MARATONA DE PROGRAMAÇÃO 2019-2

MARATONA ROBÓTICA PAULA SOUZA 2019-2

CADERNO DE PROBLEMAS

São Paulo – Brasil

Outubro, 2019

CADERNO DE PROBLEMAS
MARATONA DE PROGRAMAÇÃO 2019-2

Instruções

1) Este caderno contém 6 problemas (A-F). As páginas estão numeradas de 1 a 9, não contando a página de rosto. Verifique se o caderno está completo antes de começar;

2) Em todos os problemas, os programas deverão ler a entrada padrão para recebimento de dados externos e mostrar na saída padrão os resultados esperados;

3) Em todos os problemas, em que o final da entrada não esteja especificado no texto do problema, deverá ser considerado o final das entradas;

4) **Não é permitido** em nenhuma hipótese a utilização de código pronto. Os códigos poderão ser consultados de anotações e publicações em geral, desde que escritas ou impressas, e deverão ser digitados na hora da competição;

5) É **terminantemente proibido** a utilização da Internet para buscar códigos prontos ou qualquer outra informação, que não seja acesso e utilização da plataforma oficial da competição, isto é, o software BOCA;

6) Também **não será permitido** o uso de pendrives, hds externos, telefones celulares, tablets, ou qualquer outro dispositivo eletrônico, que não o único computador disponibilizado ao time;

7) Qualquer desrespeito às normas de restrições acima, o time será desclassificado;

8) Qualquer dúvida contate o pessoal do Staff.

CADERNO DE PROBLEMAS
MARATONA DE PROGRAMAÇÃO 2019-2

Dicas para Iniciantes

Aqui vão algumas dicas rápidas para quem não está acostumado com os tipos de problemas da maratona e de olimpíadas de informática em geral:

- Todo problema tem uma entrada exemplo e a saída esperada para aquela entrada. Você deve sempre se lembrar que essa entrada não é nem de perto a entrada que os juízes irão usar para testar seu programa. Lembre-se sempre de que é apenas um exemplo. Após fazer o programa funcionar para os exemplos, teste sempre condições extremas e de contorno (verifique os limites do problema, qual os valores mínimo e máximo que as variáveis podem atingir, etc.);

- Evite mandar programas precipitadamente. É bem melhor gastar 5 minutos testando o programa para ter certeza de que funciona para as mais diversas entradas do que receber uma mensagem de erro dos juízes (que acarreta uma penalização de 10 minutos);

- A maioria dos problemas sempre tem um "truque" escondido. Por exemplo, suponha um problema trivial de calcular fatorial. Você se lembrou de tratar o problema quando a entrada é zero? Sempre procure pensar no que o juiz deveria estar pensando para fazer a entrada. Afinal, o seu programa deve funcionar corretamente para qualquer entrada;

- Se o seu algoritmo lidar com busca, procure cortar ao máximo. Quanto menos nós expandidos, mais eficiente seu programa. Problemas de busca são, em geral, críticos com relação ao tempo, e um descuido pode acarretar em estouro do tempo limite!;

- A maratona é um torneio de criação e implementação de algoritmos. Isto significa que você nunca vai precisar se preocupar com coisas do tipo interface com o usuário, reusabilidade do código, etc. Aliás, é bem melhor usar nomes sugestivos de variáveis do que comentários;

- Evite usar as ferramentas de debug, elas consomem muito tempo. Aliás, deve-se evitar “debugar” o problema no computador, afinal, são três pessoas para apenas um computador! Enquanto você estiver “debugando” o programa na tela do computador, outros componentes do time podem estar querendo digitar um programa! O ideal é imprimir o código fonte (isso é permitido na maratona) e tentar analisar o código em busca de erros. Se for indispensável o “debug” em "tempo real" procure fazê-lo usando “print” de variáveis e técnicas do tipo. Às vezes breakpoints também podem ser úteis;

- Ninguém vai analisar seu código, portanto não interessa se ele está elegante ou não. Se você tiver uma solução “feia”, porém eficiente, essa é a que se deve usar;

- Nem sempre um algoritmo polinomial pode ser viável. Suponha que você tenha implementado um algoritmo O (n^3) para um determinado problema. Mas e se n puder assumir valores até, digamos, 1000? Com quase toda certeza seu problema irá estourar o limite de tempo. Por isso, sempre preste atenção não só à complexidade do seu algoritmo, como à viabilidade de sua implementação;

- Não há nada pior do que gastar muito tempo fazendo e implementando um algoritmo para, depois, descobrir que ele está errado. Numa maratona, esse erro é fatal. Por isso, apesar de a pressa ser necessária, procure sempre certificar-se da corretude do algoritmo ANTES de implementá-lo!

