OBI2015

Caderno de Tarefas

Modalidade **Programação** • **Nível 1** • Fase **1**

29 de maio de 2015

A PROVA TEM DURAÇÃO DE **4** HORAS

Promoção:

Sociedade Brasileira de Computação

Apoio:

Instruções

LEIA ATENTAMENTE ESTAS INSTRUÇÕES ANTES DE INICIAR A PROVA

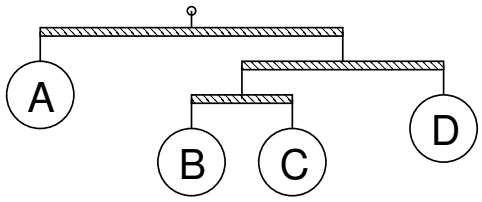
- Este caderno de tarefas é composto por 5 páginas (não contando a folha de rosto), numeradas de 1 a 5. Verifique se o caderno está completo.
- A prova deve ser feita individualmente.
- É proibido consultar a Internet, livros, anotações ou qualquer outro material durante a prova. É permitida a consulta 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.
- 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 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 devem ser arquivos com sufixo *.py*; e soluções na linguagem Javascript devem ter arquivos com sufixo *.js*. Para problemas diferentes você pode escolher trabalhar com linguagens diferentes, mas apenas uma solução, em uma única linguagem, deve ser submetida para cada problema.
- 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 o problema 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.

Móbile

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

O móbile na sala da Maria é composto de três hastes exatamente como na figura abaixo. Para que ele esteja completamente equilibrado, com todas as hastes na horizontal, os pesos das quatro bolas A, B, C e D têm que satisfazer todas as seguintes três condições:

1. $A = B + C + D$; e
2. $B + C = D$; e
3. $B = C$.



Nesta tarefa, dados os pesos das quatro bolas, seu programa deve decidir se o móbile está ou não completamente equilibrado.

Entrada

A entrada consiste de quatro linhas contendo, cada uma, um número inteiro, indicando os pesos das bolas. Os números são dados na ordem: A, B, C e D.

Saída

Seu programa deve escrever uma única linha na saída, contendo o caractere "`S`" se o móbile estiver equilibrado, ou o caractere "`N`" se não estiver equilibrado.

Restrições

- $1 \leq A, B, C, D \leq 1000$

Exemplos

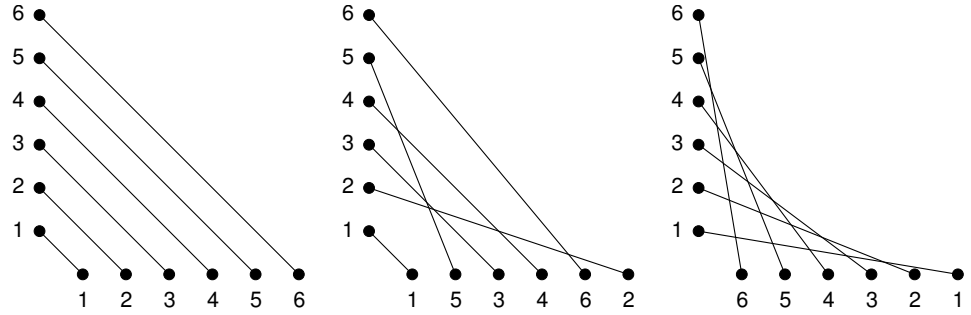
Entrada	**Saída**
12 3 3 6	S

Entrada	**Saída**
2002 560 560 882	N

Linhas Cruzadas

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

Uma das atividades de recreação preferidas de Letícia é compor desenhos com linhas coloridas esticadas entre preguinhos numa base de madeira. Quanto mais cruzamentos entre pares de linhas, mais interessante fica a figura. Neste problema temos N pregos na vertical e N pregos na horizontal, como na figura abaixo. Os pregos na vertical possuem uma numeração fixa, de 1 a N, de baixo para cima. Os pregos na horizontal também são numerados de 1 a N, mas a ordem pode ser qualquer uma. Letícia vai sempre esticar uma linha entre cada par de pregos que tiverem o mesmo número. Dada a ordem dos pregos horizontais, seu programa deve computar o número total de cruzamentos entre pares de linhas no desenho de Letícia. Por exemplo, os três desenhos da figura possuem, respectivamente, 0, 6 e 15 cruzamentos.



