OBI2016

Caderno de Tarefas

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

3 de junho de 2016

A PROVA TEM DURAÇÃO DE **3** HORAS

Promoção:

Sociedade Brasileira de Computação

Apoio:

CNPq
Conselho Nacional de Desenvolvimento Científico e Tecnológico

v1.0

Instruções

LEIA ATENTAMENTE ESTAS INSTRUÇÕES ANTES DE INICIAR A PROVA

- Este caderno de tarefas é composto por 4 páginas (não contando a folha de rosto), numeradas de 1 a 4. 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 Python2 devem ser arquivos com sufixo *.py2*; soluções na linguagem Python3 devem ser arquivos com sufixo *.py3*; 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 Python2 ou Python3: *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.

Plantação de morango

Nome do arquivo: `morango.c`, `morango.cpp`, `morango.pas`, `morango.java`, `morango.js`, `morango.py2` *ou* `morango.py3`

Os administradores da Fazenda Fartura planejam criar uma nova plantação de morangos, no formato retangular. Eles têm vários locais possíveis para a nova plantação, com diferentes dimensões de comprimento e largura. Para os administradores, o melhor local é aquele que tem a maior área. Eles gostariam de ter um programa de computador que, dadas as dimensões de dois locais, determina o que tem maior área. Você pode ajudá-los?

Entrada

A entrada contém quatro linhas, cada uma contendo um número inteiro. As duas primeiras linhas indicam as dimensões (comprimento e largura) de um dos possíveis locais. As duas últimas linhas indicam as dimensões (comprimento e largura) de um outro possível local para a plantação de morangos. As dimensões são dadas em metros.

Saída

Seu programa deve escrever uma linha contendo um único inteiro, a área, em metros quadrados, do melhor local para a plantação, entre os dois locais dados na entrada.

Restrições

- $1 \leq$ comprimento ≤ 100
- $1 \leq$ largura ≤ 100

Exemplos

Entrada	**Saída**
30 8 11 56	616

Entrada	**Saída**
12 38 5 20	456

Jogo de par ou ímpar

Nome do arquivo: `jogo.c`, `jogo.cpp`, `jogo.pas`, `jogo.java`, `jogo.js`, `jogo.py2` *ou* `jogo.py3`

Dois amigos, Alice e Bob, estão jogando um jogo muito simples, em que um deles grita ou "par" ou "ímpar" e o outro imediatamente responde ao contrário, respectivamente "ímpar" ou "par". Em seguida, ambos exibem ao mesmo tempo uma mão cada um, em que alguns dedos estão estendidos e outros dobrados. Então eles contam o número total de dedos estendidos. Se a soma for par, quem gritou "par" ganha. Se a soma for ímpar, quem gritou "ímpar" ganha.

Por exemplo, suponhamos que a Alice gritou "par" e o Bob respondeu "ímpar". Em seguida, Alice não deixou nenhum dos seus dedos estendidos, ao passo que Bob deixou três dedos estendidos. A soma então é três, que é ímpar, portanto Bob ganhou.

Seu programa deve determinar quem ganhou, tendo a informação de quem gritou par e o número de dedos estendidos de cada um.

Entrada

A entrada contém três linhas, cada uma com um número inteiro, P, D_1 e D_2, nesta ordem. Se $P = 0$ então Alice gritou "par", ao passo que se $P = 1$ então Bob gritou "par". Os números D_1 e D_2 indicam, respectivamente, o número de dedos estendidos da Alice e do Bob.

Saída

Seu programa deverá imprimir uma única linha, contendo um único número inteiro, que deve ser 0 se Alice foi a ganhadora, ou 1 se Bob foi o ganhador.

Restrições

- $P = 0$ ou $P = 1$
- $0 \leq D_1 \leq 5$
- $0 \leq D_2 \leq 5$

Exemplos

Entrada	**Saída**
0 0 3	1

Entrada	**Saída**
1 0 3	0

Entrada	**Saída**
0 1 5	0

Lâmpadas

Nome do arquivo: `lampadas.c`, `lampadas.cpp`, `lampadas.pas`, `lampadas.java`, `lampadas.js`, `lampadas.py2` *ou* `lampadas.py3`

Você está de volta em seu hotel na Tailândia depois de um dia de mergulhos. O seu quarto tem duas lâmpadas. Vamos chamá-las de A e B. No hotel há dois interruptores, que chamaremos de I_1 e I_2. Ao apertar I_1, a lâmpada A troca de estado, ou seja, acende se estiver apagada e apaga se estiver acesa. Se apertar I_2, ambas as lâmpadas A e B trocam de estado.

As lâmpadas inicialmente estão ambas apagadas. Seu amigo resolveu bolar um desafio para você. Ele irá apertar os interruptores em uma certa sequência, e gostaria que você respondesse o estado final das lâmpadas A e B.

Entrada

A primeira linha contém um número N que representa quantas vezes seu amigo irá apertar algum interruptor. Na linha seguinte seguirão N números, que pode ser 1, se o interruptor I_1 foi apertado, ou 2, se o interruptor I_2 foi apertado.

Saída

Seu programa deve imprimir dois valores, em linhas separadas.

Na primeira linha, imprima 1 se a lâmpada A estiver acesa no final das operações e 0 caso contrário. Na segunda linha, imprima 1 se a lâmpada B estiver acesa no final das operações e 0 caso contrário.

Restrições

- $1 \leq N \leq 10^5$

Informações sobre a pontuação

- Em um conjunto de casos de teste equivalente a 20 pontos, $N = 3$.

Exemplos

Entrada	**Saída**
3 1 2 2	1 0

Entrada	**Saída**
4 2 1 2 2	0 1