Competidor(a): _____

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

OBI2021

Caderno de Tarefas

Modalidade Programação • Nível 1 • Fase 2

21 de agosto de 2021

A PROVA TEM DURAÇÃO DE 2 HORAS

Promoção:

Sociedade Brasileira de Computação

Apoio:

Instruções

LEIA ATENTAMENTE ESTAS INSTRUÇÕES ANTES DE INICIAR A PROVA

- Este caderno de tarefas é composto por 7 páginas (não contando a folha de rosto), numeradas de 1 a 7. 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 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).
- 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.

Duplas de tênis

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

Quatro amigos combinaram de jogar tênis em duplas. Cada um dos amigos tem um nível de jogo, que é representado por um número inteiro: quanto maior o número, melhor o nível do jogador.

Os quatro amigos querem formar as duplas para iniciar o jogo. De forma a tornar o jogo mais interessante, eles querem que os níveis dos dois times formados sejam o mais próximo possível. O nível de um time é a soma dos níveis dos jogadores do time.

Embora eles sejam muito bons jogadores de tênis, os quatro amigos não são muito bons em algumas outras coisas, como lógica ou matemática. Você pode ajudá-los e encontrar a menor diferença possível entre os níveis dos times que podem ser formados?

Entrada

A entrada contém quatro linhas, cada linha contendo um inteiro A, B, C e D, indicando o nível de jogo dos quatro amigos.

Saída

Seu programa deve produzir uma única linha, contendo um único inteiro, a menor diferença entre os níveis dos dois times formados.

Restrições

- $0 \leq A \leq B \leq C \leq D \leq 10^4$

Exemplos

Exemplo de entrada 1	**Exemplo de saída 1**
4 7 10 20	7

Exemplo de entrada 2	**Exemplo de saída 2**
0 0 1 1000	999

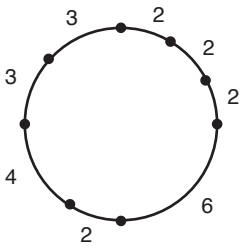
Exemplo de entrada 3	**Exemplo de saída 3**
1 2 3 4	0

Retângulo

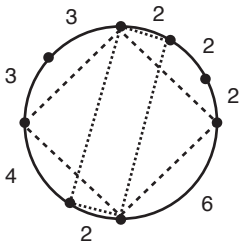
Nome do arquivo: "`retangulo.x`", *onde* `x` *deve ser* `c`, `cpp`, `pas`, `java`, `js` *ou* `py`

Vô Pedro é um fazendeiro meticuloso. Em sua fazenda ele tem uma plantação no formato circular, com algumas árvores plantadas exatamente na circunferência da plantação. A figura (a) abaixo mostra a plantação com as árvores.

Agora vô Pedro quer usar uma longa corda e quatro das árvores para demarcar um retângulo na plantação, usando as árvores como vértices, com a corda marcando os lados. A figura (b) abaixo mostra dois retângulos que podem ser demarcados usando as árvores na plantação figura (a).



(a)



(b)

Dada a descrição das posições das árvores na plantação circular de vô Pedro, sua tarefa é determinar se é possível demarcar um retângulo conforme descrito acima.

Entrada

A primeira linha da entrada contém um inteiro N indicando o número de árvores na circunferência da plantação. As árvores são representadas como pontos na circunferência. A segunda linha contém N inteiros $L_1, L_2, \ldots L_N$, indicando o comprimento do arco entre cada par de árvores consecutivas. Os arcos são dados no sentido anti-horário.

Saída

Seu programa deve produzir uma única linha, contendo um único caractere, que deve ser `S` se é possível demarcar um retângulo usando as árvores como vértices, ou `N` caso contrário.

Restrições

- $4 \leq N \leq 10^5$
- $1 \leq L_i \leq 10^6$ para $i = 1, 2, \ldots, N$

Informações sobre a pontuação

- Para um conjunto de casos de testes valendo 20 pontos, $N \leq 100$.
- Para um conjunto de casos de testes valendo outros 20 pontos, $N \leq 300$.
- Para um conjunto de casos de testes valendo outros 20 pontos, $N \leq 1000$.
- Para um conjunto de casos de testes valendo outros 40 pontos, nenhuma restrição adicional.

