Competidor(a): _____

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

Este Caderno de Tarefas não pode ser levado para casa após a prova. Após a prova entregue este Caderno de Tarefas para seu professor guardar. Os professores poderão devolver os Cadernos de Tarefas aos competidores após o término do período de aplicação das provas (15 de agosto de 2023).

Olimpíada Brasileira de Informática

OBI2023

Caderno de Tarefas

Modalidade Programação • Nível 1 • Fase 2 - Turno A

15 de agosto de 2023

A PROVA TEM DURAÇÃO DE 2 HORAS

Promoção:

Sociedade Brasileira de Computação

Apoio:

FUNDAÇÃO BEHRING

Coordenação:

INSTITUTO DE COMPUTAÇÃO UNICAMP

Instruções

LEIA ATENTAMENTE ESTAS INSTRUÇÕES ANTES DE INICIAR A PROVA

- Este caderno de tarefas é composto por 8 páginas (não contando a folha de rosto), numeradas de 1 a 8. 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 Java devem ser arquivos com sufixo *.java* e a classe principal deve ter o mesmo nome do arquivo fonte; soluções na linguagem Python 3 devem ser arquivos com sufixo *.py*; 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).
- Você pode submeter até 50 soluções para cada tarefa. A pontuação total de cada tarefa é a melhor pontuação entre todas as submissões. Se a tarefa tem sub-tarefas, para cada sub-tarefa é considerada a melhor pontuação entre todas as submissões.
- 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 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.

Prefixo

Nome do arquivo: `prefixo.c`, `prefixo.cpp`, `prefixo.java`, `prefixo.js` *ou* `prefixo.py`

Em programas de computador, *palavras*, como "otorrinolaringologista" ou "escolaridade", são normalmente representadas por uma cadeia de caracteres. Um *prefixo* de uma palavra é qualquer sequência de caracteres consecutivos que se inicia no primeiro caractere da palavra. Por exemplo, "escola" e "esco" são prefixos de "escola", mas "cola" não é.

Um *prefixo comum* de duas palavras é uma cadeia de caracteres que é prefixo das duas palavras. Por exemplo, "a", "aba" e "abaca" são exemplos de prefixos comuns das palavras "abacate" e "abacaxi".

Dadas duas palavras, escreva um programa para calcular o comprimento do maior prefixo comum, em número de caracteres.

Entrada

A primeira linha de entrada contém um inteiro N, o número de caracteres da primeira palavra. A segunda linha contém uma cadeia com N caracteres P_i, a primeira palavra. A terceira linha contém um inteiro M, o número de caracteres da segunda palavra. A quarta linha contém uma cadeia com M caracteres S_i, a segunda palavra.

Saída

Seu programa deve produzir uma única linha, contendo um único inteiro, o número de caracteres do maior prefixo que é comum às duas sequências.

Restrições

- $1 \leq N \leq 1\ 000$
- $1 \leq M \leq 1\ 000$
- As palavras contêm apenas letras minúsculas sem acentuação.

Informações sobre a pontuação

A tarefa vale 100 pontos. Estes pontos estão distribuídos em subtarefas, cada uma com suas **restrições adicionais** às definidas acima:

- **Subtarefa 1 (33 pontos):** $N = M$
- **Subtarefa 2 (67 pontos):** Nenhuma restrição adicional.

Seu programa pode resolver corretamente ambas ou somente uma das subtarefas. Sua pontuação final na tarefa é a soma dos pontos das subtarefas resolvidas corretamente por alguma das suas submissões.

Exemplos

Exemplo de entrada 1	**Exemplo de saída 1**
7 abacate 7 abacaxi	5

Explicação do exemplo 1: "abaca", com 5 caracteres, é o maior prefixo comum entre "abacate" e "abacaxi", portanto 5 é a resposta.

Exemplo de entrada 2	**Exemplo de saída 2**
8 paralelo 13 paralelogramo	8

Explicação do exemplo 2: "paralelo", com 8 caracteres, é o maior prefixo comum entre "paralelo" e "paralelogramo", portanto 8 é a resposta.

Exemplo de entrada 3	**Exemplo de saída 3**
6 escola 4 cola	0

Explicação do exemplo 3: "escola" e "cola", não têm prefixo comum, portanto a resposta é 0.

Grupos de Trabalho

Nome do arquivo: `grupos.c`, `grupos.cpp`, `grupos.java`, `grupos.js` *ou* `grupos.py`

A professora Paula divide a classe em grupos de três estudantes para os trabalhos da sua disciplina. Para minimizar descontentamentos, ela fez uma enquete no início do ano, de forma que ela tem uma lista de pares de estudantes que gostariam de estar no mesmo grupo, e uma lista de pares de estudantes que não gostariam de estar no mesmo grupo.

Para cada trabalho ela faz uma nova divisão de grupos, e claro que nem sempre vai ser possível satisfazer todas as restrições da classe!

Dados os pares de estudantes que gostariam estar no mesmo grupo, os pares de estudantes que não gostariam estar no mesmo grupo, e uma possível distribuição dos estudantes em grupos de três, sua tarefa é determinar o número total de restrições que são violadas com essa distribuição.

Entrada

A primeira linha contém três inteiros E, M e D, indicando, respectivamente, o número total de estudantes, o número de pares de estudantes que gostariam de estar no mesmo grupo e o número de pares de estudantes que não gostariam de estar no mesmo grupo. Os estudantes são identificados por números inteiros de 1 a E.

Cada uma das M linhas seguintes descreve um par de estudantes que gostariam de estar no mesmo grupo e contém dois inteiros X e Y indicando os estudantes do par. Cada uma das D linhas seguintes descreve um par de estudantes que não gostariam de estar no mesmo grupo e contém dois inteiros U e V indicando os estudantes do par.