CADERNO DE PROBLEMAS
MARATONA DE PROGRAMAÇÃO 2019-2

Problema A
Palitinho

Arquivo fonte: Palito.*{ c | cc | java | py2 | py3 }*

Autores: Prof. Gildárcio Gonçalves (ETEC de São José dos Campos),
Prof. Hamilton Machiti (ETEC de São Bernardo do Campo)
Prof. Henrique Louro (ETEC de Caraguatatuba)

Tarefa

Jiló e Ferrugem são amigos inseparáveis. Sempre que estão juntos praticam seu passatempo predileto, jogar palitinho.

No jogo, cada jogador possui 5 palitos de fósforo. Podem jogar de dois a n jogadores por jogada. Em cada jogada, os jogadores colocam as mãos para trás das costas, com os 5 palitos em uma das mãos. Em seguida passam para a outra mão a quantidade de palitos desejados, ou seja, de 0 a 5. A um sinal combinado, todos colocam a mão com os palitos escolhidos para frente. Na sequência, cada um dos jogadores tenta adivinhar a soma dos palitos de todas as mãos à frente. Esse valor será 0 (também chamado de lona) até n X 5. Não é permitido repetir valor pelos jogadores. Ganha aquele que acertar a quantidade de palitos, depois que todos abrem as mãos e se somam os palitos escolhidos por cada jogador. Poderão ocorrer jogadas em que ninguém acerte.

Jiló e Ferrugem decidiram fazer torneios, entre eles, para ver quem era o melhor jogador de palitinho. Porém, perceberam que seria difícil controlar os resultados das n jogadas escolhidas. Assim, sabendo que você manjava de programação de computadores, pediram que criasse um programa para ajudá-los a calcular quem ganhou cada um dos torneios.

Entrada

O arquivo de entrada terá vários casos de teste. Cada caso, será composto por um torneio e terá várias linhas. A primeira linha de cada caso, conterá a quantidade de jogadas N, sendo 1 <= N <= 100. As N linhas seguintes conterão as quatro informações de cada jogada. Serão mostradas duas cadeias de caracteres com as quantidades de palitos (representados pela letra I) dos dois jogadores, e os palpites (P) de Jiló e Ferrugem, respectivamente, sendo 0 <= P <= 10, separados por um espaço. Quando qualquer um dos jogadores não escolher palito algum, ou se referir a zero no seu palpite, a palavra “lona” será mostrada. As entradas deverão ser lidas da entrada padrão. Uma linha com apenas um número 0 encerra as entradas.

Saída

Para cada caso de teste, seu programa deverá mostrar uma linha com o nome do jogador ganhador do respectivo torneio, ou seja, Jilo ou Ferrugem, ou no caso de ambos ganharem o mesmo número de jogadas, a palavra “Empate”. As saídas deverão ser escritas na saída padrão.

Exemplo de entrada	Saída para o exemplo de entrada
3 III IIII 5 lona I I 2 5 IIIII II 3 8 5 II III 3 5 II II lona 3 III II 5 4 I lona 2 1 IIII III 6 9 1 I I 3 1 0	Jilo Ferrugem Empate

CADERNO DE PROBLEMAS
MARATONA DE PROGRAMAÇÃO 2019-2

Problema B
Maratona

Arquivo fonte: Maratona.{ c | cc | java | py2 | py3 }

Autores: Prof. Gildárcio Gonçalves (ETEC de São José dos Campos),
Prof. Hamilton Machiti (ETEC de São Bernardo do Campo)
Prof. Henrique Louro (ETEC de Caraguatatuba)

Tarefa

Maratona é uma corrida realizada na distância oficial de 42,195 km, normalmente em ruas e estradas.