Exemplos

Exemplo de entrada 1	**Exemplo de saída 1**
8 3 3 4 2 6 2 2 2	S

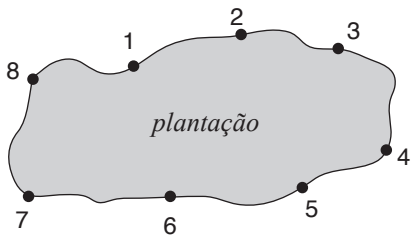
Exemplo de entrada 2	**Exemplo de saída 2**
4 14 16 15 15	N

Exemplo de entrada 3	**Exemplo de saída 3**
6 3 7 7 3 10 10	S

Robô

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

Um fazendeiro comprou um robô-espantalho para espantar os pássaros de sua plantação de milho. O robô se move ao longo de um caminho que circunda a plantação. No caminho há N estações numeradas sequencialmente, a partir de 1, no sentido horário. A figura abaixo mostra um exemplo com oito estações.



O robô inicia cada dia na estação número 1, e então obedece a uma sequência de comandos. Os comandos são gerados por um algoritmo de aprendizagem de máquina que coleta informações através de sensores espalhados na plantação, para garantir uma cobertura de vigia máxima. Cada comando faz com que o robô se mova para outra estação, vizinha à estação em que ele se encontra, ou no sentido horário ou no sentido anti-horário. O robô permanece nessa nova estação até receber um novo comando.

Apesar da promessa de que o robô protegeria a plantação, ao final de um determinado dia o fazendeiro notou que parte de sua plantação estava devastada por pássaros. O fazendeiro agora quer entender melhor o que aconteceu.

Dados o número da estação mais próxima à área devastada e a sequência de comandos que o robô obedeceu naquele dia, escreva um programa para determinar quantas vezes o robô permaneceu na estação mais próxima à àrea devastada.

Entrada

A primeira linha contém três inteiros N, C e S, representando respectivamente o número de estações, o número de comandos e o número da estação mais próxima à área devastada. A segunda linha contém C inteiros $X_1, X_2, \ldots, X_C$, representando a sequência de comandos recebidos pelo robô. Para $i = 1, 2, \ldots, C$, se X_i é 1 então o i-ésimo comando significa "mova-se para a próxima estação no sentido horário", enquanto se X_i é -1 então o i-ésimo comando significa "mova-se para a próxima estação no sentido anti-horário". O robô sempre inicia na estação número 1.

Saída

Seu programa deve produzir uma única linha, contendo um único inteiro, o número de vezes que o robô permaneceu na estação número S durante o dia.

Restrições

- $2 \leq N \leq 100$
- $1 \leq C \leq 1000$

Exemplos

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

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

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

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

Exemplo de entrada 5	**Exemplo de saída 5**
2 2 1 -1 1	2

Exemplo de entrada 6	**Exemplo de saída 6**
2 2 1 -1 -1	2

Média e mediana

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

A *média* de três números inteiros A, B e C é $(A + B + C)/3$. A *mediana* de três números inteiros é o número que ficaria no meio se os três números fossem ordenados em ordem não-decrescente.

Sua tarefa é escrever um programa que, dados dois números inteiros distintos A e B, calcule o menor inteiro possível C tal que a média e a mediana de A, B e C sejam iguais.

Entrada

A entrada é composta de uma única linha contendo dois números inteiros A e B.

Saída

Seu programa deve produzir uma única linha, contendo um único número, o menor inteiro possível C tal que a média e a mediana de A, B e C são iguais.

Restrições

- $1 \leq A \leq B \leq 10^9$

Exemplos

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

Exemplo de entrada 2	**Exemplo de saída 2**
`6 10`	`2`

Exemplo de entrada 3	**Exemplo de saída 3**
`1 1000000000`	`-999999998`