Finalmente, cada uma das $E/3$ linhas seguintes descreve um grupo de estudantes e contém três inteiros I, J e K indicando os estudantes do grupo.

Saída

Seu programa deve produzir uma única linha, contendo um único inteiro, o número total de restrições que são violadas nos grupos da entrada.

Restrições

- $3 \leq E \leq 999\ 999$ e E é divisível por 3.
- $0 \leq M \leq 100\ 000$
- $0 \leq D \leq 100\ 000$
- $M + D > 0$ e, entre todos os $M + D$ pares, cada par de estudantes aparece no máximo uma vez.
- $1 \leq X \leq E$, $1 \leq Y \leq E$ e $X \neq Y$.
- $1 \leq U \leq E$, $1 \leq V \leq E$ e $U \neq V$.
- $1 \leq I \leq E$, $1 \leq J \leq E$ e $1 \leq K \leq E$
- Cada estudante aparece em exatamente um dos $E/3$ grupos.

Informações sobre a pontuação

A tarefa vale 100 pontos. Estes pontos estão distribuídos em subtarefas, cada uma com suas **restrições adicionais** às definidas acima:

- **Subtarefa 1 (39 pontos):** $E \leq 999$, $M \leq 1\,000$ e $D \leq 1\,000$.
- **Subtarefa 2 (61 pontos):** Nenhuma restrição adicional.

Seu programa pode resolver corretamente ambas ou somente uma das subtarefas. Sua pontuação final na tarefa é a soma dos pontos das subtarefas resolvidas corretamente por alguma das suas submissões.

Exemplos

Exemplo de entrada 1	**Exemplo de saída 1**
3 1 0 1 2 2 1 3	0

Explicação do exemplo 1: Há 3 estudantes e apenas uma restrição (estudantes 1 e 2 gostariam de estar no mesmo grupo), que é obedecida no único grupo da distribuição, (2,1,3), portanto a resposta é 0.

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

Explicação do exemplo 2: Há 9 estudantes e portanto serão formados três grupos. Há 1 par de estudantes que gostariam de estar no mesmo grupo: (1,9) e três pares de estudantes que não gostariam estar no mesmo grupo: (1,3), (5,6) e (2,8). Nos grupos formados – (1,2,3), (4,5,6) e (7,8,9) –, das quatro restrições apenas o par (2,8) tem a restrição obedecida (pois não estão no mesmo grupo). Para os outros três pares, (1,9), (1,3) e (5,6), as restrições são violadas, portanto a resposta é 3.

Exemplo de entrada 3	**Exemplo de saída 3**
6 0 3 1 5 5 2 2 3 5 2 1 3 4 6	2

Explicação do exemplo 3: Há 6 estudantes e portanto serão formados dois grupos. Não há nenhum par de estudantes que gostariam de estar no mesmo grupo e há três pares de estudantes que não gostariam estar no mesmo grupo: (1,5), (5,2) e (2,3). Nos grupos formados – (5,2,1) e (3,4,6) –, das três restrições, duas são violadas: (1,5) e (5,2) não gostariam de estar no mesmo grupo, e portanto a resposta é 2.

Intervalo Distinto

Nome do arquivo: `distinto.c`, `distinto.cpp`, `distinto.java`, `distinto.js` *ou* `distinto.py`

Você foi contratado pela Agência Extra-Espacial Brasileira, que procura indícios de vida extra-terrestre.

Um dos telescópios da Agência, para o espectro ultravioleta, gera uma sequência de valores inteiros positivos que devem ser analisados diariamente. Sua primeira missão é determinar, na sequência gerada, o tamanho do maior intervalo contínuo que contém apenas números distintos.

Entrada

A primeira linha contém um inteiro N, o número de elementos da sequência. Cada uma das linhas seguintes contém um inteiro I_i, os elementos da sequência na ordem em que foram gerados.

Saída

Seu programa deve produzir uma única linha, contendo um único inteiro, o número de elementos do maior intervalo que contém apenas números distintos.

Restrições

- $1 \leq N \leq 10^5$
- $1 \leq I_i \leq 10^5$

Informações sobre a pontuação

A tarefa vale 100 pontos. Estes pontos estão distribuídos em subtarefas, cada uma com suas **restrições adicionais** às definidas acima:

- **Subtarefa 1 (23 pontos):** $N \leq 100$
- **Subtarefa 2 (18 pontos):** $N \leq 5000$
- **Subtarefa 3 (15 pontos):** $I_i \leq 100$
- **Subtarefa 4 (44 pontos):** Nenhuma restrição adicional.

Seu programa pode resolver corretamente todas ou algumas das subtarefas acima *(elas não precisam ser resolvidas em ordem)*. Sua pontuação final na tarefa é a soma dos pontos de todas as subtarefas resolvidas corretamente por alguma das suas submissões.

Exemplos

Exemplo de entrada 1	**Exemplo de saída 1**
8 3 2 1 3 2 1 3 2	3

Explicação do exemplo 1: três intervalos, com três elementos cada, contêm números distintos: [3, 2, 1], [2, 1, 3] e [1, 3, 2] – note também que os intervalos ocorrem mais de uma vez na sequência. Como não há intervalo com números distintos maior do que esses, a resposta é 3.

Exemplo de entrada 2	**Exemplo de saída 2**
6 3 2 3 8 5 5	4

Explicação do exemplo 2: o maior intervalo que contém números distintos é [2, 3, 8, 5], com 4 elementos, portanto a resposta é 4.