Uma cidade brasileira resolveu inovar e criou uma modalidade em que os atletas tinham que correr 211 Km em 5 dias. Ou seja, 42,2 Km por dia. Ganharia a competição o atleta que completasse o percurso total no menor tempo. Para tanto, o tempo de percurso individual de cada dia, é somado aos dos outros dias, e o atleta com menor tempo geral, seria o vencedor. Além disso, punições poderiam ser aplicadas aos competidores que infringissem algumas regras da competição. Essas punições seriam dadas em tempo que seriam somadas aos tempos de percursos.

Como tiveram muitos competidores inscritos, logo viram que seria um problema controlar e computar todos os tempos para definir os vencedores. Assim, pediram a você, exímio dominador das artes computacionais, que criasse um programa de computador com essa finalidade.

Entrada

A entrada é composta por um único caso de teste. A primeira linha traz um único número inteiro C, indicando o número de atletas inscritos na prova, onde 1 <= C <= 1500. As C linhas seguintes trazem o número N de inscrição do atleta, onde 1 <= N <= C e as informações dos tempos de cada competidor dia a dia. Serão 7 pares de tempos indicando o tempo do percurso diário e o tempo de punição se houver, no formato HH:MM:SS, onde 0 <= HH <= 23 que é a quantidade horas, 0 <= MM <= 59 que é a quantidade de minutos e 0 <= SS <= 59 que é a quantidade de segundos. Todos os valores separados por um espaço. As entradas deverão ser lidas da entrada padrão. As entradas encerram-se com uma linha contendo o número 0.

Saída

Como saída, seu programa terá que mostrar a classificação dos atletas por ordem de menor tempo. Ou seja, o atleta com menor tempo na primeira linha, o de segundo menor tempo na próxima e assim sucessivamente. Se houver empate, considerar a ordem de inscrição para o desempate. Assim, sua saída deverá conter C linhas. As saídas deverão ser escritas na saída padrão.

Exemplo de Entrada

```
5
1 02:45:30 00:00:00 02:50:02 00:00:00 02:47:50 00:03:00 02:46:35 00:00:00 02:48:20 00:00:00
2 03:01:20 00:03:00 03:02:10 00:00:00 03:00:50 00:00:00 03:02:05 00:00:00 03:03:10 00:00:00
3 02:50:02 00:00:00 02:45:30 00:00:00 02:46:35 00:00:00 02:47:50 00:00:00 02:48:20 00:03:00
4 02:44:25 00:00:00 02:47:30 00:00:00 02:45:02 00:00:00 02:48:15 00:00:00 02:49:10 00:00:00
5 03:10:10 00:00:00 03:05:01 00:00:00 03:02:00 00:00:00 03:01:50 00:00:00 03:05:06 00:00:00
0
```

Saída para o exemplo de entrada

```
4
1
3
2
5
```

CADERNO DE PROBLEMAS
MARATONA DE PROGRAMAÇÃO 2019-2

Problema C
Cifra de César

Arquivo fonte: Cesar.{ c | cc | java | py2 | py3 }

Autores: Prof. Gildárcio Gonçalves (ETEC de São José dos Campos),
Prof. Hamilton Machiti (ETEC de São Bernardo do Campo)
Prof. Henrique Louro (ETEC de Caraguatatuba)

Tarefa

A Cifra de César é o esquema de criptografia mais antigo que se conhece. Recebeu este nome em homenagem ao imperador romano Júlio César, que a usou para proteger importantes mensagens militares. Apesar de que, todas as mensagens de César eram escritas em Latim, naturalmente, o que as tornava incompreensíveis para a maioria das pessoas da época, ainda assim era necessário evitar que os inimigos tivessem acesso a elas. A Cifra de César é uma maneira simples de embaralhar uma mensagem escrita em uma linguagem que forma palavras a partir de um alfabeto. Ela implica em substituir cada letra de uma palavra pela letra que está a três letras de distância no alfabeto da língua. Assim, em uma mensagem em inglês, por exemplo, substitui-se cada A por D, cada B por E, cada C por F, e assim por diante. Continua-se essa abordagem até W, que é substituído pelo Z. Então, como não existem letras depois de Z, faz-se o padrão de substituição girar, substituindo-se X por A, o Y por B e o Z por C.

