Competidor(a): _____

Número de inscrição: _____–_____ *(opcional)*

OBI2018

Caderno de Tarefas

Modalidade **Programação** • **Nível 2** • Fase **Local**

17 de maio de 2018

A PROVA TEM DURAÇÃO DE **2** HORAS

Promoção:

Sociedade Brasileira de Computação

Apoio:

v1.0

Instruções

LEIA ATENTAMENTE ESTAS INSTRUÇÕES ANTES DE INICIAR A PROVA

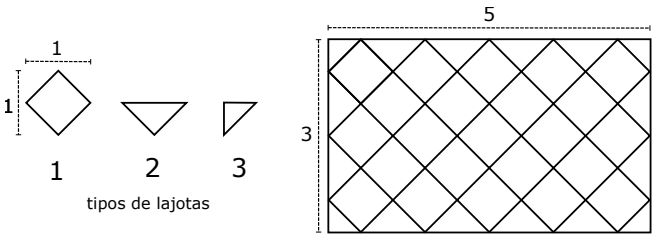
- Este caderno de tarefas é composto por 6 páginas (não contando a folha de rosto), numeradas de 1 a 6. Verifique se o caderno está completo.
- A prova deve ser feita individualmente.
- É proibido consultar a Internet, livros, anotações ou qualquer outro material durante a prova. É permitida a consulta ao *help* do ambiente de programação se este estiver disponível.
- As tarefas têm o mesmo valor na correção.
- A correção é automatizada, portanto siga atentamente as exigências da tarefa quanto ao formato da entrada e saída de seu programa; em particular, seu programa não deve escrever frases como "Digite o dado de entrada:" ou similares.
- Não implemente nenhum recurso gráfico nas suas soluções (janelas, menus, etc.), nem utilize qualquer rotina para limpar a tela ou posicionar o cursor.
- As tarefas **não** estão necessariamente ordenadas, neste caderno, por ordem de dificuldade; procure resolver primeiro as questões mais fáceis.
- Preste muita atenção no nome dos arquivos fonte indicados nas tarefas. Soluções na linguagem C devem ser arquivos com sufixo *.c*; soluções na linguagem C++ devem ser arquivos com sufixo *.cc* ou *.cpp*; soluções na linguagem Pascal devem ser arquivos com sufixo *.pas*; soluções na linguagem Java devem ser arquivos com sufixo *.java* e a classe principal deve ter o mesmo nome do arquivo fonte; soluções na linguagem Python 2 devem ser arquivos com sufixo *.py2*; soluções na linguagem Python 3 devem ser arquivos com sufixo *.py3*; e soluções na linguagem Javascript devem ter arquivos com sufixo *.js*.
- Na linguagem Java, **não** use o comando *package*, e note que o nome de sua classe principal deve usar somente letras minúsculas (o mesmo nome do arquivo indicado nas tarefas).
- Para tarefas diferentes você pode escolher trabalhar com linguagens diferentes, mas apenas uma solução, em uma única linguagem, deve ser submetida para cada tarefa.
- Ao final da prova, para cada solução que você queira submeter para correção, copie o arquivo fonte para o seu diretório de trabalho ou pen-drive, conforme especificado pelo seu professor.
- Não utilize arquivos para entrada ou saída. Todos os dados devem ser lidos da entrada padrão (normalmente é o teclado) e escritos na saída padrão (normalmente é a tela). Utilize as funções padrão para entrada e saída de dados:
  - em Pascal: *readln, read, writeln, write*;
  - em C: *scanf, getchar, printf, putchar*;
  - em C++: as mesmas de C ou os objetos *cout* e *cin*.
  - em Java: qualquer classe ou função padrão, como por exemplo *Scanner, BufferedReader, BufferedWriter* e *System.out.println*
  - em Python: *read,readline,readlines,input,print,write*
  - em Javascript: *scanf, printf*
- Procure resolver a tarefa de maneira eficiente. Na correção, eficiência também será levada em conta. As soluções serão testadas com outras entradas além das apresentadas como exemplo nas tarefas.

Piso da escola

Nome do arquivo: "`piso.x`", *onde* `x` *deve ser* `cpp`, `pas`, `java`, `js`, `py2` *ou* `py3`

O colégio pretende trocar o piso de uma sala de aula e a diretora aproveitou a oportunidade para passar uma tarefa aos alunos. A sala tem o formato de um retângulo de largura L metros e comprimento C metros, onde L e C são números inteiros. A diretora precisa comprar lajotas de cerâmica para cobrir todo o piso da sala. Seria fácil calcular quantas lajotas seriam necessárias se cada lajota fosse um quadrado de 1 metro de lado. O problema é que a lajota que a diretora quer comprar é um quadrado que possui 1 metro de diagonal, não de lado. Além disso, ela quer preencher o piso da sala com as diagonais das lajotas alinhadas aos lados da sala, como na figura.

