



MARATONA DE PROGRAMAÇÃO

InterFatecs

Fase 1 - 20/05/2017

Caderno de problemas

Apoio:



BE BLÜE®

moz://a

Realização:



www.interfatecs.com.br

1 Instruções

Este caderno contém 10 problemas – identificados por letras de A até J, com páginas numeradas de 3 até 24. Verifique se seu caderno está completo.

Informações gerais

1. Sobre a competição

- (a) A competição possui duração de 5 horas (início as 13:00h término as 18:00h);
- (b) NÃO é permitido acesso a conteúdo da Internet;
- (c) É vedada a comunicação entre as equipes durante a competição, bem como a troca de material de consulta entre elas;
- (d) Cada equipe terá acesso a 1 computador dotado do ambiente de submissão de programas (BOCA), dos compiladores, link-editores e IDEs requeridos pelas linguagens de programação permitidas;
- (e) NÃO é permitido o uso de notebooks ou outro tipo de computador ou assistente pessoal;
- (f) Os problemas têm o mesmo valor na correção. Procure identificar e resolver primeiro os problemas mais fáceis.

2. Sobre o arquivo de solução e submissão:

- (a) O arquivo de solução (o programa fonte) deve ter o mesmo nome que o especificado no enunciado (logo após o título do problema);
- (b) confirme se você escolheu a linguagem correta e está com o nome de arquivo correto antes de submeter a sua solução.

3. Sobre a entrada

- (a) A entrada de seu programa deve ser lida da entrada padrão (não use interface gráfica);
- (b) Seu programa será testado em vários casos de teste válidos além daqueles apresentados nos exemplos. Considere que seu programa será executado uma vez para cada caso de teste.

4. Sobre a saída

- (a) A saída do seu programa deve ser escrita na saída padrão;
- (b) Não exiba qualquer outra mensagem além do especificado no enunciado.

Problema A

Classificados para a final

Arquivo fonte: interfatecs.{ c | cpp | java }

Autor: Antonio Cesar de Barros Munari (Fatec Sorocaba)

A Maratona de Programação InterFatecs é uma competição que já mostrou sua importância para a comunidade tecnológica brasileira. Após várias edições consecutivas bem sucedidas, consolidou-se como um evento anual que aproxima as diversas Fatecs e permite experiências inesquecíveis para os participantes.

As equipes com melhor desempenho na prova da primeira fase são classificadas para a fase final, que é presencial, ou seja, essas equipes competirão entre si na escola designada como sede daquele ano. Muito interessante, por que tem a viagem com a galera, tem o contato com pessoal desconhecido das outras Fatecs que se interessam pelas mesmas coisas que você, tem a merenda também, camiseta, palestras, etc.

Este problema requer que determinemos os classificados para a fase final de uma edição da Maratona de Programação InterFatecs. A quantidade de vagas para a fase final é divulgada pela escola sede daquele ano, que considera a sua capacidade física e administrativa em acolher um certo número de equipes. A equipe melhor classificada de cada Fatec na primeira fase já está automaticamente classificada para a final. Se ainda restarem vagas a serem preenchidas, isso é feito com base no placar final da primeira fase.

Suponha que, além das equipes campeãs de cada Fatec, ainda restem 15 vagas a serem preenchidas. Então as 15 melhores equipes que não foram campeãs de sua escola serão as convidadas para preencher essas vagas. O critério de ordenação do placar final considera como elemento principal a quantidade de problemas resolvidos pela equipe naquela prova: quem resolveu mais problemas fica à frente de quem resolveu menos problemas. Se duas equipes resolveram a mesma quantidade de problemas, o desempate é pelo tempo total computado para a equipe, que é calculado automaticamente pelo ambiente de submissão usado na prova: quem gastou um tempo menor para resolver aqueles problemas ficará à frente de quem gastou mais tempo para resolver a mesma quantidade de problemas. Há ainda critérios adicionais de desempate que não serão necessários neste problema.

Sua tarefa é, dadas as quantidades de vagas disponíveis na final e de Fatecs participantes e os resultados de cada equipe na primeira fase, determinar os classificados para a fase final da competição.

Entrada

A entrada é iniciada por uma linha contendo os inteiros V ($F \leq V \leq 100$), F ($1 \leq F \leq 100$) e Q ($F \leq Q \leq 1000$) que indicam, respectivamente, a quantidade de vagas na final, a quantidade de Fatecs participantes e a quantidade de equipes que competiram na primeira fase. Seguem-se Q linhas, cada uma contendo os inteiros U ($1 \leq U \leq 1000$), E ($1 \leq E \leq 1000$), A ($0 \leq A \leq 20$) e T ($0 \leq T \leq 100000$) que informam os resultados de cada equipe participante, sendo U o número identificador da Fatec daquela equipe, E o número identificador da equipe, A a quantidade de problemas que acertou na prova e T o tempo total computado pela equipe. Assuma, por simplicidade, que não haverá equipes em situação de empate (ou seja, com a mesma quantidade de acertos e mesmo tempo total).

