Competidor(a): _____

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

OBI2020

Caderno de Tarefas

Modalidade **Programação** • **Nível 2** • Fase **Nacional**

28 de novembro de 2020

A PROVA TEM DURAÇÃO DE **5** HORAS

Promoção:

Sociedade Brasileira de Computação

Apoio:

CNPq
Conselho Nacional de Desenvolvimento Científico e Tecnológico

Instruções

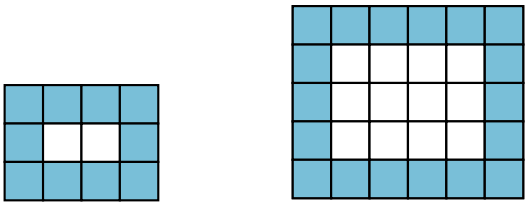
LEIA ATENTAMENTE ESTAS INSTRUÇÕES ANTES DE INICIAR A PROVA

- Este caderno de tarefas é composto por 11 páginas (não contando a folha de rosto), numeradas de 1 a 11. 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.

Atlanta

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

Documentos recentemente encontrados por pesquisadores mostram que na Sala de Audiências do palácio Real na cidade perdida de Atlanta o piso era formado por ladrilhos 20 cm x 20 cm. Ladrilhos de duas cores foram usados: o centro da Sala era formado por ladrilhos brancos e exatamente uma fileira de ladrilhos azuis foram colocados em cada lateral do Sala, como nas figuras abaixo.



Os pesquisadores não encontraram vestígios da Sala de Audiências (nem da cidade de Atlanta!), mas os documentos recentes, se forem autênticos, indicam também a quantidade de ladrilhos que foram utilizados no piso da Sala.

Sua tarefa é, dadas as quantidades de azulejos azuis e brancos, determinar as dimensões da Sala de Audiências.

Entrada

A primeira linha da entrada contém um inteiro A, o número de azulejos azuis. A segunda linha contém um número inteiro B, o número de azulejos brancos.

Saída

Seu programa deve produzir uma única linha, contendo dois números inteiros, representando as dimensões da Sala (largura e comprimento). Se a largura for diferente do comprimento, seu programa deve imprimir primeiro a menor dimensão, seguida da maior dimensão. Se as quantidades de azulejos não forem corretas para construir o piso da Sala no formato descrito acima, seu programa deve imprimir `-1 -1`.

Restrições

- $1 \leq A \leq 10^6$
- $1 \leq B \leq 10^6$

Exemplos

Exemplo de entrada 1	**Exemplo de saída 1**
`10` `2`	`3 4`

Exemplo de entrada 2	**Exemplo de saída 2**
`8` `2`	`-1 -1`

Exemplo de entrada 3	**Exemplo de saída 3**
3996 996004	1000 1000

Candidatas

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

Carla é uma biotecnóloga que está investigando proteínas para usar em uma vacina. As proteínas estão codificadas em uma longa sequência de números naturais, que vamos chamar de S.

Através de múltiplos experimentos, Carla determinou que uma subsequência contígua de S é *candidata* se o máximo divisor comum dos elementos da subsequência é maior do que 1.

Carla quer investigar o número de subsequências candidatas contidas em alguns intervalos contíguos de S, possivelmente também alterando alguns elementos de S. Mais precisamente, durante suas pesquisas ela deseja realizar operações de dois tipos:

1. alterar o valor de um elemento da sequência S; e
2. consultar o número de subsequências candidatas em um dado trecho contíguo de S.

Ajude Carla a realizar suas pesquisas e avançar na busca pela vacina!

Entrada

A primeira linha da entrada contém dois inteiros N e M, indicando respectivamente o número de elementos na sequência S e o número de operações a serem realizadas. A segunda linha contém N inteiros S_i, os elementos da sequência S. Os elementos são identificados por índices de 1 a N. Cada uma das M linhas seguintes descreve uma operação e contém três inteiros. O primeiro inteiro, T, indica o tipo de operação e pode ser 1 ou 2. Se a operação é do tipo 1, os dois números seguintes na linha são I e V, indicando que o elemento de índice I da sequência S deve ter o valor atualizado para V. Se a operação é do tipo 2, os dois números seguintes na linha são E e D, indicando respectivamente o índice do elemento inicial e o índice do elemento final de um intervalo contíguo da sequência S.

Saída

Para cada operação do tipo 2, seu programa deve produzir uma única linha, contendo um único inteiro, o número de subsequências candidatas no intervalo indicado.

Restrições

- $1 \leq N \leq 10^5$
- $1 \leq M \leq 10^5$
- $1 \leq S_i \leq 10^9$, para $1 \leq i \leq N$
- $1 \leq I \leq N$
- $1 \leq V \leq 10^9$
- $1 \leq E \leq D \leq N$

Exemplos

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

`5 1`
`9 3 4 8 1`
`2 2 5`
	`4`

Exemplo de entrada 2	**Exemplo de saída 2**
4 3 4 4 4 4 2 1 4 1 3 5 2 1 4	10 5

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

Rede social

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

Uma nova rede social foi lançada e fez sucesso imediato. Nessa nova rede é possível *postar* mensagens que são recebidas por *seguidores*; um seguidor pode decidir *repostar* uma mensagem que recebeu e seus seguidores também receberão a mensagem e poderão por sua vez repostá-la.

Para medir a *influência* de um usuário na nova rede foi criado um novo critério, chamado de Fator de Influência, descrito a seguir.

- Inicialmente vamos definir o *índice de repostagem* de uma mensagem M de um usuário U como sendo o número de usuários diferentes de U que repostaram M.
- O Fator de Influência de um usuário U é o máximo valor de *FI* tal que U postou *FI* mensagens que, cada uma, tem um índice de repostagem de pelo menos *FI*.

Por exemplo, se João postou quatro mensagens, com índices de repostagem 1, 1, 5, 6, seu Fator de Influência é 2, pois postou duas mensagens com índice de repostagem maior ou igual a 2.

Dada uma lista com os índices de repostagens de todas as mensagens postadas por um usuário, escreva um programa para calcular o Fator de Influência do usuário.

Entrada

A primeira linha da entrada contém um inteiro N, o número de mensagens postadas pelo usuário. Cada uma das N linhas seguintes contém um inteiro R_i, o índice de repostagem de uma mensagem.

Saída

Seu programa deve produzir uma única linha, contendo um único número inteiro, o Fator de Influência para o usuário.

Restrições

- $1 \leq N \leq 5 \times 10^5$
- $0 \leq R_i \leq 10^6$ para $1 \leq i \leq N$

Informações sobre a pontuação

- Para um conjunto de casos de testes valendo 20 pontos, $1 \leq N \leq 1000$.
- Para um conjunto adicional de casos de testes valendo 80 pontos, nenhuma restrição adicional.

Exemplos

Exemplo de entrada 1	**Exemplo de saída 1**
5 1 4 1 7 1	2

Exemplo de entrada 2	**Exemplo de saída 2**
5 12 5 3 5 15	4

Exemplo de entrada 3	**Exemplo de saída 3**
4 3 3 3 3	3

Jogo do Preto e Branco

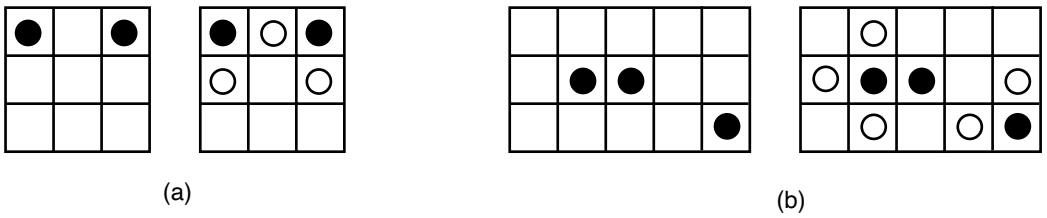
Nome do arquivo: "`jogo.x`", *onde* `x` *deve ser* `c`, `cpp`, `pas`, `java`, `js`, `py2.py` *ou* `py3.py`

Você gosta de quebra-cabeças? O jogo do Preto e Branco é um quebra-cabeças que usa um tabuleiro retangular com L linhas e C colunas, formando $L \times C$ casas. No tabuleiro são posicionadas algumas peças pretas, cada peça em uma casa diferente.

O objetivo do jogo é colocar o maior número possível de peças brancas no tabuleiro, obedecendo às seguintes restrições:

- cada casa do tabuleiro pode conter no máximo uma peça;
- uma peça branca deve ter ao menos uma peça preta como vizinha, à direita, à esquerda, acima ou abaixo;
- uma peça branca não pode ter outra peça branca como vizinha, à direita, à esquerda, acima ou abaixo;

A figura abaixo mostra dois exemplos de jogos, com as respectivas soluções, um com um tabuleiro 3×3 e outro com um tabuleiro 3×5.



Sua tarefa é escrever um programa que, dadas as descrições do tabuleiro e das peças pretas posicionadas, determine o maior número de peças brancas que podem ser colocadas.

Entrada

A primeira linha contém dois inteiros L e C, o número de linhas e o número de colunas do tabuleiro. As linhas são numeradas de 1 a L e as colunas são numeradas de 1 a C. A segunda linha contém um inteiro P, o número de peças pretas colocadas no tabuleiro. Cada uma das P linhas seguintes descreve a posição de uma peça preta e contém dois inteiros X_i e Y_i, indicando a linha e a coluna em que a peça foi colocada.

Saída

Seu programa deve produzir uma única linha, contendo um único inteiro, o maior número de peças brancas que podem ser colocadas no tabuleiro.

Restrições

- $1 \leq L \leq 6$
- $1 \leq C \leq 6$
- $1 \leq P \leq 10$
- $1 \leq X_i \leq L$ para $1 \leq i \leq L$
- $1 \leq Y_i \leq C$ para $1 \leq i \leq C$

Informações sobre a pontuação

- Para um conjunto de casos de testes valendo 20 pontos, $L = 1$.
- Para um conjunto adicional de casos de testes valendo 80 pontos, nenhuma restrição adicional.

Exemplos

Exemplo de entrada 1	**Exemplo de saída 1**
3 3 3 1 1 1 3 3 2	3

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

Exemplo de entrada 3	**Exemplo de saída 3**
3 5 3 2 2 2 3 3 5	5

Trem da mina

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

Uma antiga mina de ouro foi desativada e Herculano quer torná-la uma atração turística. A mina contém uma verdadeira rede ferroviária subterrânea, composta de estações e ramos de trilhos, pelos quais trafegavam os trens carregando minério. Cada ramo de trilho liga duas estações distintas e pode ser usado nas duas direções. Um "ciclo" na rede ferroviária é uma sequência de estações $s_1, s_2, \ldots, s_n, s_{n+1} = s_1$, tais que $s_i \neq s_{i+1}$ e (s_i, s_{i+1}) é um ramo de trilho, para $1 \leq i \leq n$. A rede ferroviária pode conter ciclos, mas cada estação faz parte de no máximo um ciclo da rede ferroviária. Os ramos de trilhos e estações são tais que, se uma parte do trem ocupa um ramo de trilho ou estação, não há espaço para outro (ou o mesmo!) trem entrar novamente nesse ramo de trilho ou estação.

Algumas estações da rede ferroviária têm acesso ao direto ao solo, para descarregar o minério. Herculano tem um mapa que descreve a rede ferroviária da mina, informando para cada ramo de trilho o seu comprimento e quais duas estações o ramo de trilho liga.

Para planejar o passeio turístico de trem pela mina Herculano quer saber, para as estações que têm acesso ao solo, conhecendo o comprimento do trem, se é possível que o trem entre na mina pela estação, percorra a menor distância possível dentro da mina e saia novamente pela mesma estação que entrou, sempre andando para a frente, sem nunca dar marcha-a-ré. Você pode ajudá-lo?

Entrada

A primeira linha contém dois inteiros E e R representando respectivamente o número de estações e o número de ramos de trilhos da rede ferroviária da mina. As estações são identificadass por inteiros de 1 a E. Cada uma das R linhas seguintes descreve um ramo de trilho e contém três inteiros A, B e C, onde A e B representam as estações ligadas pelo ramo de trilho, e C representa o comprimento do ramo de trilho. Uma estação é ligada por ramos de trilhos a no máximo outras 100 estações e cada duas estações são ligadas por no máximo um ramo de trilho. A próxima linha contém um inteiro K, que indica o número de consultas. Cada uma das K linhas seguintes descreve uma consulta, e contém dois inteiros X e T, que indicam respectivamente a estação pela qual Herculano quer que o trem entre e o comprimento do trem.

Saída

Para cada consulta da entrada seu programa deve produzir uma única linha, contendo um único número inteiro, o comprimento do percurso mínimo que o trem deve percorrer dentro da mina para entrar e sair pela estação indicada na consulta, sem dar marcha-a-ré. Se não for possível para o trem entrar e sair sem dar marcha-a-ré, a linha deve conter o valor -1.

Restrições

- $2 \leq E \leq 10^4$
- $1 \leq R \leq 2 \times E$
- $1 \leq A < B \leq E$
- $1 \leq C \leq 100$
- $1 \leq K \leq 100$
- $1 \leq X \leq E$
- $1 \leq T \leq 10^5$
- Uma estação é ligada por ramos de trilhos a no máximo outras 100 estações e cada duas estações são ligadas por no máximo um ramo de trilho.

Exemplos

Exemplo de entrada 1	**Exemplo de saída 1**
4 4 1 2 10 1 3 12 3 4 7 1 4 6 4 2 18 1 10 4 26 3 25	45 25 -1 25

Exemplo de entrada 2	**Exemplo de saída 2**
7 8 1 2 2 2 3 2 2 5 10 5 6 25 2 6 20 3 7 1 4 7 4 3 4 3 4 1 6 4 50 7 56 7 5	16 65 -1 8