OBI2013

Caderno de Tarefas

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

9 de março de 2013

A PROVA TEM DURAÇÃO DE **5** 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 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.
- 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.

Lençol

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

João dispõe de dois pedaços retangulares de tecido, e quer usá-los para fazer um lençol, também retangular, de dimensões $A \times B$. Se necessário, os dois pedaços retangulares podem ser unidos por uma costura, mas João quer que a costura seja paralela aos lados dos retângulos. Os cortes, se necessários, também devem ser paralelos aos lados dos retângulos.

Dadas as dimensões dos pedaços de tecido e do lençol, escreva um programa que determina se é possível João fazer o lençol com as dimensões desejadas.

Entrada

A entrada contém uma única linha, com seis inteiros A_1, B_1, A_2, B_2, A e B, representando, respectivamente, as dimensões dos dois retângulos disponíveis, e as dimensões do retângulo desejado.

Saída

Seu programa deve imprimir uma única linha contendo um caractere `S` se é possível fazer o lençol, e `N` caso contrário.

Restrições

- $1 \leq A_1, B_1, A_2, B_2, A, B \leq 10^6$

Exemplos

Entrada	**Saída**
`4 2 3 5 4 4`	`S`

Entrada	**Saída**
`4 2 2 5 4 5`	`N`

Entrada	**Saída**
`1 2 3 5 5 2`	`S`

Entrada	**Saída**
`3 4 10 9 9 10`	`S`

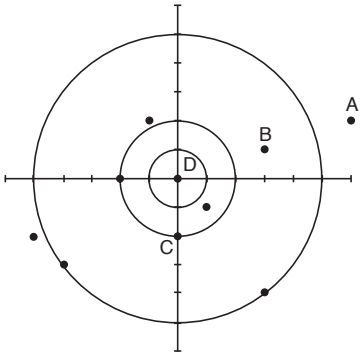
Tiro ao Alvo

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

Recentemente Juquinha ganhou de aniversário um joguinho bem clássico: Tiro ao Alvo. Ele arrumou um ótimo lugar em seu quarto para se divertir com o jogo, porém após ler todas as regras do jogo ele percebeu que precisa da sua ajuda para calcular a pontuação obtida.

Segundo as regras, o alvo do jogo é composto por C círculos, todos centrados na origem (0,0). Juquinha atira T vezes e após cada tiro informa suas coordenadas. A pontuação de cada tiro é feita da seguinte forma: para cada círculo em que o tiro estiver contido Juquinha recebe um ponto.

Considere por exemplo a figura abaixo. O tiro marcado com a letra A recebe zero pontos, pois não está contido por nenhum círculo. O tiro marcado com a letra B recebe um ponto, pois está contido por um círculo (o mais externo). O tiro marcado com a letra C recebe dois pontos, pois está contido por dois círculos (note que este caso mostra que tiros exatamente na borda de um círculo são considerados como contidos pelo círculo). Já o tiro marcado com a letra D recebe três pontos, pois está contido pelos três círculos. Considerando todos os pontos, a pontuação total de Juquinha é de 13 pontos.



Dados os raios de C círculos centrados na origem e as coordenadas dos T tiros realizados por Juquinha, escreva um programa que calcula o total de pontos que Juquinha obteve.

Entrada

A primeira linha da entrada contém dois inteiros positivos, C e T, que representam, respectivamente, o número de círculos do alvo e o número de tiros.

Cada uma das C linhas seguintes contém um inteiro positivo. O i-ésimo inteiro R_i representa o raio do i-ésimo círculo. Os raios R_i são fornecidos em ordem crescente.

Cada uma das T linhas seguintes contém um par X, Y de inteiros, que representam as coordenadas de cada tiro.

Saída

Seu programa deve imprimir uma única linha, contendo apenas um inteiro, o total de pontos obtidos por Juquinha.

Restrições

- $1 \leq C \leq 10^5$
- $1 \leq R_i \leq 10^6$ para $1 \leq i \leq C$
- $R_i > R_{i-1}$ para $2 \leq i \leq C$
- $1 \leq T \leq 10^5$
- $-10^5 \leq X, Y \leq 10^5$

Informações sobre a pontuação

Em um conjunto de casos de teste que totaliza 30 pontos:

- $1 \leq C \leq 10^3$
- $1 \leq R_i \leq 10^4$ para $1 \leq i \leq N$
- $1 \leq T \leq 10^3$
- $-10^2 \leq X, Y \leq 10^2$

Exemplos

Entrada	**Saída**
3 10 1 2 5 0 0 -2 0 0 -2 3 -4 -4 -3 3 1 6 2 -1 2 -5 -2 1 -1	13

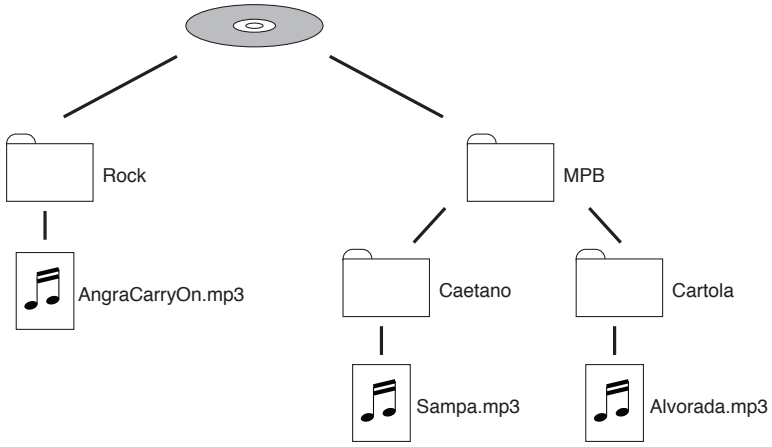
Entrada	**Saída**
3 6 1 2 5 1 0 0 3 -5 0 0 0 -3 -3 1 1	11

Catálogo de Músicas

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

Joyce é uma menina que gosta muito de ouvir música, e possui uma enorme coleção de músicas num DVD. Ela é uma menina organizada e deixa suas músicas em pastas, mas como o número de músicas e de pastas é grandre, Joyce construiu um catálogo para melhor localizá-las.