Entrada

A entrada é composta por vários casos de testes. Cada caso é apresentado em uma única linha, contendo uma cadeia (C) de caracteres criptografados, conforme explicação anterior, onde C é a quantidade mínima e máxima de caracteres na cadeia, sendo 1 <= C <=50. Deverá ser lida da entrada padrão e encerra-se com uma linha contendo o número 0.

Saída

Como saída, seu programa terá que mostrar uma linha para cada caso de teste, com a respectiva cadeia de caracteres da entrada descriptografada. As saídas deverão ser escritas na saída padrão.

Exemplo de Entrada	**Saída para o exemplo de entrada**
DEFGHI ABC WXYZ 0	ABCDEF XYZ TUVW

CADERNO DE PROBLEMAS
MARATONA DE PROGRAMAÇÃO 2019-2

Problema D
Jogo da Velha

Arquivo fonte: Velha.{ c | cc | java | py2 | py3 }

Autores: Prof. Gildárcio Gonçalves (ETEC de São José dos Campos),
Prof. Hamilton Machiti (ETEC de São Bernardo do Campo)
Prof. Henrique Louro (ETEC de Caraguatatuba)

Tarefa

Como todas as crianças em idade escolar sabem, o jogo da velha é um jogo de tabuleiro de 3 por 3 posições. Dois jogadores - X e O - se alternam colocando suas respectivas marcas nas células deste tabuleiro, iniciando pelo X. Se um jogador for bem-sucedido em obter três de suas marcas em uma linha, coluna ou diagonal, então será o vencedor. A graça do jogo está no fato de que é um exemplo simples de demonstrar como arranjos de duas dimensões podem ser usados em jogos de posição. A ideia básica é usar um arranjo bidimensional, matriz (array), para manter o tabuleiro do jogo. As células deste arranjo armazenam valores que indicam se a célula está vazia ou armazena um X ou O. Sendo o jogo uma matriz 3 x 3, ou seja, com 3 linhas e 3 colunas, cujas células da linha do meio consistem nas células [1] [0], [1] [1] e [1][2], por exemplo.

Entrada

A entrada é composta por vários casos de testes, sendo que cada caso corresponde a uma partida. Assim, cada caso é composto por 3 linhas e 3 colunas. Definiu-se que as células do arranjo seriam inteiros, com 0 indicando célula vazia, 1 indicando um **X** e -1 indicando **O.** Os inteiros em uma linha são separados por um espaço. Essa codificação permite uma maneira simples de testar uma dada configuração de tabuleiro vencedor para o X ou O, apenas testando se os valores de uma linha, coluna, ou diagonal somam 3, ou -3. A entrada encerra-se com uma linha contendo apenas um número 0.

Saída

Como saída, seu programa deverá mostrar o jogador vencedor X ou O ou indicar Empate.

Exemplo de Entrada	**Saída para o exemplo de entrada**
-1 1 -1 -1 1 1 1 -1 1 1 -1 1 -1 1 -1 1 -1 1 1 0 -1 1 -1 0 -1 0 0 0	Empate X O

CADERNO DE PROBLEMAS
MARATONA DE PROGRAMAÇÃO 2019-2

Problema E
Luzes

Arquivo fonte: Luzes.*{ c | cc | java | py2 | py3 }*
Autores: Prof. Gildárcio Gonçalves (ETEC de São José dos Campos),
Prof. Hamilton Machiti (ETEC de São Bernardo do Campo)
Prof. Henrique Louro (ETEC de Caraguatatuba)

