INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SÃO PAULO

VI MARATONA DE PROGRAMAÇÃO INTERIF 2023

FASE FINAL

Caderno de Problemas

Informações Gerais

A) Sobre a entrada

1) A entrada de seu programa deve ser lida da entrada padrão.
2) A entrada é composta de um ou mais casos de teste, depende do problema.
3) Quando uma linha da entrada contém vários valores, estes são separados por um único espaço em branco; a entrada não contém nenhum outro espaço em branco.
4) Cada linha, incluindo a última, contém o caractere final-de-linha.
5) O final da entrada pode coincidir com o final do arquivo ou com uma entrada determinada

B) Sobre a saída

1) A saída de seu programa deve ser escrita na saída padrão.
2) Espaços em branco só devem ser colocados quando solicitado.
3) Cada linha, incluindo a última, deve conter o caractere final-de-linha.

C) Regras

1) Só é permitida a comunicação entre os membros de um mesmo grupo.
2) Não é permitida a comunicação com o técnico (coach) do time.
3) Eventuais dúvidas sobre a prova utilizar o menu “clarification” do sistema de submissão.

D) Ambiente computacional

O sistema de correção das submissões será executado utilizando a distribuição Ubuntu GNU/Linux 20.04.2 LTS amd64, tendo os seguintes compiladores/interpretadores configurados:

- C - gcc version 9.3.0 (Ubuntu 9.3.0-17ubuntu1~20.04)
- C++ - gcc version 9.3.0 (Ubuntu 9.3.0-17ubuntu1~20.04)
- Python 3 - Python 3.8.10
- Java - openjdk-11.0.11
- C# - mono JIT 6.12

Problema A
A arte de Bryan

Por Jones Mendonça de Souza (IFSP – Campus Barretos)

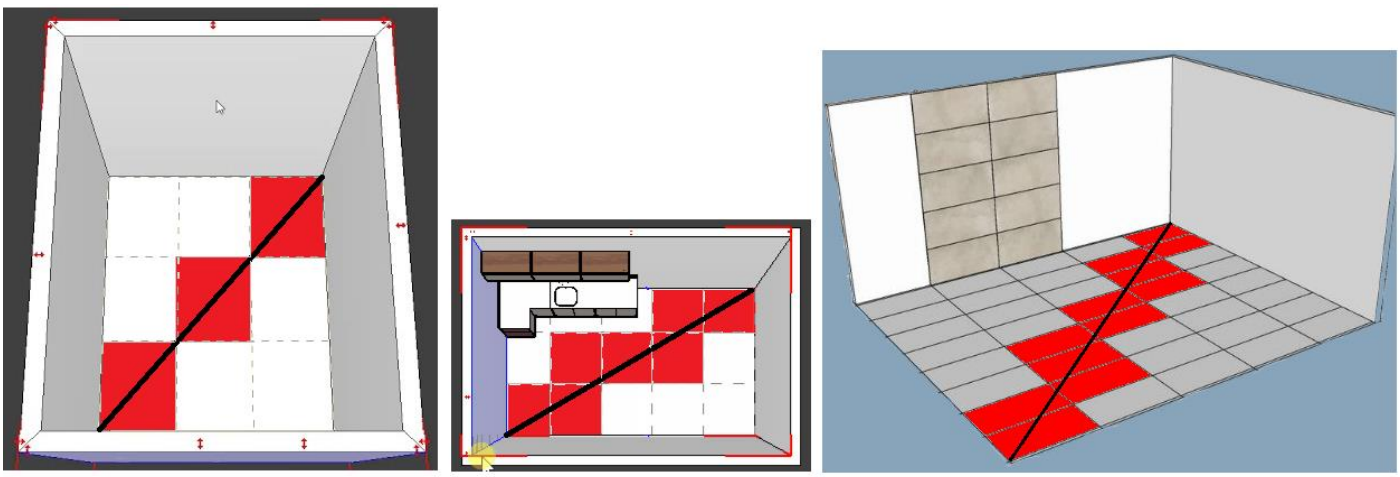
Arquivo: arte.[c|cpp|java\cs\py]

