MARATONA DE PROGRAMAÇÃO 2018-2

MARATONA ROBÓTICA PAULA SOUZA 2018-2

CADERNO DE PROBLEMAS

Caraguatatuba – São Paulo – Brasil

Novembro, 2018

CADERNO DE PROBLEMAS
MARATONA DE PROGRAMAÇÃO 2018-2

Instruções

1) Este caderno contém 5 problemas (A-E). 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 pen drives, 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 2018-2

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 2018-2

Problema A
Mega Dezenas

Arquivo fonte: dezenas.{ c | cc | java | py2 | py3 }
Autor: Prof. Henrique Louro (ETEC de Caraguatatuba)

Juvenal Apostador e aficionado por jogos de azar, principalmente os de loterias. Fica analisando os resultados dos jogos, procurando padrões nos números sorteados.

Numa dessas suas observações, percebeu que nos números sorteados da MegaSena, os números podem começar com até 7 dezenas diferentes, ou seja de 0 a 6. Assim, as unidades possuem dezena 0, os da primeira dezena 1 e assim sucessivamente até as dezenas começadas em 6. (Nos jogos da MegaSena, são aceitos números que vão de 1 até 60).

Ficou curioso para saber a quantidade de números, em cada sorteio começados com a mesmas dezenas. Por exemplo: num sorteio onde foram sorteados os números, 01, 02, 10, 14, 20 e 45, tivemos dois começados na dezena 0, dois na dezena 1, um na dezena 2 e 1 na dezena 4.

Como são muitos os sorteios já realizados, e como Juvenal não entende nada de programação de computadores, pediu a você que desenvolva um algoritmo computacional que possa identificar e mostrar esses padrões em cada um dos jogos já realizados.

Sua tarefa será desenvolver um algoritmo que receba vários casos de teste. Cada caso corresponderá a um dos sorteios da MegaSena, com seis dezenas. Deverá listar na saída padrão a quantidade de dezenas começadas com cada um dos 7 algarismos possíveis.

Entrada

Cada linha de entrada contém 6 inteiros *N* (1 <= *N* <= 60), representando os seis números sorteados em um dos sorteios da MegaSena, separados por um espaço. As entradas deverão ser lidas da entrada padrão. As entradas encerram-se com uma linha contendo o número 0.

Saída

Para cada saída apresente uma única linha composta por uma sequência de dois inteiros, separados por espaços, D e Q (0 <= D <= 6 e 1 < Q < 6), que representam o início das dezenas e a quantidade de números iniciados por elas, apenas das dezenas existentes em cada sorteio. As saídas deverão se escritas na saída padrão.

Exemplo de Entrada	**Exemplo de Saída**
1 2 10 14 20 45 12 15 30 31 51 52 20 21 25 45 59 60 0	0 2 1 2 2 1 4 1 1 2 3 2 5 2 2 3 4 1 5 1 6 1

CADERNO DE PROBLEMAS
MARATONA DE PROGRAMAÇÃO 2018-2

Problema B

Cartão de Respostas

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

Autor: Prof. Cláudio José Silva Gomes (ETEC de São José dos Campos)

Para facilitar a correção das questões da prova, a Profa. Diana fez todas as questões de múltipla escolha e distribuiu aos seus alunos um cartão de resposta com 10 questões com 5 alternativas (A, B, C, D e E), cada. Cada questão só tem uma alternativa correta.

Como são muitos alunos, a Profa. Diana pediu sua ajuda para criar um programa que a ajude a corrigir todos os cartões.

Exemplo do Cartão de Respostas:

ALTERNATIVAS	A	B	C	D	E
QUESTÕES					
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					

Entrada

A entrada é composta por vários casos de teste. Cada caso começa com uma linha que contém gabarito das 10 questões e a quantidade (Q) de alunos na sala, que fizeram a prova (1 <= Q <= 50). As demais Q linhas contém as 10 alternativas das questões de cada aluno, que deverão ser comparadas ao gabarito. 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 exibir uma linha com o total de pontos (P) de cada aluno, o percentual de acertos (A) e o de erros (E), separados por um espaço, onde 0 <= P =< 10, 0 <= A € Z_+ <= 100 e 0 <= E € Z_+ <= 100 (desprezar as casas decimais). As saídas deverão ser escritas na saída padrão.