Saída

A saída deve indicar os identificadores das equipes classificadas para a fase final da competição. Os valores devem estar em ordem crescente do identificador, com uma vírgula e um espaço em branco após cada equipe, exceto a última, que deverá ser seguida apenas por um ponto final e uma quebra de linha, conforme mostra o exemplo de saída fornecido a seguir.

Exemplo de Entrada 1

```
10 5 20
2 4 0 0
4 6 5 489
2 5 7 728
1 1 5 600
4 3 6 699
1 15 4 183
1 17 9 1329
2 10 3 92
4 9 0 0
4 14 3 205
2 13 4 432
3 11 8 1040
2 16 9 1200
3 20 6 879
1 19 5 525
4 8 7 824
5 2 1 40
3 18 7 876
3 12 1 50
5 7 5 592
```

Exemplo de Saída 1

```
3, 5, 6, 7, 8, 11, 16, 17, 18, 20.
```

Problem B

Urinals

Source file: urinals.{ c | cpp | java }

Author: Antonio Cesar de Barros Munari (Fatec Sorocaba)

A very common behavior observed in male individuals is the choice of urinals inside public restrooms. In various cultures, male individuals prefer a free urinal whose neighbors are equally free. It seems that this is related to some sense of privacy when doing the famous "number 1". As we have recently seen a certain interest in unusual aspects of econometrics, there has even been a specific informal term to describe this problem: *International Choice of Urinal Protocol* <https://blog.xkcd.com/2009/09/02/urinal-protocol-vulnerability>. According to this very important protocol, the best scenario would be one in which between every male individual there is always a free toilet isolating the person from its nearest neighbor. It is clear that in situations where the Nature call is too strong the individual can ignore the protocol, probably at the cost of some psychological discomfort for their toilet neighbors.

Your goal in this problem is to determine, for a certain amount of concurrent urinal users, what would be the minimum quantity of urinals so that the best scenario described by the *International Choice of Urinal Protocol* is achieved.

Input

The input consists of an integer N ($1 \leq N \leq 500$), which indicates the number of people who wish to use an urinal at any given time.

Output

Print an integer corresponding to the minimum number of urinals required so that two people do not need to use neighboring urinals. Place a line break after the value.

Example of Input 1

1	1
---	---

Example of Output 1

Example of Input 2

2	3
---	---

Example of Output 2

Example of Input 3

42	83
----	----

Example of Output 3

Esta página foi propositadamente deixada em branco.

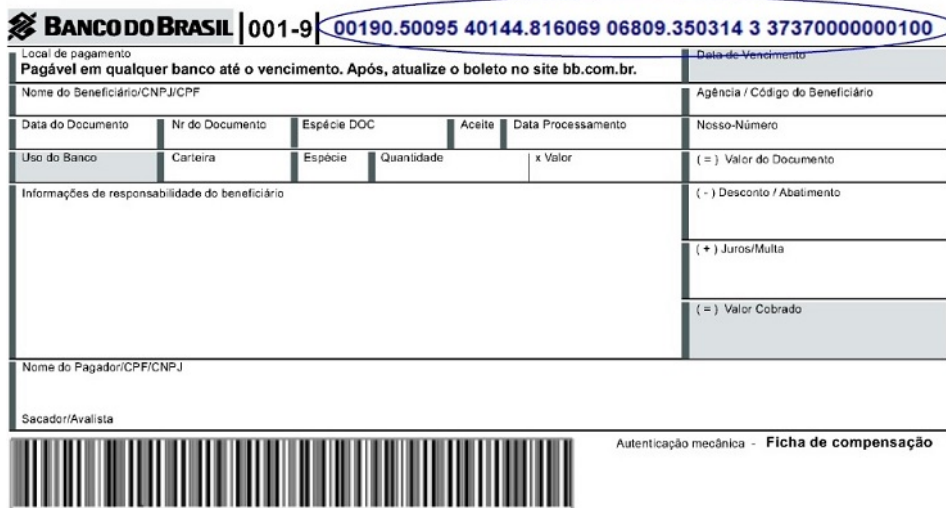
Problema C

Boleto

Arquivo fonte: boleto.{ c | cpp | java }

Autor: Antonio Cesar de Barros Munari (Fatec Sorocaba)