***Timelimit:** 1*

Taismara levou seu filho Bryan para conhecer as obras que estão em andamento na cidade de Barretos. No momento em que mãe conversava com os mestres de obras, ele resolveu demonstrar sua paixão pela geometria. Então, abriu a mochila e pegou seu estilete que guardava em seu estojo e, com muito cuidado e precisão, riscou os pisos dos cômodos diagonalmente. Diversas formas geométricas surgiram em diferentes partes da casa, todos cuidadosamente traçados e simetricamente dispostos.

Ao chegar em casa, Bryan contou a sua mãe sobre sua obra de arte, a mãe sem reação começou a entrar em desespero, abriu o whatsapp para apurar a informação e se deparou com diversas imagens enviadas pelos mestres de obras. Para agir de forma rápida sem que os clientes percebessem o ocorrido, Taismara resolveu contratar mão de obra noturna, para que o serviço de restauração fosse realizado antes da visita dos seus clientes, que acontecia pela manhã. Para isso, ela precisava estimar a quantidade de pisos que foram danificados pelo estilete, e fazer a compra do material necessário. No entanto, para se locomover até as obras ela gastaria muitas horas e muito combustível. Logo, teve a seguinte ideia, pegar os projetos de paginação de piso das obras que estava no seu computador e traçar uma diagonal nos desenhos, de modo a simular a façanha do seu filho, e então conseguir contar o número de pisos que foram danificados. Mas, ela viu que essa ideia levaria muito tempo, e como "tempo" não é um problema para você que está competindo na etapa final da VI Maratona de Programação InterIF, ela resolveu pedir sua ajuda.

Para te ajudar, Taismara forneceu algumas ilustrações. Abaixo, na ilustração apresentada em (A) temos um cômodo contendo 3 colunas x 3 linhas de pisos, o corte realizado por Bryan está representado pela linha na diagonal principal, em cor preta. Para este cômodo, serão necessários 3 pisos para a restauração, ilustrados em cor vermelha na imagem. Em (B) temos um cômodo de 5 colunas x 3 linhas de piso, observando a ilustração, serão necessários comprar 7 pisos para restauração. Já para o cômodo ilustrado em (C), contendo 8 colunas x 5 linhas de pisos, serão necessários 12 pisos para restauração.



(A) **(B)** **(C)**

Entrada

A entrada é composta por uma única linha contendo dois inteiros M e N ($1 \leq M, N \leq 1018$) representando as dimensões do cômodo.

Saída

A saída deve apresentar a quantidade de peças de pisos que Taismara deverá comprar.

Exemplos de Entradas	Exemplos de Saídas
3 3	3
5 3	7
8 5	12
4 2	4

Problema B
La maison est tombée!

Por Daniel Corrêa Lobato (IFSP – Campus São José do Rio Preto)

Arquivo: maison.[c|cpp|java\cs\py]

***Timelimit:** 1*

Um grupo de pessoas trabalhou durante três meses cavando um túnel, de aproximadamente 80 metros de comprimento por 70 cm de largura, que ligava uma casa alugada a uma agência de um banco nas proximidades. Os vizinhos da casa suspeitaram de nada, pois para todos os efeitos, a casa tinha sido alugada e estava em reforma: por isso, vários "trabalhadores" chegavam todos os dias, e caminhões de terra saiam do local no final do dia. Quando o grupo finalmente conseguiu chegar até a sala cofre do banco, o alarme tocou: la maison est tombée! Agora, para o grupo, não restava outra coisa a não ser pegar a maior quantidade possível de dinheiro e sair correndo antes da polícia chegar. O grupo possui uma caixa capaz de carregar uma determinada quantidade de dinheiro (medida em kg). O dinheiro está organizado em maços, com massas variáveis, mas que nunca se repetem, e não podem ser divididos (não é possível levar uma parte do maço). O seu papel é conseguir colocar a maior quantidade possível de dinheiro dentro da caixa, respeitando a capacidade máxima dela, e dizer quanto dinheiro eles vão conseguir levar. É possível que a caixa tenha uma capacidade menor do que a massa do menor maço, e nesse caso, o grupo sai com a caixa vazia.

