diferentes (ou seja, não há dois fósseis com valores iguais).

A última linha tem o percurso definido pelo amigo de IA, que é uma sequência de P direções. Cada direção é o carácter `N`, `S`, `E` ou `O`.

Restrições

$1 \leq D \leq 50$	Número de divisórias da caixa
$1 \leq L \leq 100$	Número de linhas da matriz
$1 \leq C \leq 100$	Número de colunas da matriz
$0 \leq f_{ij} \leq 100\,000$	Valor na posição (i, j) da matriz (com $0 \leq i < L$ e $0 \leq j < C$)
$1 \leq P \leq 11\,000$	Número de direções no percurso

Output

O output tem $F + 1$ linhas, sendo F o número de fósseis na caixa. Todas as linhas têm três inteiros, separados por um espaço: x, y e v. Os dois primeiros números definem a posição (x, y) do terreno: x é o número da linha e y é o número da coluna.

Na primeira linha, (x, y) é a posição final do paleontólogo e v é a soma dos valores dos fósseis na caixa, quando o percurso termina. Cada uma das restantes linhas refere-se a um fóssil na caixa: (x, y) é a posição no terreno onde o fóssil foi encontrado e v é o seu valor. As últimas F linhas têm de aparecer pela ordem em que os fósseis foram recolhidos.

Exemplo 1

Input

```
1 3 4
0 10 20 0
0 0 30 0
0 50 0 0
EESNNO
```

Output

```
0 1 30
1 2 30
```

Exemplo 2

Input

```
2 3 4
0 10 20 0
0 0 30 0
0 50 0 0
ESNNO
```

Output

```
0 0 10
0 1 10
```

Exemplo 3

Input

```
2 3 4
0 10 20 0
0 0 0 30
0 50 0 0
SSNEESE
```

Output

```
2 3 0
```

Exemplo 4

Input

```
3 2 4
100 10 20 0
0 0 120 30
EESOEE
```

Output

```
1 3 250
0 0 100
1 2 120
1 3 30
```

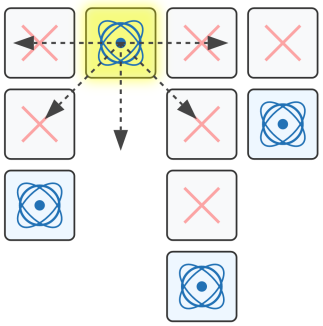
Uma Quinta Muito Bem Organizada

O agricultor Edgar tem muito orgulho na sua quinta. Passa muito tempo a cuidar das suas plantações e culturas, consciente da importância que a rega tem em todo este processo. Recentemente, o agricultor Edgar adquiriu um sistema de rega de última geração que agora pretende instalar. Sendo um grande adepto da organização e da eficiência, Edgar já concebeu o melhor método para instalar o seu novo sistema de rega.

A quinta do agricultor Edgar está organizada como uma matriz de N colunas, em que a primeira linha é comum a todas as colunas. No entanto, o antigo proprietário cimentou uma parte do terreno, porque gostava muito de histogramas, e as colunas podem ter um número de linhas *diferente* entre si. O novo sistema de rega é composto por N aspersores, que devem ser cuidadosamente instalados. Por questões de otimização do abastecimento de água às diferentes culturas, o agricultor estipulou as seguintes regras de instalação:

- Todas as colunas devem conter um, e um só, aspersor;
- Todas as linhas devem conter um, e um só, aspersor;
- Dois aspersores não podem estar colocados numa mesma diagonal da matriz.

Para exemplificar, suponha que a quinta do agricultor Edgar tem a forma esquematizada na figura abaixo. É uma matriz com quatro colunas, em que a primeira coluna tem três linhas, a segunda coluna tem uma linha, a terceira coluna tem quatro linhas e a última coluna tem duas linhas.



Nesta quinta, existe apenas uma maneira de colocar os quatro aspersores, como se ilustra na figura. As setas colocadas no segundo aspersor indicam as direções em que a água será pulverizada. Por isso, nenhum outro aspersor poderá ser colocado no caminho destas setas.

Se a quinta fosse uma matriz quadrada regular (4 linhas em cada uma das 4 colunas), existiriam duas alternativas diferentes para instalar os aspersores.

Tarefa

Escreva um programa que, dada a descrição da quinta do agricultor Edgar, indica o número de alternativas diferentes para instalar o sistema de rega.

Input

A primeira linha do input contém um inteiro N, que representa o número de colunas da quinta (que é igual ao número de aspersores a instalar). De seguida, há N linhas, cada uma com um inteiro L_i, que representa o número de linhas da i-ésima coluna (com $i = 1, \ldots, N$). Garante-se que alguma coluna tem N linhas.

Restrições

$1 \leq N \leq 10$ Número de colunas

$1 \leq L_i \leq N$ Número de linhas da i-ésima coluna (com $i = 1, \ldots, N$)

Output

O output tem uma única linha, com um inteiro que representa o número de alternativas diferentes para instalar os aspersores.

Exemplo 1

Input

```
4
3
1
4
2
```

Output

```
1
```

Exemplo 2

Input

```
4
4
4
4
4
```

Output

```
2
```

Exemplo 3

Input

```
3
3
3
3
```

Output

```
0
```

Exemplo 4

Input

```
8
8
8
8
8
8
8
8
8
```

Output

```
92
```

Explicação

Os dois primeiros exemplos correspondem aos casos referidos na página anterior.