Para o catálogo Joyce utilizou uma convenção usual em sistemas operacionais, em que a descrição da localização de cada arquivo é formada pela sequência dos nomes das pastas no caminho da raiz do DVD até o arquivo, separados pelo caractere barra ('/'). Por exemplo, na figura abaixo, a descrição da música `Sampa.mp3` no catálogo é `MPB/Caetano/Sampa.mp3`.



Utilizando essa convenção, o catálogo do DVD mostrado na figura é:

```
Rock/AngraCarryOn.mp3
MPB/Caetano/Sampa.mp3
MPB/Cartola/Alvorada.mp3
```

Como o DVD de Joyce tem muitas músicas e pastas, o catálogo é muito grande. Joyce notou no entanto que o catálogo poderia ser menor (ter um número menor de caracteres) caso ela utilizasse outro conceito usual na nomeação de arquivos em sistemas operacionais: usar uma pasta como referência, ao invés da raiz.

Se uma pasta diferente da raiz for escolhida como referência, então para todos os arquivos que estejam diretamente nessa pasta ou em alguma subpasta não será mais necessário escrever o nome da pasta referência no catálogo. Para as demais pastas, é necessário indicar o caminho utilizando as pastas acima (na direção da raiz) utilizando a convenção '`../`' para a pasta imediatamente acima da pasta referência. No exemplo da figura acima, no caso de a referência ser a pasta `Caetano`, a música `Sampa.mp3` seria simplesmente descrita como `Sampa.mp3`. Já a música `Alvorada.mp3` seria descrita como `../Cartola/Alvorada.mp3`.

Assim, se a pasta `Caetano` for utilizada como referência, o catálogo será:

```
../../Rock/AngraCarryOn.mp3
Sampa.mp3
../Cartola/Alvorada.mp3
```

Nesse caso, a descrição do catálogo tem 59 carateres, menor do que quando a referência utilizada é a raiz do DVD.

Seu objetivo é, dada a informação de todas as músicas do catálogo, determinar o número mínimo de caracteres necessários para descrever o catálogo.

Entrada

A primeira linha da entrada contém um inteiro N, indicando quantos arquivos Joyce possui no DVD. Cada uma das N linhas seguintes contém a descrição de um arquivo, a partir da raiz.

Saída

Seu programa deve imprimir uma única linha, contendo apenas um inteiro, o número mínimo de caracteres necessários para descrever o catálogo.

Restrições

- $1 \leq N \leq 10^5$
- Número de pastas na entrada $\leq 10^5$
- O nome de cada pasta e de cada arquivo é composto por no máximo 20 caracteres, entre letras minúsculas, maiúsculas e ponto (`.`)
- Cada pasta possui no máximo 100 pastas como filhas diretas.

Informações sobre a pontuação

- Em um conjunto de casos de teste que totaliza 30 pontos, $N \leq 1000$ e o número de pastas ≤ 1000.

Exemplos

Entrada	**Saída**

`3`
`Rock/AngraCarryOn.mp3`
`MPB/Caetano/Sampa.mp3`
`MPB/Cartola/Alvorada.mp3`
	`59`

Entrada	**Saída**

`2`
`Preferidas/chacoalha/uia.mp3`
`Preferidas/chacoalha/eia.mp3`
	`14`

Entrada	**Saída**

`6`
`delta/india/juliet/lima`
`bravo/echo`
`bravo/foxtrot`
`charlie/hotel`
`delta/india/kilo`
`bravo/golf`
	`76`

Vende-se

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

A Otacílio Busílis Imóveis (OBI) é a maior imobiliária de Nlogópolis, especializada no aluguel de prédios comerciais; todas as suas propriedades se localizam na Avenida Doutor Otacílio Busílis, assim chamada em homenagem ao fundador da OBI.

Devido à crise econômica mundial, a OBI precisa vender K de seus imóveis para levantar capital de giro. Dr. Otacílio quer que os prédios restantes após a venda sejam o mais próximos possível — ou seja, a distância entre o primeiro e o último prédios restantes deve ser a menor possível.

Infelizmente, a OBI é proprietária de tantos prédios que o Dr. Otacílio não sabe quais prédios ele deve vender; ele lhe contratou para que você escreva um programa que determina qual é a mínima distância possível entre o primeiro e o último prédios da OBI na avenida, após a venda de K prédios.

Entrada

A primeira linha da entrada contém os inteiros N e K, indicando, respectivamente, quantos prédios a OBI possui, e quantos prédios ela pretende vender. A linha seguinte contém N inteiros X_i, indicando a distância de cada um dos N prédios ao início da avenida, em metros.

Saída

A saída deve conter um único inteiro indicando a menor distância possível entre o primeiro e o último prédio possuídos pela OBI após a venda.

Restrições

- $3 \leq N \leq 10^5$
- $N - K \geq 2$
- $1 \leq X_i \leq 10^6$
- todos os X_i são distintos

Informações sobre a pontuação

- Em um conjunto de casos de teste que totaliza 30 pontos, $N \leq 10$
- Em um conjunto de casos de teste que totaliza 70 pontos, $N \leq 2000$

Exemplos

Entrada	**Saída**
5 2 10 7 4 8 2	3

Entrada	**Saída**
8 6 16 11 1 7 29 4 22 2	1