Entrada

Uma linha contendo um número inteiro C ($1 \leq C \leq 10^4$) indicando a capacidade da caixa, em kg; uma linha contendo um número inteiro N ($1 \leq N \leq 300$) indicando a quantidade de maços de cédulas que existem na sala cofre, e; N linhas contendo a massa em kg, P, de cada um dos maços de dinheiro disponíveis ($1 \leq P \leq 10^5$).

Saída

Um valor inteiro indicando a soma máxima de massa que é possível levar na caixa.

Exemplos de Entradas	Exemplos de Saídas
10 3 1 4 8	9
8 5 1 6 4 3 9	8

Problema C
Em Busca da Torre Mais Alta

Por Cássio Agnaldo Onodera (IFSP – Campus Birigui)
Arquivo: torre.[c|cpp|java\cs\py]
***Timelimit:** 1*

Era uma vez, em um reino distante, um talentoso arquiteto chamado Aron. Aron possuía o dom de projetar torres majestosas e queria construir a torre mais alta de todo o reino. No entanto, ele tinha um desafio: a torre deveria seguir uma sequência específica de alturas para criar uma harmonia visual.

A sequência de alturas das seções da torre foi representada por números inteiros. Cada número simbolizava a altura da seção correspondente da torre. Aron sabia que para alcançar a grandiosidade visual desejada, era essencial criar subconjuntos crescentes dentro da sequência de alturas.

Aron procurou a ajuda dos talentosos programadores do reino para determinar o tamanho do maior subconjunto crescente possível. Ele queria garantir que cada seção da torre fosse mais alta que a anterior, criando uma sensação de ascensão constante.

Os programadores, entusiasmados com o desafio, começaram a trabalhar em seus códigos para encontrar a solução. Eles sabiam que, ao resolver esse problema, não apenas ajudariam Aron a construir a torre dos sonhos, mas também aprimorariam suas habilidades de programação.

Conforme os programadores mergulhavam nos códigos, o reino aguardava ansiosamente para ver a magia da programação transformar números em uma obra-prima arquitetônica. A jornada para construir a torre mais alta estava prestes a começar, e todos estavam prontos para testemunhar a grandiosidade surgir a partir de linhas de código cuidadosamente escritas.

Você, como integrante deste seleto grupo de programadores tem um desafio: dado uma sequência de números inteiros que representam as alturas das seções das torres, encontrar o tamanho do maior subconjunto crescente. Este texto foi gerado com a assistência da inteligência artificial da OpenAI (https://chat.openai.com).

Entrada

A primeira linha contém um inteiro N (1 <= N <= 1000), representando o número de elementos na sequência.

A segunda linha contém N inteiros separados por espaço, representando a sequência de números.

Saída

Uma única linha contendo um inteiro, representando o tamanho do maior subconjunto crescente.

Exemplos de Entradas	Exemplos de Saídas
6 5 2 8 6 3 6	3
6 3 4 3 5 2 7	4
10 157 914 662 868 526 233 930 584 794 332	4

Problema D
Biometria das Girafas

Por Jones Mendonça de Souza (IFSP – Campus Barretos)

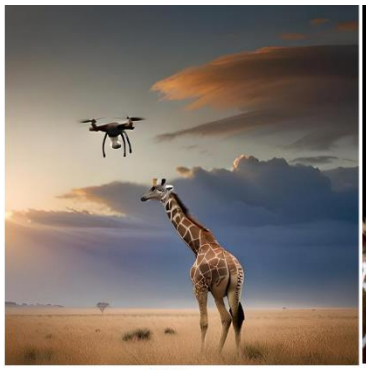
Arquivo: biometria.[c|cpp|java\cs\py]