Exemplo de Entrada	**Exemplo de Saída**
ABBEDCEAAC 4 ABAEDCEDAB AEECDDEDAB AEEEDEEDAB ABBEDCEAAC ACEDBADEBC 1 ACEDEADEDC 0	7 70 30 4 40 60 5 50 50 10 100 0 8 80 20

CADERNO DE PROBLEMAS
MARATONA DE PROGRAMAÇÃO 2018-2

Problema C
Tijolos

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

Autor: Prof. Márcio Rogério Santos Ferraz (ETEC Martinho Di Ciero - Itu)

Tiago é proprietário de uma casa de material de construção na cidade de São Paulo, especializada na venda de tijolos especiais de diversos tamanhos e medidas. Com o intuito de ajudar seus clientes e agilizar os pedidos, solicitou ao seu filho (estudante de Análise de Sistemas), um pequeno programa que faça o seguinte procedimento: Informar ao cliente o número de tijolos necessários para uma determinada área e o valor final a ser pago, tendo em vista que acima de mil tijolos, aplica-se desconto de 10% sobre o valor final.

Vamos ajudar o filho de Tiago a desenvolver esse programa! Após compreender a necessidade de Tiago e como funciona o cálculo da quantidade de tijolos para uma determinada área, monte um sistema que receba os dados (como mostrados abaixo) e apresente o número inteiro de tijolos necessários por metro quadrado, número inteiro de tijolos para toda a área (sempre desprezando os decimais) e o valor inteiro final a ser pago (desprezar os centavos).

Entrada

Cada caso de testes estará contido em apenas uma linha. O programa deverá estar preparado para receber vários casos de testes, ou seja, várias linhas. Cada linha será composta por 5 números inteiros separados por um espaço, A L C P V, onde: altura do tijolo (5cm <= A <= 20cm), largura do tijolo (5cm <= L <= 20cm), comprimento do tijolo (15cm <= C <= 50cm) perímetro do cômodo (8m <= P <= 200m), em metros e valor do milheiro (R$ 50 <= V <= R$ 2670). A entrada terminará com uma linha com um único número zero e deverá ser lida da entrada padrão.

Saída

A saída deverá produzir uma única linha para cada caso de testes contendo 3 números inteiros N T Q, separados por um espaço, onde: N é o número de tijolos por m^2, T número de tijolos para toda a área em m^2 e Q o valor final a pagar em R$ (desprezar os centavos). A saída deverá ser escrita na saída padrão.

Para tanto, são mostrados abaixo alguns dos cálculos necessários:

Área = perímetro X 2.80m (sempre considerando o pé direito com 2,8m)

Cálculo da quantidade de tijolos por m^2: $\frac{1}{(C+eh).(A+ev)}$ onde:

C = comprimento do tijolo
A = altura do tijolo
eh = espessura horizontal da argamassa (considerar 0.01, ou seja, 10mm)
ev = espessura vertical da argamassa (considerar 0.01, ou seja, 10mm)

Exemplo de Entrada	Exemplo de Saída
19 9 39 30 890 19 9 29 50 661 0	12 1008 897 16 2240 1480

CADERNO DE PROBLEMAS
MARATONA DE PROGRAMAÇÃO 2018-2

Problema D
O guerreiro da solda

Arquivo fonte: jose.{ c | cc | java | py }

Autor: Prof. Érico de Souza Veriscimo (ETEC de Guaianazes)

José é conhecido pelos amigos como o guerreiro da solda por conta de suas habilidades com o ferro de solda. Em uma eventual competição de robótica o guerreiro acabou quebrando seu pen drive, no qual continha seus arquivos e algoritmos. No entanto com suas habilidades soldou as trilhas danificadas do pen drive e o mesmo voltou a funcionar.

Além das competências com o ferro de solda, José também gosta muito de programar e deseja desenvolver uma aplicação que o ajude com a fila de dispositivos que tem para consertar. Porém, ele está muito ocupado resolvendo os problemas dos dispositivos e não tem tempo para desenvolver a aplicação, por isso pediu a sua ajuda para desenvolve-lo.

Seu algoritmo deve seguir os seguintes critérios:

- Os dispositivos ficam em uma fila, ou seja, o primeiro dispositivo que chega é o primeiro a ser atendido;
- Quando o dispositivo é de um familiar, passa a frente de 3 dispositivos;
- Quando o dispositivo é de um amigo, passa a frente de 2 dispositivos;
- Quando o dispositivo é da Vanessa (sua namorada), passa a ser a prioridade (primeiro da fila);
- Quando o dispositivo é do próprio José, só é consertado quando não há mais outro dispositivo na fila, ou seja, sempre que chega à vez de ser consertado ele volta para o final da fila, a não ser que seja o último dispositivo da fila.

Sua tarefa é: dada uma lista de dispositivos, apresentar a ordem em que os mesmos deverão ser consertados.

Entrada

A primeira linha contém um inteiro Q representado a quantidade de casos de teste (1 <= Q <= 50). Cada caso de teste contém um inteiro *N* (1 <= *N* <= 10000) representando a quantidade de dispositivos. As N linhas seguintes contém uma palavra *D* e um caractere *I* representando respectivamente, o nome do dispositivo e quem é o dono no mesmo. Caso *I* seja **N** o dono será uma pessoa qualquer, quando *I* for **F** significa que é de um familiar, quando *I* for **A** corresponde a um amigo, quando *I* for **V** o dispositivo é da Vanessa e quando *I* for **J** o dispositivo é do próprio José. A entrada deverá ser lida da entrada padrão.

Saída

Para cada caso de teste apresente uma lista com o nome dos dispositivos e o identificador do dono na ordem correta para o José consertar. Cada dispositivo e seu dono deverão estar em uma linha. Entre os casos de testes sempre imprima uma linha em branco. A saída deverá ser escrita na saída padrão.

Exemplo de Entrada	Exemplo de Saída
2 10 Pen1 N JPen J Pen2 N Pc V Pc2 N Monitor A Celular F Pen V abc N teste A 1 Pen N	Pen V Pc V Pen1 N Celular F Monitor A Pen2 N teste A Pc2 N abc N JPen J Pen N

CADERNO DE PROBLEMAS
MARATONA DE PROGRAMAÇÃO 2018-2

Problema E
Combinações

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

Autor: Prof. Henrique Louro (ETEC de Caraguatatuba)

Jair adora joguinhos de combinação em celular, daqueles em que você combina imagens de três em três, quatro em quatro, ou mais e ganha pontos. Quanto mais combinações, mais pontos, que geralmente ajudam a passar de nível.

Como Jair é um estudante iniciando na área de tecnologia da informação, criou um programa que gera várias cadeias de caracteres (letras maiúsculas), para que ele possa encontrar combinações de três, quatro ou cinco caracteres iguais, sequenciais.

Agora ele precisa criar uma rotina que encontre automaticamente essas combinações nas cadeias geradas e atribua a elas 10 pontos, quando forem de 3 caracteres iguais, 30 de quatro e 50 de cinco. No entanto, seus conhecimentos de programação ainda não permitem tal façanha.

Sabendo que você é um programador avançado, pediu ajuda nessa empreitada. Sua tarefa será desenvolver um algoritmo que receba várias sequências de caracteres aleatórios (letras maiúsculas). Você deverá encontrar em cada sequência as combinações descritas acima e atribuir a pontuação relativa. Na saída deverá mostrar a soma dessas pontuações em um único número inteiro para cada sequências recebidas.

Entrada

A entrada é formada por vários casos de testes. Cada linha da entrada contém uma sequência de caracteres aleatórios S, formada por letras maiúsculas, com uma quantidade de caracteres Q (3 <= Q <= 500). As entradas deverão ser lidas da entrada padrão. As entradas encerram-se com uma linha contendo o número 0.

Saída

Para cada caso de testes na entrada a saída deverá apresentar uma única linha com um número inteiro representando a soma das pontuações auferidas de acordo com as combinações encontradas, conforme descrito anteriormente. As saídas deverão ser escritas na saída padrão.

Exemplo de Entrada	**Exemplo de Saída**
AAAXYZBBBBMNOCCCCCZ	90
ABCDEFGHIJKLMNOOOPQRST	10
MNOPQRSTUVXYWZZZZ	30
ACEGIKMOQSUUUUU	50
BBCCDDEEEFFGGGGHH	40
ABCDEFGHHHHIJKKKLLLLLMMNNOOOPPQQR	100
0	

BOA SORTE!