Afrânio trabalha no banco como programador e recebeu a incumbência de fazer a validação de um dos dígitos verificadores do código numérico dos boletos de cobrança. Sabe aquele número enorme que precisamos digitar no caixa eletrônico ou no computador quando não podemos usar o leitor de código de barras? Então, é esse mesmo. Todas as informações que existem no código de barras estão naquele número enorme, chamado tecnicamente de “Linha Digitável”. A figura 1 mostra um exemplo de boleto para refrescar a sua memória. Um daqueles dígitos é o chamado “dígito verificador do código de barras”, e ele é calculado por meio de uma variação de um algoritmo bastante conhecido no setor de Processamento de Dados chamado Módulo 11. A fórmula usada é bem simples: pegamos cada dígito numérico útil do código e o multiplicamos por um peso específico, acumulando o somatório desses produtos. Em seguida dividimos o valor da soma por 11 e guardamos o resto, que será então subtraído de 11. Esse resultado é o dígito verificador pelo módulo 11 e, na variação deste algoritmo utilizada neste problema, caso o valor obtido seja zero ou algo maior do que 9, consideramos o dígito verificador como sendo o inteiro 1. A título de exemplo, vamos considerar um boleto em que o somatório das multiplicações seja 445. O resto da divisão de 445 por 11 é 5 e 11 menos 5 produz o resultado 6, que é o dígito verificador esperado. Caso o somatório fosse 551, o resto seria 1 e 11 menos 1 dá 10 e, nesse caso, o dígito verificador esperado seria 1. Para determinar os pesos a serem usados na multiplicação de cada dígito, começamos pelo último dígito (aquele mais à direita) que é multiplicado por 2, o penúltimo por 3, o antepenúltimo por 4 e assim por diante, ignorando obviamente o próprio dígito verificador que faz parte do código lido. Nunca o valor do peso é maior do que 9, já que quando esse valor, na sequência, chegaria a 10 nós o trocamos pelo 2, reiniciando assim a progressão numérica.



BANCO DO BRASIL | 001-9 | 00190.50095 40144.816069 06809.350314 3 37370000000100

Local de pagamento: **Pagável em qualquer banco até o vencimento. Após, atualize o boleto no site bb.com.br.**

Nome do Beneficiário/CNPJ/CPF: _____ Agência / Código do Beneficiário: _____

Data do Documento: _____ Nr do Documento: _____ Espécie DOC: _____ Aceite: _____ Data Processamento: _____ Nosso-Número: _____

Uso do Banco: _____ Carteira: _____ Espécie: _____ Quantidade: _____ x Valor: _____ (=) Valor do Documento: _____

Informações de responsabilidade do beneficiário: _____ (-) Desconto / Abatimento: _____

_____ (+) Juros/Multa: _____

_____ (=) Valor Cobrado: _____

Nome do Pagador/CPF/CNPJ: _____

Sacador/Avalista: _____

Autenticação mecânica - **Ficha de compensação**

Figura C.1: Exemplo de boleto com código de barras e Linha Digitável em destaque.

Agora vamos aos problemas, por que até aqui está muito fácil. Nosso amigo Afrânio precisa calcular o dígito verificador do código de barras, mas a partir dos dados da Linha Digitável. Ocorre que infelizmente a sequência dos dados na Linha Digitável é diferente da ordem em que os mesmos dados aparecem no código de barras, e isso afeta a forma de cálculo, pois a fórmula apresentada anteriormente se aplica apenas

à sequência em que os dígitos aparecem no código de barras e não na Linha Digitável. As Tabelas 1 e 2 mostram a sequência aplicada em cada um dos casos.

Posição	Conteúdo
01 a 03	Código do Banco
04 a 04	Código da Moeda = 9 (Real)
05 a 05	Dígito Verificador (DV) do código de Barras
06 a 09	Fator de Vencimento
10 a 19	Valor
20 a 44	Campo Livre

Tabela 1: Sequência dos dados no código de barras do boleto.

BBBMC.CCCCX DDDDD.DDDDDY EEEEE.EEEEEZ K FFFFVVVVVVVVVV	
Onde:	
B	Código do banco
M	Código da Moeda
C	Campo Livre (dígitos de 20 a 24 do código de barras)
X	Dígito verificador do primeiro grupo
D	Campo Livre (dígitos de 25 a 34 do código de barras)
Y	Dígito verificador do segundo grupo
E	Campo Livre (dígitos de 35 a 44 do código de barras)
Z	Dígito verificador do terceiro grupo
K	Dígito verificador do código de barras (que Afrânio terá que calcular)
F	Fator de Vencimento
V	Valor

Tabela 2: Formatação e sequência dos dados na Linha Digitável do boleto.