***Timelimit:** 1*

Um grupo de pesquisadores do IFSP se reuniu para realizar um estudo inédito sobre as girafas. Sabe-se que a diferenciação de sexo entre esses animais é realizada pelo peso, comprimento das pernas e do pescoço. No entanto, um estudo de 2020 na Alemanha revelou que essas características já não são confiáveis devido à falta de nutrientes no sul do Saara, que afeta o desenvolvimento dos machos, tornando suas características semelhantes às fêmeas.

Preocupados com a extinção, os professores Murilo Vargas, Jones Souza e Cassio Onodera, estão desenvolvendo um algoritmo usando os padrões de manchas na pele das girafas para identificar o sexo. A professora Giovana Nakashima, especialista em biologia animal, confirmou que os machos possuem 50% a menos de manchas na região central do pescoço. Para contar as manchas, Cassio criou um drone (Figura A), que obtém coordenadas e captura imagens de alta resolução da área do pescoço (Figura B). Jones desenvolveu algoritmos de processamento de imagens (Figura C, D, E, F), extração de características (Figura G) e representação de dados (Figura H). Atualmente, o projeto está parado há 1 ano com o professor Murilo, que tem em mãos os dados, mas não está conseguindo desenvolver um algoritmo que realize a contagem das manchas dos animais.

Como Murilo estará na final da VI Maratona InterIF, ele resolveu compartilhar seu problema com o grupo de alunos que estarão competindo na Maratona. Será que você consegue ajudar o professor Murilo?

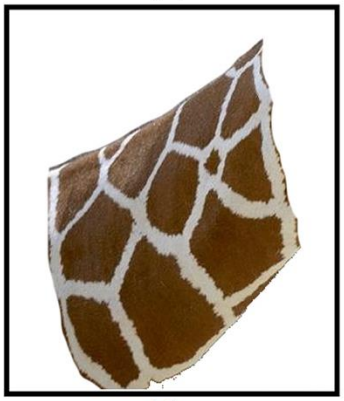


(A)

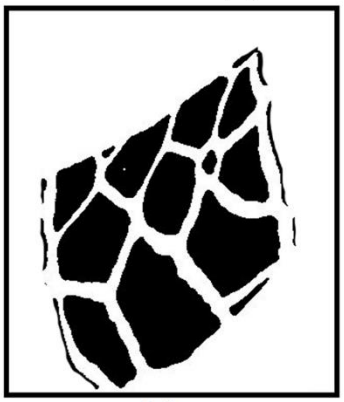
(B)

(C)

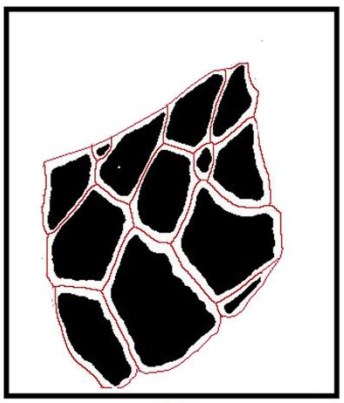
(D)



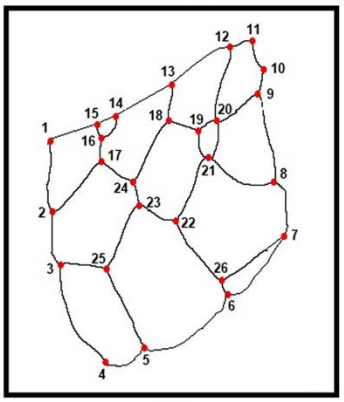
(E)



(F)



(G)



(H)

