Competidor(a): _____

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

OBI2019

Caderno de Tarefas

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

14 de agosto de 2019

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 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; 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 tarefas 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 2 devem ser arquivos com sufixo *_py2.py*; soluções na linguagem Python 3 devem ser arquivos com sufixo *_py3.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.

Matriz super-legal

Nome do arquivo: `matriz.c`, `matriz.cpp`, `matriz.pas`, `matriz.java`, `matriz.js`, `matriz_py2.py` *ou* `matriz_py3.py`

Denotando por $A_{i,j}$ o elemento na i-ésima linha e j-ésima coluna da matriz A, dizemos que uma matriz é "legal" se a condição

$$A_{1,1} + A_{lin,col} \leq A_{1,col} + A_{lin,1}$$

é verdadeira para todo $lin > 1$ e $col > 1$.

Adicionalmente, dizemos que a matriz é "super-legal" se cada uma de suas submatrizes com pelo menos duas linhas e duas colunas é legal. Lembre que uma submatriz S de uma matriz $M_{L \times C}$ é uma matriz que inclui todos os elementos $M_{i,j}$ tais que $l_1 \leq i \leq l_2$ e $c_1 \leq j \leq c_2$, para $1 \leq l_1 \leq l_2 \leq L$ e $1 \leq c_1 \leq c_2 \leq C$.

A sua tarefa é, dada uma matriz A, determinar a maior quantidade de elementos de uma submatriz super-legal da matriz A.

Entrada

A primeira linha contém dois inteiros L e C indicando respectivamente o número de linhas e o número de colunas da matriz. Cada uma das L linhas seguintes contém C inteiros X_i representando os elementos da matriz.

Saída

Seu programa deve produzir uma única linha, com apenas um número inteiro, a maior quantidade de elementos de uma submatriz super-legal da matriz da entrada, ou zero no caso de não existir uma submatriz super-legal.

Restrições

- $2 \leq L, C \leq 1000$
- $-10^6 \leq X_i \leq 10^6$

Informações sobre a pontuação

- Para um conjunto de casos de testes valendo 10 pontos, $L, C \leq 3$.
- Para um conjunto de casos de testes valendo outros 50 pontos, $L, C \leq 300$.

Exemplo de entrada 1	**Exemplo de saída 1**

`3 3`
`1 4 10`
`5 2 6`
`11 1 3`
	`9`

Exemplo de entrada 2	**Exemplo de saída 2**

`3 3`
`1 3 1`
`2 1 2`
`1 1 1`
	`4`

Exemplo de entrada 3	**Exemplo de saída 3**
5 6 1 1 4 0 3 3 4 4 9 7 11 13 -3 -1 4 2 8 11 1 5 9 5 9 10 4 8 10 5 8 8	15

Nota esquecida

Nome do arquivo: `nota.c`, `nota.cpp`, `nota.pas`, `nota.java`, `nota.js`, `nota_py2.py` *ou* `nota_py3.py`

João aprendeu na escola que a média de dois números é o valor da soma desses dois números dividido por dois. Ou seja, a média de dois números A e B é $M = \frac{A+B}{2}$.

A professora contou para João as notas que ele tirou nas duas provas de Geografia. As duas notas são números inteiros entre 0 e 100. João prontamente calculou a média das duas provas, que também resultou em um número inteiro.

Mas João é muito esquecido, e agora não consegue lembrar-se das duas notas que tirou na prova. Ele consegue se lembrar de apenas uma das notas das provas. Por sorte, ele consegue se lembrar também da média entre as duas notas.

Você pode ajudar João a determinar a nota da outra prova?

Entrada

A primeira linha contém um número inteiro A, indicando a nota de uma prova. A segunda linha contém um número inteiro M, indicando a média entre as duas notas das provas.

Saída

Seu programa deve produzir uma única linha, com apenas um número inteiro, a nota da outra prova, que João não consegue recordar.

Restrições

- $0 \leq A \leq 100$
- $0 \leq M \leq 100$

Informações sobre a pontuação

- Para um conjunto de casos de testes valendo 10 pontos, as duas notas das provas são iguais.

Exemplo de entrada 1	**Exemplo de saída 1**
100 70	40

Exemplo de entrada 2	**Exemplo de saída 2**
80 75	70

Exemplo de entrada 3	**Exemplo de saída 3**
1 50	99

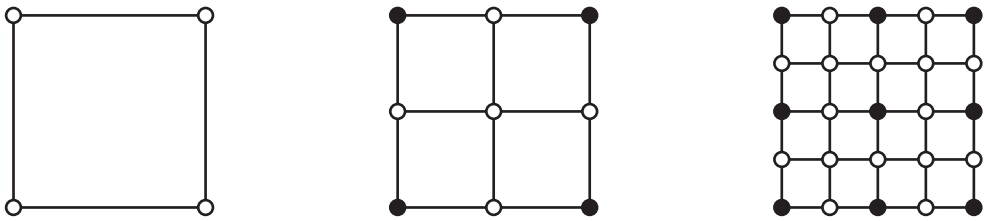
Ponto do meio

Nome do arquivo: `ponto.c`, `ponto.cpp`, `ponto.pas`, `ponto.java`, `ponto.js`, `ponto_py2.py` *ou* `ponto_py3.py`

Paulo foi contratado por uma companhia de mapas digitais para implementar melhorias em seus mapas. Seu primeiro trabalho na empresa é implementar o algoritmo denominado *deslocamento do ponto do meio*. Não vamos descrever aqui o algoritmo completo, vamos focar apenas num aspecto importante para Paulo, que está preocupado em otimizar o uso de memória em sua implementação do algoritmo. O algoritmo funciona em passos. Inicialmente, quatro pontos do mapa são selecionados, formando um quadrado. Então a cada passo, para cada quadrado, faça:

- adicione quatro novos pontos, um ponto em cada lado do quadrado, exatamente no meio do lado.
- adicione também mais um novo ponto, exatamente no meio do quadrado.

O algoritmo utiliza os pontos criados para calcular e armazenar valores do mapa, mas Paulo está interessado apenas no número de pontos criados pelo algoritmo. Na figura abaixo, pontos brancos representam pontos adicionados no passo corrente, pontos pretos representam pontos adicionados em passos anteriores.



Paulo notou que o algoritmo gera muitos pontos, e muitos pontos pertencem a mais de um quadrado. Para economizar memória, Paulo planeja calcular e armazenar cada ponto apenas uma vez.

Sua tarefa, dado o número de passos que Paulo planeja executar, é determinar a quantidade de pontos únicos que Paulo necessita calcular e armazenar.

Entrada

A entrada consiste de uma única linha que contém um inteiro N, o número de passos.

Saída

Seu programa deve produzir uma única linha, com apenas um número inteiro, a quantidade de pontos únicos que Paulo deve calcular e armazenar em N passos.

Restrições

- $1 \leq N \leq 50$

Informações sobre a pontuação

- Para um conjunto de casos de testes valendo 10 pontos, $N \leq 3$.
- Para um conjunto de casos de testes valendo outros 40 pontos, $4 \leq N \leq 10$.

Exemplo de entrada 1	**Exemplo de saída 1**
2	25

Exemplo de entrada 2	**Exemplo de saída 2**
5	1089

Exemplo de entrada 3	**Exemplo de saída 3**
1	9