Competidor(a): _____

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

OBI2019

Caderno de Tarefas

Modalidade **Programação** • **Nível Júnior** • Fase **Local**

30 de maio de 2019

A PROVA TEM DURAÇÃO DE **2** HORAS

Promoção:

Sociedade Brasileira de Computação

Apoio:

CNPq
Conselho Nacional de Desenvolvimento Científico e Tecnológico

Instruções

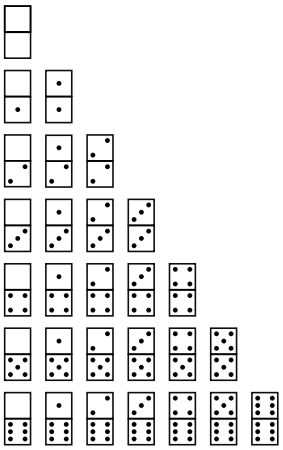
LEIA ATENTAMENTE ESTAS INSTRUÇÕES ANTES DE INICIAR A PROVA

- 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; 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 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 2 devem ser arquivos com sufixo *.py2*; soluções na linguagem Python 3 devem ser arquivos com sufixo *.py3*; 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.

Dominó

Nome do arquivo: "`domino.x`", *onde* `x` *deve ser* `c`, `cpp`, `pas`, `java`, `js`, `py2` *ou* `py3`

O jogo de dominó tradicional, conhecido como duplo-6, possui 28 peças. Cada peça está dividida em dois quadrados e dentro de cada quadrado há entre 0 e 6 círculos. O jogo é chamado de duplo-6 justamente porque esse é o maior número de círculos que aparece num quadrado de uma peça. A figura ao lado mostra uma forma de organizar as 28 peças do jogo duplo-6 em 7 linhas. Essa figura permite ver claramente quantas peças haveria num jogo de dominó, por exemplo, do tipo duplo-4: seriam todas as peças das 5 primeiras linhas, 15 peças no total. Também poderíamos ver, seguindo o padrão da figura, quantas peças possui o jogo de dominó conhecido como mexicano, que é o duplo-12. Seriam 91 peças, correspondendo a 13 linhas.



Para a nossa sorte, existe uma fórmula com a qual podemos calcular facilmente o número de peças de um jogo do tipo duplo-N, para um número N natural qualquer: `((N+1)*(N+2))/2`. Neste problema, estamos precisando da sua ajuda para escrever um programa que, dado o valor N, use esta fórmula para calcular e imprimir quantas peças existem num jogo de dominó do tipo duplo-N.

Entrada

A primeira linha da entrada contém um número natural N representando o tipo do jogo de dominó: duplo-N.

Saída

Seu programa deve imprimir uma linha contendo um número natural representando quantas peças existem num jogo de dominó do tipo duplo-N.

Restrições

- $0 \leq N \leq 10000$

Exemplos

Exemplo de entrada 1	**Exemplo de saída 1**
6	28

Exemplo de entrada 2	**Exemplo de saída 2**
12	91

A idade de Dona Mônica

Nome do arquivo: "`idade.x`", *onde* `x` *deve ser* `c`, `cpp`, `pas`, `java`, `js`, `py2` *ou* `py3`

Dona Mônica é mãe de três filhos que têm idades diferentes. Ela notou que, neste ano, a soma das idades dos seus três filhos é igual à idade dela. Neste problema, dada a idade de dona Mônica e as idades de dois dos filhos, seu programa deve computar e imprimir a idade do filho mais velho.

Por exemplo, se sabemos que dona Mônica tem 52 anos e as idades conhecidas de dois dos filhos são 14 e 18 anos, então a idade do outro filho, que não era conhecida, tem que ser 20 anos, pois a soma das três idades tem que ser 52. Portanto, a idade do filho mais velho é 20. Em mais um exemplo, se dona Mônica tem 47 anos e as idades de dois dos filhos são 21 e 9 anos, então o outro filho tem que ter 17 anos e, portanto, a idade do filho mais velho é 21.

Entrada

A primeira linha da entrada contém um inteiro M representando a idade de dona Mônica. A segunda linha da entrada contém um inteiro A representando a idade de um dos filhos. A terceira linha da entrada contém um inteiro B representando a idade de outro filho.

Saída

Seu programa deve imprimir uma linha, contendo um número inteiro, representando a idade do filho mais velho de dona Mônica.

Restrições

- $40 \leq M \leq 110$
- $1 \leq A < M$
- $1 \leq B < M$
- $A \neq B$

Exemplos

Exemplo de entrada 1	**Exemplo de saída 1**
52 14 18	20

Exemplo de entrada 2	**Exemplo de saída 2**
47 21 9	21

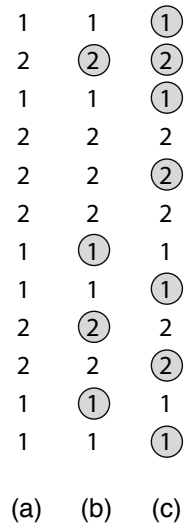
Sequência Secreta

Nome do arquivo: "`secreta.x`", *onde* `x` *deve ser* `c`, `cpp`, `pas`, `java`, `js`, `py2` *ou* `py3`

Na calçada em frente ao Palácio Imperial, não se sabe a razão, existe uma sequência de N números desenhados no chão. A sequência tem a seguinte forma: ela começa e termina com o número 1; apenas os números 1 e 2 aparecem nela; e o número 2 aparece pelo menos uma vez. Veja um exemplo na coluna (a) da figura ao lado.

Ninguém sabe o significado da sequência e, justamente por isso, várias teorias malucas surgiram. Uma delas diz que a sequência representa, na verdade, apenas um valor que estaria relacionado a um segredo dos imperadores. Esse valor é a quantidade *máxima* de números da sequência que poderiam ser marcados com um círculo, de modo que a sequência de números marcados não contenha dois números iguais consecutivos.

A coluna (b) da figura ao lado ilustra uma sequência de 4 números marcados que obedece a restrição acima. Só que é possível marcar 7 números, como mostra a coluna (c) da figura.



Neste problema, dada a sequência original de números desenhados no chão da calçada, seu programa deve computar e imprimir a quantidade máxima de números da sequência que poderiam ser marcados com um círculo sem que haja dois números iguais consecutivos na sequência marcada.

Entrada

A primeira linha da entrada contém um inteiro N representando o tamanho da sequência. As N linhas seguintes contêm, cada uma, um inteiro V_i, para $1 \leq i \leq N$, definindo a sequência de números desenhados no chão da calçada imperial.

Saída

Seu programa deve imprimir uma linha contendo um número inteiro representando a quantidade máxima de números da sequência que poderiam ser marcados com um círculo sem que haja dois números iguais consecutivos na sequência marcada.

Restrições

- $3 \leq N \leq 500$
- V_i é igual a 1 ou 2, para $1 \leq i \leq N$

Exemplos

Exemplo de entrada 1	**Exemplo de saída 1**
5 1 1 1 2 1	3

Exemplo de entrada 2	**Exemplo de saída 2**
12 1 2 1 2 2 2 1 1 2 2 1 1	7

Exemplo de entrada 3	**Exemplo de saída 3**
3 1 2 1	3