Entrada

A entrada é composta por um grafo bidirecional, representando o mapeamento das manchas. A primeira linha possui dois inteiros, separados por um espaço, um inteiro **N** ($2 \leq N \leq 10^9$), que indica o número de pontos do grafo e um inteiro **M** ($1 \leq M \leq 10^9$), que indica o número de ligações entre pares de pontos. Depois, separados por um espaço, seguem-se **M** linhas com os inteiros distintos **U (1 ≤ U ≤ N)** e **V (1 ≤ V ≤ N),** separados por um espaço, indicando que existe uma ligação bidirecional entre **U** e **V.** Como pode ocorrer erros na etapa de segmentação das características, não é garantido que todos os pontos estejam interligados, ou seja, pode acontecer de termos vértices isolados em alguns grafos.

Saída

A saída deve apresentar um único inteiro indicando a quantidade de manchas existentes na pele do animal.

Exemplos de Entradas	Exemplos de Saídas
4 5 1 2 2 3 2 4 3 4 3 1	2
26 38 1 2 1 15 2 3 2 17 3 25 3 4 4 5 5 6 5 25 25 23 23 24 23 22 24 18 24 17 17 16 16 15 16 14 15 14 14 13 13 12 13 18 18 19 19 20 19 21 20 12 20 9 20 21 21 22 21 8 8 9 8 7 9 10 10 11 11 12 7 26 7 6 6 26 26 22	13

Problema E
Sequência da Taça de Hogwarts

Por Ester Cattaneo (IFSP – Campus Campinas)

Arquivo: taca.[c|cpp|java\cs\py]

***Timelimit:** 1*

Está chegando o aguardado Prêmio Anual das Taças de Hogwarts, e a competição está acirrada! A Casa Grifinória está empatada com Sonserina na busca pela taça. No entanto, o destino da vitória depende de desvendar uma sequência geométrica mágica escondida.

Enquanto seus melhores amigos Hermione Granger e Harry Potter cumprem um castigo imposto pelo severo Professor Snape, Rony se encontra sozinho nessa missão decisiva. Ele descobriu uma sequência mágica que pode garantir a vitória para a Grifinória, ganhando assim 150 pontos no placar da Casa, e precisa da sua ajuda para decifrá-la antes que a oportunidade se perca.

Instruções:

Na "Sequência das Taças de Hogwarts", sua missão é ajudar Rony a descobrir a razão que determina a sequência de números apresentada. Após identificar a razão, você deve inserir a letra 'X' para indicar o término da entrada dos números da sequência. O programa, então, revelará o próximo número mágico na sequência. Rony está determinado a ajudar Grifinória à vitória na Taça de Hogwarts. Ajude-o a encontrar o próximo número!

Restrições:

- 3 < números na sequência <=10
- -1000 <= número da sequência <= 2000

Entrada

A entrada consistirá em uma única linha contendo uma sequência de números inteiros separados por espaço. A sequência terá pelo menos três números e no máximo dez números. Cada número na sequência estará entre -1000 e 2000. Para ajudar Rony, insira a letra 'X' após os números da sequência para indicar o fim da entrada.

Saída

Seu programa deve imprimir um único inteiro, que é o próximo número mágico na sequência de acordo com a razão da progressão geométrica.

Exemplos de Entradas	Exemplos de Saídas
2 6 18 54 X	162
-3 -6 -12 -24 X	-48
6 18 54 162 X	486
-2 4 -8 16 -32 X	64

Problema F
Turnos, sonecas e monstros!

Por Márcio Kassouf Crocomo (IFSP – Campus Piracicaba)
Arquivo: soneca.[c|cpp|java\cs\py]
***Timelimit:** 1*

No vilarejo de Faz-de-contIF, bravos guerreiros fazem a guarda da pequena cidade em turnos alternados. A cidade sempre possui alguém atento a potenciais perigos, já que monstros famintos estão sempre à espreita e podem atacar a qualquer instante! Entre os monstros, a cidade tem uma perigosa fama de que seus habitantes são petiscos apetitosos!
Embora os Faz-de-contIFanos sejam cidadãos pacíficos e durmam quase o tempo todo, são fortes e sabem se defender quando necessário. Para sua defesa, se organizam em turnos, desta forma: os turnos se alternam entre N guerreiros do vilarejo, onde cada um é responsável por uma mesma quantidade de horas, e estes se alternam de forma cíclica, isto é, após o turno do enésimo guerreiro, é a vez do primeiro que fez a guarda novamente, e todo o processo se repete.

Entrada

A entrada é composta por duas linhas. Na primeira, temos quatro números separados por espaços em branco, na ordem N, H, C, A. Sendo N um número inteiro entre 2 e 10, que representa a quantidade de guerreiros que fazem o turno. H é um número inteiro entre 1 e 12, que representa a duração em horas de cada turno. C é um número inteiro variando entre 1 e 10, representando uma nota para a capacidade da criatura atacante de se mover silenciosamente e, por fim, A é um número inteiro entre 0 e 1000 representando a quantidade de horas a partir do instante 0 em que a criatura realizará o ataque. Na segunda linha, encontram-se N números inteiros variando entre 1 e 10, representando a capacidade auditiva de cada um dos guerreiros que irão se alternar em turnos fazendo a guarda do vilarejo, na ordem em que os turnos serão feitos. Isto é, o primeiro número representa a capacidade auditiva do primeiro guerreiro que ficará de guarda a partir do instante 0, o segundo número a nota do segundo guerreiro, e assim por diante. Considere que, caso a criatura ataque no instante exato em que a troca de turnos está sendo realizada, o guerreiro que estará iniciando seu turno é quem deverá ser considerado como estando de guarda.

Saída

A mensagem “O vilarejo de Faz-de-contIF foi alertado”, caso a criatura tenha atacado em um momento em que a capacidade auditiva do guerreiro que fazia a guarda era igual ou superior à capacidade da criatura de se

mover silenciosamente, ou a mensagem “O vilarejo de Faz-de-contIF foi devorado”, caso a capacidade auditiva do guerreiro que fazia a guarda no momento do ataque da criatura fosse inferior à capacidade da criatura de se mover silenciosamente. Fique atento para a que a mensagem seja exibida exatamente da forma destacada (atenção às letras maiúsculas e minúsculas).

Exemplos de Entradas	Exemplos de Saídas
3 2 7 4 9 8 3	O vilarejo de Faz-de-contIF foi devorado
3 2 7 7 9 8 3	O vilarejo de Faz-de-contIF foi alertado
10 3 5 66 9 8 3 5 7 4 8 2 10 4	O vilarejo de Faz-de-contIF foi devorado

Problema G
Lucky Bet Little Tiger Pix

Por Murilo Varges da Silva (IFSP – Campus Birigui)
Arquivo: loteria.[c|cpp|java\cs\py]
***Timelimit:** 1*

Você está participando de uma competição de programação e o seu desafio é criar um programa que calcule a probabilidade de ganhar em uma nova loteria denominada "**LUCKY BET LITTLE TIGER PIX**". A loteria é realizada semanalmente e consiste em sortear "**C**" números diferentes de 1 a "**N**". Cada bilhete de loteria contém "**C**" números escolhidos pelo participante.

Seu programa deve calcular a probabilidade de ganhar o prêmio principal, que ocorre quando os "**C**" números sorteados são exatamente iguais aos seis números escolhidos pelo participante no bilhete.

Uma forma de permitir que o participante aumente suas probabilidades de ganho é fazer jogos com uma quantidade maior de números, então seu programa deve permitir que o jogador informe o "**C_MIN**" e "**C_MAX**", correspondendo a quantidade mínima e máxima de números que poderão ser escolhidos. Lembrando que a quantidade mínima corresponde ao número de acertos mínimos para ganhar o prêmio da loteria, e que quando a quantidade máxima é maior que a mínima seu programa deve calcular a probabilidade de ganhar a loteria considerando a quantidade de jogos que se pode fazer utilizando aquela quantidade de números.

Entrada

A entrada consiste de uma única linha contendo três inteiros e um valor de ponto flutuante: **N** ($10 \leq$ **N** ≤ 100), **C_MIN** ($2 \leq$ **C_MIN** $\leq$ N), **C_MAX** (C_MIN $\leq$ **C_MAX** $\leq$ N) e **V** ($0.50 \leq$ **V** ≤ 599.99). Sendo **N** o número máximo de números que poderão ser escolhidos (1 até N), **C_MIN** a quantidade mínima de números que devem ser escolhidos, **C_MAX** a quantidade máxima de números que podem ser escolhidos e **V** o valor da aposta unitária.

Saída

Seu programa deve calcular e imprimir, para cada **C**, variando de **C_MIN** a **C_MAX** (**C** $\leq$ **C** $\leq$ **C_MAX**) a quantidade de números jogados, a probabilidade e o valor da aposta, ao final de cada da impressão, colocar um duplo final de linha para colocar um espaço entre as apostas.

Exemplos de Entradas	Exemplos de Saídas
25 15 17 2.50	Quantidade numeros jogados: 15 Probabilidade: 1 em 3268760 Probabilidade: 0.000031 % Valor aposta: 1 X R$ 2.50 = R$ 2.50 Quantidade numeros jogados: 16 Probabilidade: 1 em 204298 Probabilidade: 0.000489 % Valor aposta: 16 X R$ 2.50 = R$ 40.00 Quantidade numeros jogados: 17 Probabilidade: 1 em 24035 Probabilidade: 0.004161 % Valor aposta: 136 X R$ 2.50 = R$ 340.00

60 6 10 3.00	Quantidade numeros jogados: 6 Probabilidade: 1 em 50063860 Probabilidade: 0.000002 % Valor aposta: 1 X R$ 3.00 = R$ 3.00 Quantidade numeros jogados: 7 Probabilidade: 1 em 7151980 Probabilidade: 0.000014 % Valor aposta: 7 X R$ 3.00 = R$ 21.00 Quantidade numeros jogados: 8 Probabilidade: 1 em 1787995 Probabilidade: 0.000056 % Valor aposta: 28 X R$ 3.00 = R$ 84.00 Quantidade numeros jogados: 9 Probabilidade: 1 em 595998 Probabilidade: 0.000168 % Valor aposta: 84 X R$ 3.00 = R$ 252.00 Quantidade numeros jogados: 10 Probabilidade: 1 em 238399 Probabilidade: 0.000419 % Valor aposta: 210 X R$ 3.00 = R$ 630.00

Problema H
Colhendo Ovos

Por Jorge Francisco Cutigi (IFSP – Campus São Carlos)

Arquivo: ovos.[c|cpp|java\cs\py]

***Timelimit:** 1*

Otto adora ir passear na casa dos seus avós e bisavós. A parte do passeio que ele mais gosta é quando ele vai colher os ovos direto no galinheiro. Então, Otto e seu vô Sérgio pegam caixas de ovos vazias e correm para o galinheiro. Chegando lá, passam em cada canto recolhendo os ovos e colocando nas caixas. O processo é simples: eles pegam a primeira caixa e vão colocando os ovos até completar. Assim que a caixa completa, eles pegam outra caixa. O processo se repete até acabar os ovos. Otto e seu vô Sérgio têm muitas caixas, então elas sempre são suficientes para armazenar todos os ovos.

Faça um programa que, dado a quantidade de ovos, informe quantas caixas foram usadas na coleta. Considere que em todas as caixas cabem uma dúzia de ovos.

Entrada

Um número inteiro **N** (0 <= **N** <= 10000), que representa o número de ovos.

Saída

A saída deve conter uma linha com um número inteiro que indica quantas caixas foram usadas, independe de estar completa de ovos ou não.

Exemplos de Entradas	Exemplos de Saídas
24	2
25	3
5	1
30	3