Tarefa

No período escolar, Arthur gosta de aproveitar o tempo do intervalo brincando com os amigos da escola onde estuda. Uma das suas brincadeiras preferidas é o "Guri Acende-Apaga Luz". A brincadeira começa com a escolha de um dos participantes, o guri. Em seguida, os outros participantes definem um número e o guri deve passar por todas as salas da escola, que sejam múltiplos desse número, para inverter o estado da luz de cada sala – ou seja, se a luz estiver acesa, ele a apagará; se estiver apagada, ele a acenderá. A brincadeira segue com os participantes definindo novos números e o guri acendendo/apagando as luzes das salas múltiplas desses números. Quando o grupo desejar, ou o guri estiver muito cansado, ou algum professor reclamar, cada participante da brincadeira, com exceção do guri, é questionado sobre o estado das luzes das salas (na ordem do mais perto ao mais longe, tomando por base a entrada da escola que dá início à brincadeira). Aquele que acertar, ganha um sorvete e fica livre de ser guri durante todo o dia.

Como Arthur adora sorvete, ele procura de alguma maneira sempre ganhar a brincadeira. Para isso, ele pediu para o seu pai, um especialista em informática, para desenvolver um algoritmo que, dados os números que serão especificados pelos participantes, determinam o estado final das luzes das salas.

Entrada

A entrada é formada por vários casos de testes. A primeira linha contém dois inteiros X e Y, separados por espaço, indicando o número de salas (1 <= X <= 30), e a quantidade de números que serão informados pelos participantes (1 <= Y <= 100), respectivamente. Cada umas das N linhas seguintes contém um dos N números especificados pelos participantes (1 <= N <= X). A entrada encerra-se com uma linha contendo apenas um número 0.

Saída

Para cada caso de teste, imprima uma linha contendo X caracteres, indicando o estado de cada uma das luzes das salas da escola (o caractere mais à esquerda representa a primeira sala; o mais à direita representa a sala mais longe, tomando por base a entrada da escola que dá início à brincadeira). Caso a luz da sala esteja acesa, imprima o caractere 'L'. Caso esteja apagada, imprima o caractere 'D'.

Exemplo de Entrada	**Saída para o exemplo de entrada**
10 5 2 4 9 10 1 3 3 1 2 3 0	LDLLLDLLDL LDD

CADERNO DE PROBLEMAS
MARATONA DE PROGRAMAÇÃO 2019-2

Problema F
Bebedouro de Água

Arquivo fonte: Bebedouro.*{ c | cc | java | py2 | py3 }*
Autores: Prof. Gildárcio Gonçalves (ETEC de São José dos Campos),
Prof. Hamilton Machiti (ETEC de São Bernardo do Campo)
Prof. Henrique Louro (ETEC de Caraguatatuba)

Tarefa

O departamento de TI de uma empresa possui 3 andares. Em determinadas épocas do ano, os funcionários do departamento bebem muita água devido ao calor, mas como lá não há um bebedouro de água, os funcionários têm que trazer água de casa ou até mesmo comprar no barzinho ao lado. Por conta disso, o diretor do departamento de TI decidiu comprar um bebedouro para seus funcionários. Ele será instalado em um dos 3 andares, mas a instalação deve ser feita de forma que as pessoas não percam muito tempo subindo e descendo escadas.

Cada funcionário do departamento de TI bebe 1 garrafa de água por dia. Ele precisa ir do andar onde trabalha até o andar onde está o bebedouro e voltar para seu posto de trabalho. Todo funcionário leva 1 minuto para subir ou descer um andar e chegar ao bebedouro. Como o diretor do departamento de TI se importa muito com a eficiência, ele quer posicionar o bebedouro de forma a minimizar o tempo total gasto pelos funcionários, subindo e descendo escadas.

Sua tarefa é ajudar o diretor a posicionar o bebedouro de forma a minimizar o tempo total gasto pelos funcionários subindo e descendo escadas.

Entrada

A entrada consiste em 3 números, A1;A2;A3 (1 <= A1;A2;A3 <= 1000), um por linha, onde Ai representa o número de pessoas que trabalham no i-ésimo andar.

Saída

Como saída, seu programa deve imprimir uma única linha, contendo o número total de minutos a serem gastos com o melhor posicionamento possível do bebedouro.

Exemplo de Entrada	Saída para o exemplo de entrada
10 20 30 10 30 20 30 10 20 0	80 60 100

BOA SORTE!

Apoio: