OBI2013

Caderno de Tarefas

Modalidade **Programação** • Nível **1**, Fase **2**

31 de agosto de 2013

A PROVA TEM DURAÇÃO DE **4** HORAS

Promoção:

Sociedade Brasileira de Computação

Patrocínio:

Fundação Carlos Chagas

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.
- 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; e soluções na linguagem Python devem ser arquivos com sufixo *.py*. 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 disquete, 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,print,write*
- 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.

Distância de Manhattan

Nome do arquivo fonte: `manhattan.c`, `manhattan.cpp`, `manhattan.pas`, `manhattan.java`, *ou* `manhattan.py`

Maria é uma moradora de Nlogópolis, uma cidade na Nlogônia que tem uma característica muito interessante: todas as ruas da cidade ou são orientadas no sentido norte-sul ou são orientadas no sentido leste-oeste. Isso significa que, dadas duas ruas, ou elas são paralelas ou elas são perpendiculares entre si.

Todas as ruas da cidade são de mão dupla e é possível seguir em qualquer direção em um cruzamento.

Agora Maria está atrasada para uma reunião e precisa de sua ajuda. Dadas as coordenadas iniciais de Maria e da reunião, determine o número mínimo de cruzamentos que Maria deve atravessar para chegar ao seu destino. Esse número inclui o cruzamento onde ocorrerá a reunião mas não inclui a posição inicial de Maria.

Entrada

A única linha da entrada contém quatro inteiros, X_m, Y_m, X_r, Y_r, indicando as coordenadas de Maria (X_m, Y_m) e da reunião (X_r, Y_r). O ponto de partida de Maria nunca será igual ao local da reunião, ou seja, pelo menos uma das coordenadas será diferente.

Saída

Seu programa deve imprimir uma única linha contendo um único inteiro: o número mínimo de cruzamentos que Maria precisa atravessar para chegar até o local da reunião.

Restrições

- $0 \leq X_m, Y_m \leq 1000$
- $0 \leq X_r, Y_r \leq 1000$

Exemplos

Entrada	**Saída**
`0 0 5 6`	`11`

Entrada	**Saída**
`52 75 120 75`	`68`

Polígono

Nome do arquivo fonte: `poligono.c`, `poligono.cpp`, `poligono.pas`, `poligono.java`, *ou* `poligono.py`

Renato gosta muito de geometria e acaba de achar alguns palitos em seu quarto. Ele está tentando utilizar esses palitos de forma a fazer um polígono com o maior número de lados possíveis.

Para montar o polígono, Renato não quer cruzar os palitos; ou seja, os palitos devem se tocar apenas nas pontas. Ele também não quer quebrar nenhum palito, de forma que todos os palitos que forem usados devem manter sua medida original.

Se, por exemplo, os palitos têm medidas 3, 4 e 5, é possível utilizar todos os três palitos para formar um triângulo. Mas se as medidas são 1, 1, 1 e 5, é possível formar um triângulo com três lados iguais a 1 mas não é possivel formar um polígono com todos os 4 palitos.

Você consegue ajudar Renato a descobrir qual é o maior número de palitos que ele consegue usar?

Entrada

A primeira linha contém apenas um inteiro N que indica o número de palitos. A segunda linha possui N inteiros indicando as medidas dos palitos.

Saída

Se programa deve imprimir uma única linha, contendo um único inteiro, o maior número de lados que o polígono pode ter seguindo as restrições do enunciado. Se não for possível formar nenhum polígono usando os palitos, imprima 0.

Restrições

- $3 \leq N \leq 100000$
- As medidas dos palitos são inteiros positivos menores ou iguais a 10000

Informações sobre a pontuação

- Para um conjunto de entradas totalizando 50 pontos, $N \leq 1000$

Exemplos

Entrada	**Saída**
3 3 4 5	3

Entrada	**Saída**
3 3 1 2	0

Entrada	**Saída**
4 1 1 5 1	3

Quadradinho de 8

Nome do arquivo fonte: `quadrado.c`, `quadrado.cpp`, `quadrado.pas`, `quadrado.java`, *ou* `quadrado.py`

Fernando ficou sabendo de um novo jogo chamado quadradinho de 8. Nesse jogo, é apresentado ao jogador uma fileira de quadrados, um do lado do outro. Em cada quadrado há um número escrito. Veja abaixo um exemplo de fileira de quadrados:

3	4	6	0	2	9

Para ganhar, o jogador deve escolher alguns quadrados de forma que eles juntos formem apenas um retângulo contíguo e que a soma de seus números seja divisível por 8. Na fileira de quadrados acima, o jogador ganha se escolher os quadrados com os números 6, 0 e 2. O jogador perde se escolher os quadrados com 3, 4 e 9, apesar da soma ser divisivel por 8, os quadrados não estão juntos, eles acabam formando dois retângulos separados.

Você deve estar pensando agora que Fernando quer sua ajuda para que você mostre a ele como ganhar o jogo, mas Fernando é um garoto muito esperto e sabe resolver o jogo rapidamente. Ele quer na verdade que você o ajude a descobrir de quantas formas é possível ganhar esse jogo.

Entrada

A entrada possui duas linhas. A primeira linha contém apenas um inteiro N que indica o número de quadrados na fileira de um jogo. A segunda linha contém N inteiros indicando na ordem os números presentes nos quadrados da fileira de um jogo.

Saída

Seu programa deve imprimir uma única linha, contendo apenas um inteiro, o número de maneiras de ganhar o jogo apresentado na entrada. Se não for possível que o jogador ganhe o jogo, imprima 0.

Restrições

- $1 \leq N \leq 1000000$
- Os números nos quadrados são inteiros não negativos menores ou iguais a 1000.

Informações sobre a pontuação

- Para um conjunto de entradas totalizando 50 pontos, $N \leq 200$
- Para um conjunto de entradas totalizando 70 pontos, $N \leq 5000$

Exemplos

Entrada	**Saída**
6 3 4 6 0 2 9	3

Entrada	**Saída**
7 1 1 1 1 1 1 1	0

Entrada	**Saída**
5 8 0 8 0 8	15

Troco

Nome do arquivo fonte: `troco.c`, `troco.cpp`, `troco.pas`, `troco.java`, *ou* `troco.py`

Você está num supermercado e está na fila do caixa para comprar alguns produtos. Assim que você termina de passar as compras pelo caixa, se lembra que tem várias moedas em seu bolso, algumas repetidas, e fica pensando se com elas dá para pagar exatamente o valor das compras (para assim se livrar destas moedas e ficar com os bolsos mais leves). Você consegue pagar o valor exato da conta usando estas moedas?

Entrada

A primeira linha da entrada contém dois números inteiros V e M, indicando, respectivamente, o valor final da compra e o número de moedas que você tem em seu bolso. A segunda linha contém M números inteiros que descrevem o valor M_i de cada moeda existente em seu bolso.

Saída

Seu programa deve imprimir apenas uma linha, contendo apenas um caractere: `S` caso seja possível pagar o valor exato da conta usando apenas suas moedas, ou `N` caso contrário.

Restrições

- $1 \leq V \leq 10^5$
- $1 \leq M \leq 10^3$
- $1 \leq M_i \leq 10^5$

Informações sobre a pontuação

- Para um conjunto de entradas que vale 70 pontos, $M \leq 20$

Exemplos

Entrada	**Saída**
`16 4` `25 10 5 1`	`S`

Entrada	**Saída**
`20 4` `25 10 5 1`	`N`