A loja vai fornecer lajotas do tipo 1: inteiras; do tipo 2, que correspondem à metade das do tipo 1, cortadas ao longo da diagonal; e lajotas do tipo 3, que correspondem à metade do tipo 2. Veja os três tipos de lajotas na figura.



Está muito claro que sempre serão necessárias 4 lajotas do tipo 3 para os cantos da sala. A tarefa que a diretora passou para os alunos é calcular o número de lajotas dos tipos 1 e 2 que serão necessárias. Na figura, para $L = 3$ e $C = 5$, foram necessárias 23 do tipo 1 e 12 do tipo 2.

Seu programa precisa computar, dados os valores de L e C, a quantidade de lajotas do tipo 1 e do tipo 2 necessárias.

Entrada

A primeira linha da entrada contém um inteiro L indicando a largura da sala. A segunda linha contém um inteiro C representando o comprimento da sala.

Saída

Imprima duas linhas na saída. A primeira deve conter um inteiro, representando o número de lajotas do tipo 1 necessárias. A segunda deve conter um inteiro, indicando o número de lajotas do tipo 2.

Restrições

- $1 \leq L, C \leq 100$

Exemplo de entrada 1	**Exemplo de saída 1**
3 5	23 12

Exemplo de entrada 2	**Exemplo de saída 2**
1 1	1 0

Figurinhas da copa

Nome do arquivo: "`figurinhas.x`", *onde* `x` *deve ser* `cpp`, `pas`, `java`, `js`, `py2` *ou* `py3`

Em ano de Copa do Mundo de Futebol, o álbum de figurinhas oficial é sempre um grande sucesso entre crianças e também entre adultos. Para quem não conhece, o álbum contém espaços numerados de 1 a N para colar as figurinhas; cada figurinha, também numerada de 1 a N, é uma pequena foto de um jogador de uma das seleções que jogará a Copa do Mundo. O objetivo é colar todas as figurinhas nos respectivos espaços no álbum, de modo a *completar* o álbum (ou seja, não deixar nenhum espaço sem a correspondente figurinha).

Algumas figurinhas são *carimbadas* (efetivamente têm um carimbo impresso sobre a fotografia do jogador) e são mais raras, mais difíceis de conseguir.

As figurinhas são vendidas em envelopes fechados, de forma que o comprador não sabe quais figurinhas está comprando, e pode ocorrer de comprar uma figurinha que ele já tenha colado no álbum. Para ajudar os usuários, a empresa responsável pela venda do álbum e das figurinhas quer criar um aplicativo que permita gerenciar facilmente as figurinhas que faltam para completar o álbum.

Dados o número total de espaços e figurinhas do álbum (N), a lista das figurinhas carimbadas e uma lista das figurinhas já compradas (que pode conter figurinhas repetidas), sua tarefa é determinar quantas *figurinhas carimbadas* faltam para *completar* o álbum.

Entrada

A primeira linha contém três números inteiros N, C e M indicando respectivamente o número de figurinhas (e espaços) do álbum, o número de figurinhas carimbadas do álbum e o número de figurinhas já compradas. A segunda linha contém C números inteiros distintos X_i indicando as figurinhas carimbadas do álbum. A terceira linha contém M números inteiros Y_i indicando as figurinhas já compradas.

Saída

Seu programa deve produzir um inteiro representando o número de *figurinhas carimbadas* que falta para completar o álbum.

Restrições

- $1 \leq N \leq 100$
- $1 \leq C \leq N/2$
- $1 \leq M \leq 300$
- $1 \leq X_i, Y_i \leq N$

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

`10 2 5`
`4 7`
`7 1 2 8 3`
	`1`

Exemplo de entrada 2	**Exemplo de saída 2**

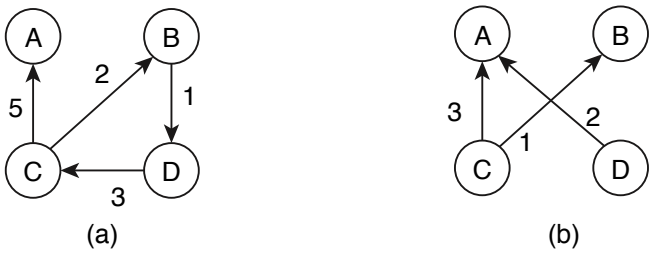
`10 2 6`
`4 7`
`7 1 8 4 9 3`
	`0`

Exemplo de entrada 3	**Exemplo de saída 3**
8 4 10 2 4 6 8 3 1 1 5 9 1 7 7 1 1	4

Câmara de Compensação

Nome do arquivo: "`compensacao.x`", *onde* `x` *deve ser* `cpp`, `pas`, `java`, `js`, `py2` *ou* `py3`

Em uma cidade, muitas pessoas emprestam dinheiro para outras pessoas. A coisa chegou a um tal ponto que tem gente que é ao mesmo tempo devedor e credor. As pessoas resolveram então pagar suas dívidas e cada uma emitiu os cheques para pagar suas dívidas. Por exemplo, na figura, item (a), a pessoa C emitiu um cheque de 5 dinheiros para a pessoa A, e a pessoa D emitiu um cheque de 3 dinheiros para a pessoa C. Ou seja, a pessoa C recebeu da pessoa D e pagou a pessoa A. Pior ainda, existe um ciclo vicioso, em que a pessoa D emitiu um cheque de 3 dinheiros para a pessoa C, que por sua vez emitiu um cheque de 2 dinheiros para a pessoa B, que por sua vez emitiu um cheque de 1 dinheiro para a pessoa D. A situação mostrada no item (a) da Figura abaixo é descrita através de uma *lista de cheques*, com quatro triplas da forma (X, V, Y), para indicar que X emitiu um cheque de V dinheiros para Y. Na mesma Figura, no item (b), a situação é descrita com uma lista de apenas três cheques.



Entretanto, as duas listas são equivalentes: o saldo na conta bancária de uma pessoa é o mesmo em ambas as listas de cheques. Em ambos os casos, completada a compensação de todos os cheques, a pessoa A terminará com 5 dinheiros a mais na sua conta, a pessoa B terminará com 1 dinheiro a mais na sua conta, a pessoa C terminará com 4 dinheiros a menos na sua conta e a pessoa D terminará com 2 dinheiros a menos na sua conta.

Vamos então definir equivalência de listas de cheques emitidos: duas listas de cheques são *equivalentes* se, ao final do processo de compensação de todos os cheques, o seguinte vale para cada pessoa: seu saldo bancário ao final da compensação de uma lista é o mesmo que o saldo bancário da pessoa ao final da compensação da outra lista.

O total de valores compensados no item (a) da figura é igual a 11 dinheiros ao passo que no item (b) o total é de apenas 6 dinheiros!

Este problema tem duas subtarefas:

- *Subtarefa A*: determinar, dada uma lista de cheques, se é possível ou não diminuir o total de valores compensados utilizando uma outra lista de cheques equivalente.
- *Subtarefa B*: determinar o total mínimo de valores compensados em uma lista de cheques equivalente.

Você deve escrever um programa que resolva apenas a Subtarefa A ou que resolva as duas subtarefas.

Entrada

A primeira linha contém dois inteiros, M e N, onde M é o número de cheques emitidos e N é o número de habitantes da cidade. Os habitantes são identificados por números inteiros de 1 a N. Cada uma das M linhas seguintes descreve um cheque da lista e contém três inteiros X, V e Y, que indica que X emitiu um cheque de V dinheiros a favor de Y. É possível que haja mais de um cheque de X a Y. Também é possivel que haja cheques de X a Y e de Y a X, mas não de X a X.

Saída

Seu programa deve produzir duas linhas na saída. A primeira linha descreve a resposta para a Subtarefa A e deve conter um único caractere. O caractere deve ser `S` para indicar que é possível diminuir o total dos cheques compensados com uma lista de cheques equivalente, ou `N` para indicar que não é possível diminuir o total de cheques compensados.

Se o seu programa resolve também a Subtarefa B, a segunda linha descreve a resposta para essa subtarefa e deve conter um número inteiro, o valor mínimo do total de cheques compensados, em uma lista equivalente. Se o seu programa não resolve a Subtarefa B, você pode deixar a linha em branco ou colocar um valor inteiro arbitrário.

Restrições

- $1 \leq M \leq 10^6$
- $2 \leq N \leq 10^3$
- $1 \leq X \leq N$, $1 \leq Y \leq N$, $X \neq Y$
- $1 \leq V \leq 10^2$

Informações sobre a pontuação

- Subtarefa A: 20 pontos.
- Subtarefa B: em um conjunto de casos de testes que vale 20 pontos $1 \leq N \leq 10$.

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

Exemplo de entrada 2	**Exemplo de saída 2**
5 4 4 50 3 2 25 1 3 10 2 2 100 1 4 50 3	S 215

Exemplo de entrada 3	**Exemplo de saída 3**
4 4 3 10 1 2 40 1 2 30 4 2 20 4	N 100