A formatação utilizada na Tabela 2 é aquela adotada nos boletos, com pontos separando algumas partes e espaços em branco em outras. Repare que existem dígitos verificadores adicionais na Linha Digitável, que devem ser ignorados no cálculo. Considere também que o peso a ser aplicado sobre um determinado dígito útil da Linha Digitável é o mesmo peso que seria aplicado no cálculo diretamente sobre o código de barras. Assim, por exemplo, o último dígito do chamado Campo Livre (que corresponde à posição 44 do código de barras) deve ser multiplicado pelo peso 2. Na Linha Digitável esse mesmo dígito ocorre na posição 36 e, apesar disso, deve ser multiplicado também por 2 para respeitar a fórmula.

Seu objetivo é, para um código de boleto informado no formato da Linha Digitável apresentado anteriormente, determinar o valor correto do dígito verificador do código de barras (o elemento K da Tabela 2).

Entrada

A entrada consiste em uma string de 54 caracteres contendo o conteúdo referente a uma Linha Digitável, na formatação apresentada na Tabela 2. Os grupos numéricos terão os separadores padrão previstos: o ponto e o espaço em branco. O valor do dígito verificador do código de barras estará substituído pela letra X.

Saída

Imprimir o valor do dígito verificador do código de barras esperado para aquele código, seguido de uma quebra de linha.

Exemplo de Entrada 1

03399.20159 52800.234156 03210.501023 X 70630000095453

Exemplo de Saída 1

7

Exemplo de Entrada 2

03399.74644 70300.000000 00003.201019 X 70500000005000

Exemplo de Saída 2

1

Exemplo de Entrada 3

03399.74644 70300.000000 00002.801017 X 70210000005000

Exemplo de Saída 3

5

Exemplo de Entrada 4

03399.74644 70300.000000 00001.401017 X 68310000005000

Exemplo de Saída 4

4

Exemplo de Entrada 5

03399.20159 51000.234156 03201.501024 X 69060000093279

Exemplo de Saída 5

9

Esta página foi propositadamente deixada em branco.

Problema D

Bingo

Arquivo fonte: bingo.{ c | cpp | java }

Autor: Leandro Luque (Fatec Mogi das Cruzes)

Genovevo fundou uma organização sem fins lucrativos que promove eventos para arrecadar fundos e investir na educação de crianças carentes, segundo ele um dos únicos atos de caridade que valem a pena.

A organização está crescendo muito rapidamente e, para atender à crescente demanda, ele deseja criar um bingo eletrônico beneficente. Nesta modalidade, cada jogador ganha uma cartela eletrônica, o sistema faz o sorteio dos números e sintetiza a voz do número sorteado, simulando um bingo tradicional.

Ele pediu sua ajuda para implementar um programa que, dadas as cartelas e os números sorteados em um bingo, verifica em qual rodada algum dos jogadores ganhou (se algum ganhou). Para ganhar, um jogador deve preencher completamente uma linha ou coluna.

As cartelas podem ter qualquer quantidade de linhas e colunas, dentro de um limite pré-estabelecido. Ainda, algumas cartelas podem conter espaços com desenhos, que não precisam ser preenchidos pelo jogador. A seguir, são apresentados dois exemplos de cartelas.

Tabela 3: Exemplos de cartelas

10	5	1
3	4	25
32	45	99

10	5	1	33
3	4	25	29
32	45	Δ	49
54	87	50	81
27	55	48	62

Entrada

A primeira linha da entrada contém dois números inteiros L e C , ($3 \leq L, C \leq 9$), que especificam, respectivamente, o número de linhas e colunas das cartelas do bingo. A próxima linha contém um número inteiro J , ($2 \leq J \leq 100$), que especifica o número de jogadores que participarão do jogo. As próximas $J \cdot L$ linhas contém C números inteiros N cada, ($0 \leq N \leq 99$), especificando o conteúdo da cartela de cada um dos jogadores. Cada número é separado do subsequente por um espaço em branco. Um valor de N igual a 0 indica que a posição da cartela não contém um número, mas sim um desenho. A linha seguinte contém um número inteiro B , ($1 \leq B \leq 99$), especificando o número de bolas sorteadas. Finalmente, as B linhas seguintes contém um inteiro K cada, ($1 \leq K \leq 99$), especificando, em ordem, cada bola sorteada. Embora um jogo de bingo real pare quando algum jogador ganha, você deverá ler todas as bolas informadas, até o fim, mesmo que algum jogador ganhe no meio do jogo.