Entrada

A primeira linha da entrada contém um número natural N. A segunda linha contém N números naturais distintos de 1 a N, representando a ordem dos pregos na horizontal.

Saída

Seu programa deve escrever uma linha na saída, contendo o número de cruzamentos entre pares de linhas, conforme a descrição anterior.

Restrições

- $2 \leq N \leq 60000$

Informações sobre a pontuação

- Em um conjunto de casos de teste somando 60 pontos, $N \leq 30000$

Exemplos

Entrada	**Saída**
6 1 5 3 4 6 2	6

Entrada	**Saída**
15 5 8 15 12 2 1 9 7 4 11 14 10 3 6 13	49

Fita Colorida

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

Roberto tem um conjunto de lápis com 10 tons diferentes de uma mesma cor, numerados de 0 a 9. Numa fita quadriculada, alguns quadrados foram coloridos inicialmente com o tom 0. Roberto precisa determinar, para cada quadrado Q não colorido, qual é a distância dele para o quadrado mais próximo de tom 0. A distância entre dois quadrados é definida com o número mínimo de movimentos para a esquerda, ou para a direita, para ir de um quadrado para o outro. O quadrado Q, então, deve ser colorido com o tom cuja numeração corresponde à distância determinada. Se a distância for maior ou igual a 9, o quadrado deve ser colorido com o tom 9. Seu programa deve colorir e imprimir a fita quadriculada dada na entrada.

Entrada

A primeira linha da entrada contém apenas um inteiro N, indicando o número de quadrados da fita. A segunda linha contém N números inteiros: "`-1`" se o quadrado não está colorido, e "`0`" se está colorido com o tom 0.

Saída

Seu programa deve escrever na saída a fita totalmente colorida, de acordo com a regra definida acima.

Restrições

- $3 \leq N \leq 10000$;
- Sempre existe pelo menos um "`0`" inicialmente na fita.

Informações sobre a pontuação

- Em um conjunto de casos de teste somando 80 pontos, $N \leq 1000$

Exemplos

Entrada	**Saída**
`8` `-1 -1 0 -1 -1 -1 0 -1`	`2 1 0 1 2 1 0 1`

Entrada	**Saída**
`13` `-1 0 -1 -1 -1 -1 -1 -1 -1 -1 0 -1 -1`	`1 0 1 2 3 4 4 3 2 1 0 1 2`

Entrada	**Saída**
`6` `0 -1 -1 -1 -1 -1`	`0 1 2 3 4 5`

Arquivos

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

Aldo tem N arquivos em seu computador, cada um com um tamanho em bytes. Ele quer dividir estes arquivos em pastas, porém o sistema do computador é velho e só aceita pastas com as duas seguintes limitações:

- Uma pasta pode ter no máximo **dois** arquivos
- A soma dos tamanhos dos arquivos na pasta não pode exceder B bytes

Como ele tem muitos arquivos ele prefere não criar muitas pastas. Dado o tamanho dos arquivos, calcule o número mínimo possível de pastas.

Vamos supor um exemplo que temos os arquivos de tamanho 1, 2 e 3, e que o limite de bytes seja 3. A solução é colocar os dois primeiros arquivos juntos, totalizando apenas 2 pastas.

Entrada

A entrada consiste de duas linhas. A primeira linha contém os números inteiros N e B. A segunda linha contém N inteiros indicando o tamanho de cada arquivo.

Saída

Seu programa deve escrever uma única linha na saída, contendo um único número inteiro, a quantidade mínima possível de pastas.

Restrições

- $1 \leq N \leq 10^5$
- $1 \leq B \leq 10^9$
- Os arquivos terão tamanho entre 1 e B, inclusive

Informações sobre a pontuação

- Em um conjunto de casos de teste somando 20 pontos, $N \leq 10$
- Em um conjunto de casos de teste somando 50 pontos, $N \leq 1000$

Exemplos

Entrada	**Saída**
`3 3` `1 2 3`	`2`

Entrada	**Saída**
`5 4` `4 3 1 2 2`	`3`