Saída

Caso apenas um jogador tenha ganhado, imprima uma linha contendo dois inteiros X ($1 \leq X \leq J$) e Y ($1 \leq Y \leq B$), separados por um espaço em branco, indicando, respectivamente, o número do jogador que ganhou e o número da rodada em que ele ganhou. Caso mais de um jogador tenha ganhado, imprima a palavra "EMPATE", em maiúsculas e sem aspas. Caso nenhum jogador tenha ganhado, imprima a palavra

"NADA", em maiúsculas e sem aspas. Ao final, não esqueça de inserir uma quebra de linha.

Exemplo de Entrada 1

```
3 3
2
11 15 44
33 59 28
22 10 81
21 11 49
73 12 98
97 55 66
10
3
5
14
59
22
12
28
33
99
98
```

Exemplo de Saída 1

```
1 8
```

Exemplo de Entrada 2

```
3 3
2
11 15 44
33 0 28
22 10 81
21 11 49
73 12 98
97 55 66
5
3
5
14
59
12
```

Exemplo de Saída 2

```
NADA
```

Problema E

Nomes

Arquivo fonte: nomes.{ c | cpp | java }

Autor: Danilo Ruy Gomes (Fatec Itapetininga)

O povo brasileiro tem uma criatividade para nomes inigualável. Cada família possui um jeito característico de nomear os filhos, uns são por nome de santos, outros por nomes de frutas e outros simplesmente inventam nomes baseados em nomes dos pais.

Um caso curioso é justamente o nome baseado num nome qualquer, exemplo se o nome do pai é José e nasce uma filha os pais resolvem colocar o nome da menina de Josefina ou ainda o nome da mãe é Maria e os pais acabam colocando o nome do filho de Maristelo e por aí vai. A criatividade é tão grande que atualmente existe até uma recomendação dos cartórios para que os nomes que podem cair numa situação de gozação, o tabelião não o registre ou mesmo sugira outro nome.

Pensando nessa ideia, você foi escalado para criar uma rotina para um sistema de cartório que faça a checagem com base nas iniciais do nome do pai ou da mãe e verifique se pelo menos os quatro caracteres iniciais forem iguais ao do pai ou da mãe. Neste caso, o sistema deve emitir a mensagem "VERIFICAR". Caso contrário, deve emitir a mensagem "NORMAL".

Entrada

A primeira linha da entrada consiste do nome do pai (ou da mãe) de no máximo 20 caracteres. Na segunda linha tem-se a quantidade de nomes a serem comparados Q ($2 \leq Q \leq 10$). As próximas linhas contém os nomes dos Q filhos, cada qual com no máximo 20 caracteres.

Saída

Seu programa deve imprimir na saída a palavra "VERIFICAR" para cada nome de filho com pelo menos 4 ou mais caracteres iniciais coincidentes com o nome do Pai ou da Mãe, e "NORMAL", caso contrário. Após a palavra, acrescentar uma quebra de linha.

Exemplo de Entrada 1

```
JOSE
3
JOSEFINA
JOSEMIRO
JOSIANE
```

Exemplo de Saída 1

```
VERIFICAR
VERIFICAR
NORMAL
```

Exemplo de Entrada 2

```
CARLOS
2
MARIO
JOSE
```

Exemplo de Saída 2

```
NORMAL
NORMAL
```

Esta página foi propositadamente deixada em branco.

Problema F

Mario

Arquivo fonte: mario.{ c | cpp | java }

Autor: Anderson V. de Araujo/Alessandro F./Lucas T. (UFMS)

A maioria dos competidores de programação já jogou ou pelo menos ouviu falar do clássico *Super Mario Bros.* Para aqueles que não conhecem, o jogo tem como foco principal a saga do herói Mario para salvar a princesa Peach das garras do vilão Bowser.

Na aventura de hoje, Mario mais uma vez se dirigia para enfrentar Bowser, quando acabou caindo em uma caverna subterrânea. Para sair dessa caverna e prosseguir sua jornada, Mario precisa encontrar um tubo que o transporte de volta a superfície. Mas para complicar a vida do nosso herói, devido à alta temperatura do local, durante sua caminhada dentro da caverna, ele perde energia.

A energia do nosso herói, como muitos devem saber, é medida por vidas. A cada movimento dentro da caverna, nosso herói perde um quarto de uma vida. Nosso herói pode se movimentar apenas para as células vizinhas que não contenham obstáculos e nas direções horizontal e vertical. Ele não pode se mover na diagonal ou para alguma célula com obstáculo. Sendo assim, a cada 4 células percorridas, nosso herói perde uma vida completa.

Dada a quantidade de vidas do nosso herói e o mapa da caverna, queremos saber se é possível que Mario alcance um tubo antes que suas vidas cheguem a zero. Lembrando que caso ele consiga chegar a um tubo no movimento que o faria perder a última vida, ele milagrosamente consegue escapar.

Entrada

A entrada é composta por uma linha contendo dois inteiros H e W ($1 \leq H, W \leq 1000$), indicando a altura e largura do mapa da caverna. As próximas H linhas contém W caracteres cada, representando o mapa da caverna. O mapa da caverna é representado por $H \times W$ células contendo os seguintes caracteres:

- '.' representa uma célula livre na caverna, onde Mario pode movimentar;
- 'x' representa um obstáculo na caverna, onde Mario não pode acessar;
- 'S' representa a posição inicial do Mario, ao cair na caverna; aparece uma única vez em todo o mapa;
- 'T' representa um tubo, por onde Mario pode escapar da caverna.

Por fim, após as H linhas, há uma linha contendo um inteiro C ($0 \leq C \leq 2000$), indicando o número de vidas de Mario ao cair na caverna.

Saída

Você deve imprimir a palavra **SIM** se é possível Mario escapar da caverna e **NAO** caso não seja possível que ele escape. Finalize com uma quebra de linha.

Exemplo de Entrada 1

```
1 6
.S...T
1
```

Exemplo de Saída 1

SIM

Exemplo de Entrada 2

```
1 6
S....T
1
```

Exemplo de Saída 2

NAO

Exemplo de Entrada 3

```
3 2
S.
.x
xT
1000
```

Exemplo de Saída 3

NAO

Exemplo de Entrada 4

```
2 6
S.x...
x...xT
2
```

Exemplo de Saída 4

SIM

Problema G

Questão de Lógica

Arquivo fonte: logica.{ c | cpp | java }

Autor: Lucio Nunes de Lira (Fatec São Paulo)

Mariazinha é muito inteligente e gosta de exercitar seu raciocínio resolvendo problemas de lógica e matemática. Recentemente, ela aprendeu que a soma de dois números naturais é par se ambos forem pares ou se ambos forem ímpares, e que a soma de um número par com um ímpar sempre gerará um ímpar.

Durante a aula de matemática, Mariazinha conheceu a "Sequência de Fibonacci", que é uma sequência de números naturais cujos dois primeiros termos são 1 e 1 e cada termo subsequente é a soma dos dois anteriores, conforme ilustrado na Figura 1.



Figura G.1: Amostra da Sequência de Fibonacci.

Querendo combinar seus conhecimentos, Mariazinha quer saber: dado um número natural N , que representa o N -ésimo termo da sequência de Fibonacci, esse número é par ou ímpar?

Entrada

A entrada tem um inteiro N ($0 < N < 2^{64}$), indicando o N -ésimo termo da sequência que será avaliado.

Saída

Deverá ser impresso "par" (sem aspas e minúsculo) ou "impar" (sem aspas, minúsculo e sem acentuação) caso o termo seja um número par ou ímpar, respectivamente, e uma quebra de linha.

Exemplo de Entrada 1

1

Exemplo de Saída 1

impar

Exemplo de Entrada 2

2

Exemplo de Saída 2

impar

Exemplo de Entrada 3

3

Exemplo de Saída 3

par

Exemplo de Entrada 4

18446744073709551615

Exemplo de Saída 4

par

Esta página foi propositadamente deixada em branco.

Problema H

Logística de Container

Arquivo fonte: `logistica.{ c | cpp | java }`

Autor: Julio Lieira (Fatec Lins)

Você acabou de ingressar na equipe de TI da empresa WA Exportações, a qual trabalha com exportações para empresas brasileiras. Uma empresa que queira exportar seus produtos deve colocar seus produtos em caixas e enviá-las para o Porto de Santos, onde atua a WA Exportações. Tais caixas são colocadas sobre Pallets (suporte de madeira) e acomodadas em Containers para serem transportados pelos navios. Cada Container tem 8 metros de comprimento. Os Pallets variam de 1, 2, 3 e 4 metros de comprimento. Para simplificar vamos assumir que os Pallets possuem a mesma largura e altura dos Containers.

A WA Exportações possui uma cota de 50 Containers e a logística de preenchimento dos Containers é a seguinte: cada Pallet recebe um número sequencial de identificação, iniciando em 1, e de acordo com seu comprimento é colocado no primeiro Container onde couber. Os Containers também possuem uma numeração sequencial iniciando em 1. Um novo Container é utilizado quando o anterior já está totalmente preenchido ou quando o Pallet a ser acomodado não cabe totalmente no Container atual. A Figura 1 ilustra o preenchimento dos Containers de acordo com a sequência de entrada do Exemplo 1. O número dentro do círculo é a identificação de cada Pallet dentro do Container, e recebe esta identificação de acordo com a ordem de chegada. O comprimento de cada Pallet (1, 2, 3 ou 4 metros) é mostrado no lado externo do Container e corresponde ao comprimento indicado na entrada.

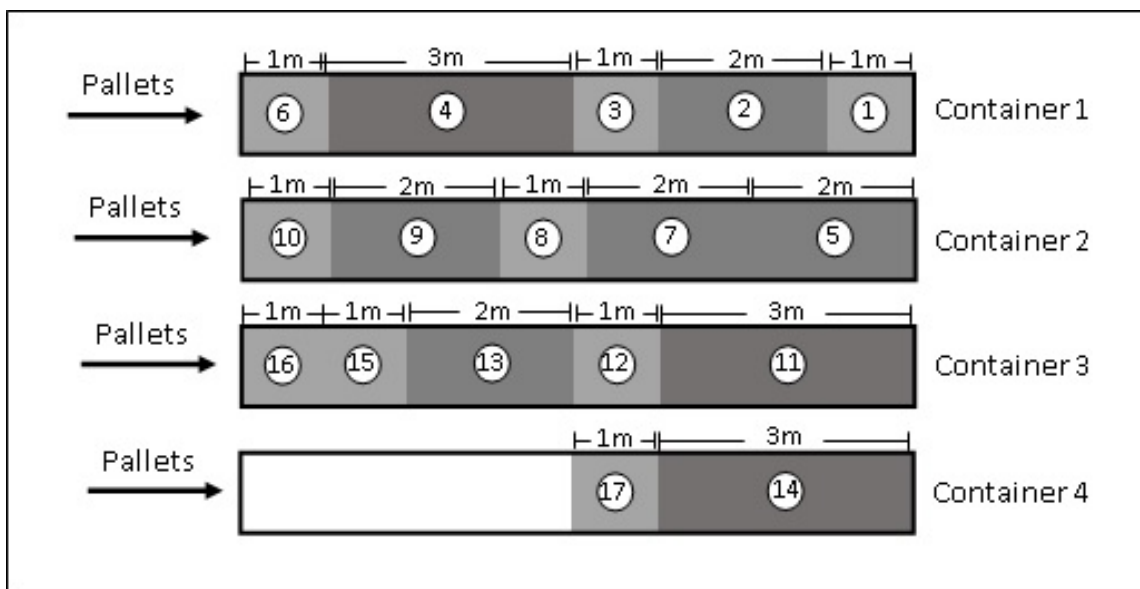


Figura H.1: Exemplo de preenchimento dos Containers pelos Pallets.

Um problema enfrentado pela empresa é a identificação do Container que contém um determinado Pallet depois que os mesmos já foram alocados. Seu “teste de batismo” na empresa será criar um sistema que, dado o número de sequência de um Pallet, dizer em qual Container ele foi alocado e também mostrar quais Pallets devem ser retirados para chegar até o Pallet procurado. Note ainda que os Pallets só podem entrar e sair do Container pela porta dianteira, indicada na Figura 1 pela seta.

Entrada

A primeira linha da entrada contém até 400 números, cada qual indicando o comprimento de um Pallet (1, 2, 3 ou 4 metros), e separados por um espaço em branco. O conjunto de pallets sempre caberá nos 50 containers disponíveis. Cada uma das próximas linhas contém um número inteiro N ($1 \leq N \leq 400$), indicando o número de sequência de um Pallet procurado.

Saída

Para cada Pallet procurado, imprima na saída o número do Pallet procurado, o número do Container e o número dos Pallets que devem ser retirados para se ter acesso ao Pallet dentro do Container. Imprima a palavra TOPO para indicar que não existem Pallets obstruindo o acesso ao Pallet procurado. Note que existe um espaço entre cada palavra e valor apresentado na saída. Finalize cada linha, inclusive a última, com uma quebra de linha.

Exemplo de Entrada 1

```
1 2 1 3 2 1 2 1 2 1 3 1 2 3 1 1 1
9
12
6
```

Exemplo de Saída 1

```
Pallet 9 Container 2 Retirar 10
Pallet 12 Container 3 Retirar 16 15 13
Pallet 6 Container 1 TOPO
```

Exemplo de Entrada 2

```
3 3 3 3 1 1 2 3 3
5
6
8
9
```

Exemplo de Saída 2

```
Pallet 5 Container 1 Retirar 6
Pallet 6 Container 1 TOPO
Pallet 8 Container 3 Retirar 9
Pallet 9 Container 3 TOPO
```

Exemplo de Entrada 3

```
4 4 3 3 3 2 2 2 2
3
5
6
9
```

Exemplo de Saída 3

```
Pallet 3 Container 2 Retirar 6 4
Pallet 5 Container 3 Retirar 8 7
Pallet 6 Container 2 TOPO
Pallet 9 Container 4 TOPO
```

Problema I

Funil de Estrelas

Arquivo fonte: funil.{ c | cpp | java }
Autor: Lucio Nunes de Lira (Fatec São Paulo)

Megan adora números! Ela recentemente começou a desenhar números de um jeito diferente: (I) primeiro escolhe um número natural N maior do que 1; (II) na primeira linha vaga escreve uma sequência decrescente, começando em N e terminando em 1; (III) na mesma linha repete a sequência, porém em ordem crescente; (IV) na linha abaixo os passos (II) e (III) são repetidos, porém subtraindo 1 de N e começando da coluna i (sendo i a linha atual), preenchendo os espaços vazios das colunas da esquerda e da direita com estrelas; (V) repete-se o passo (IV) até que não haja mais números.

Ficou confuso? Então veja a Figura 1, que ilustra exatamente o que Megan faz. Note que os números vão afunilando em um funil de estrelas!

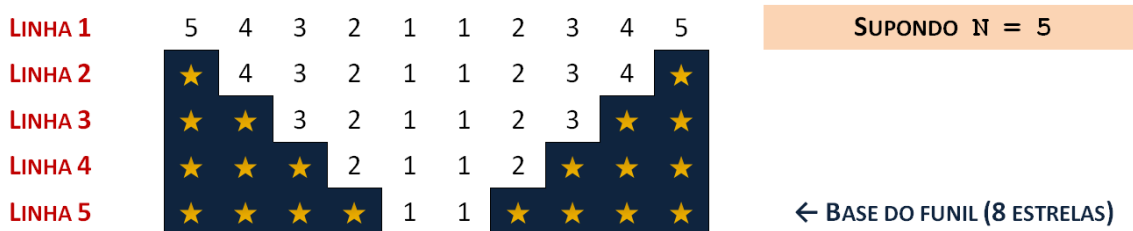


Figura I.1: Exemplo de um desenho feito por Megan com $N = 5$.

Dado um número N , e desenhando como Megan, qual será a quantidade de estrelas na base do funil?

Entrada

A entrada contém um número natural N ($1 < N < 1000$) representando a escolha de Megan.

Saída

Deverá ser impresso um número inteiro que represente a quantidade de estrelas da base do funil para o N informado e uma quebra de linha.

Exemplo de Entrada 1

2

Exemplo de Saída 1

2

Exemplo de Entrada 2

3

Exemplo de Saída 2

4

Exemplo de Entrada 3

4

Exemplo de Saída 3

6

Esta página foi propositadamente deixada em branco.

Problema J

Celular do Juvenal

Arquivo fonte: celular.{ c | cpp | java }

Autor: Antonio Cesar de Barros Munari (Fatec Sorocaba)

Juvenal é um hipster. E, talvez até por ser hipster, é esquisitão. E tem um celular antigão, por que ele acha isso muito descolado. E nesta fase de sua vida hipster, Juvenal só usa o aparelho para voz e texto, por que ele é descolado e o aparelho, limitado. A figura 1 mostra como é o celular do Juvenal. No teclado, se ele está digitando texto, com um toque ele consegue a letra ‘a’ (em minúsculas), com dois toques, consegue um ‘b’ (em minúsculas), com três toques, um ‘c’ (em minúsculas), com quatro toques, um ‘A’ (em maiúsculas) e assim por diante. Nesse modo de edição de textos, para digitar um espaço em branco ele precisa pressionar uma vez a tecla do zero, como mostra a imagem. Para um determinado texto digitado pelo hipster em seu



Figura J.1: Visão parcial do celular do Juvenal.

aparelho antigão, determine a quantidade mínima de vezes que ele precisa pressionar as teclas do celular a partir do momento em que entrou em modo texto.

Entrada

A entrada é composta por uma cadeia não vazia composta por caracteres alfabéticos maiúsculos e minúsculos e eventuais espaços em branco. O comprimento dessa cadeia é de no máximo 200 caracteres.

Saída

O programa deverá imprimir um inteiro correspondente à quantidade total de toques requeridos para a digitação da mensagem. Após o valor numérico deverá ser colocada uma quebra de linha.

Exemplo de Entrada 1

batata

Exemplo de Saída 1

7

Exemplo de Entrada 2

BatataA

Exemplo de Saída 2

13

Exemplo de Entrada 3

Batatinha quando nasce	42
------------------------	----

Exemplo de Saída 3**Exemplo de Entrada 4**

Centro Paula Souza	49
--------------------	----

Exemplo de Saída 4