Lista avaliativa 2

Esta é a lista avaliativa 2 da disciplina de Algoritmos e Programação Estruturada. A lista, que equivale a 2 pontos na média final, é composta de 3 questões. **A primeira questão vale 0.5 e as duas últimas questões (2 e 3) valem 0.75 cada.**

Para receber o valor total de cada questão, **o código enviado deve gerar 100% das saídas corretamente, de acordo com a descrição de cada problema.** Nesse sentido, teste sua solução no marvin, lembrando-se sempre de checar a escrita correta das mensagens de saída e se foram acrescentadas as quebras de linha em cada comando *printf*. **Caso apenas uma parte das saídas para os exemplos usados para corrigir determinada solução esteja correta, o aluno receberá 0. Isso significa que somente no caso de 100% das saídas estarem de acordo com o gabarito, o aluno receberá a nota da questão, o que torna a correção binária, ou seja, ou se acerta e recebe a pontuação total da questão ou a nota para essa questão é 0.**

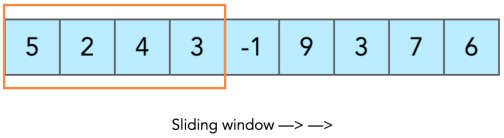
Soluções com erros de sintaxe (falta de ";", nome de funções ou tipos escritos errados, etc.) ou de execução (variáveis indefinidas, escolha do identificador de tipo inadequado, etc.) também serão penalizadas e receberão nota 0. Portanto, testem o código antes de enviá-lo!

Plágio não será tolerado também! Códigos claramente iguais, com alto grau de semelhança (mesmos nomes de variáveis, lógica e estrutura praticamente idêntica, comentários iguais, etc.) receberão nota 0!

A data de entrega da lista é dia 06/11, até às 23:59. O envio deve ser feito pelo AVA e deve conter um link para uma pasta do GitHub contendo os códigos que solucionam os problemas abaixo.

Questão 1

Crie um programa que percorra um conjunto de 10 valores por meio de uma janela deslizante de tamanho 2. A cada par de valores sequenciais encontrados, realize uma soma, gerando um novo elemento. Ao fim da primeira passagem, um novo conjunto terá sido gerado. Repita o processo de somas e geração de novas sequências para esse novo conjunto e assim por diante, até o processamento gerar apenas um elemento.
Um exemplo de uma janela deslizante de tamanho 4 pode ser vista no gif abaixo:



Entrada

A entrada é composta de 10 valores inteiros positivos separados por espaço, cada um entre 0 e 100.

Saída

A saída deve ser formada pela entrada, na primeira linha, seguida pelos novos conjuntos gerados, um em cada linha. Os elementos de cada conjunto devem ser separados **por um único espaço.** Após o último elemento de cada linha, **deve haver apenas uma quebra de linha**, ou seja, **não há espaços ao fim da linha.**

Exemplo de entrada	**Saída**
1 2 3 4 5 6 7 8 9 10	1 2 3 4 5 6 7 8 9 10 3 5 7 9 11 13 15 17 19 8 12 16 20 24 28 32 36 20 28 36 44 52 60 68 48 64 80 96 112 128 112 144 176 208 240 256 320 384 448 576 704 832 1280 1536 2816
1 1 1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 4 4 4 4 4 4 4 4 8 8 8 8 8 8 8 16 16 16 16 16 16 32 32 32 32 32 64 64 64 64 128 128 128 256 256 512

Questão 2

Crie um programa que permita checar qual time é o mais forte, tendo como base seu elenco titular e a força de cada jogador. Para isso, você deve ler o nome, a posição e a força de cada um dos jogadores e determinar a força ponderada de cada time. Para isso, deve ser usada a seguinte fórmula:

F = [8*G + 10(L1 + L2) + 5*(Z1 + Z2) + 8*(V1 + V2) + 11*(M1 + M2) + 12*(A1 + A2)]/100

Onde:
G = força do goleiro
L1 e L2 = forças dos laterais
Z1 e Z2 = forças dos zagueiros
V1 e V2 = forças dos volantes
M1 e M2 = forças dos meias
A1 e A2 = forças dos atacantes

Entrada

A entrada é composta por dois grupos de valores, cada um com 12 linhas. A primeira linha de cada grupo contém o nome do time, uma string de até 30 caracteres, que pode conter espaços. Já cada uma das próximas 11 linhas contém três informações, separadas por ponto e vírgula (“;”):

- O nome de um jogador, representado por uma string de até 30 caracteres, podendo conter espaços;
- Uma posição, representada por um caractere;
- Um nível de força, um inteiro entre 1 e 99;

Além disso, cada time possui um número limitado de jogadores por posição, sendo:

- 1 goleiro, representado pelo caractere “G”;
- 2 laterais, representados pelo caractere “L”;
- 2 zagueiros, representados pelo caractere “Z”;
- 2 volantes, representados pelo caractere “V”;
- 2 meias, representados pelo caractere “M”;
- 2 atacantes, representados pelo caractere “A”.

Saída

A saída deve mostrar a força ponderada de cada time, com 2 casas decimais, no formato abaixo, e qual dos dois é o mais forte.

Exemplo de entrada	**Saída**
Flamengo Diego Alves;G;80 Rafinha;L;85 Rodrigo Caio;Z;83 Pablo Marí;Z;84 Filipe Luís;L;87 William Arão;V;80 Gerson;V;88 De Arrascaeta;M;94 Everton Ribeiro;M;90 Bruno Henrique;A;91 Gabigol;A;90 River Plate Franco Armani;G;81 Gonzalo Montiel;L;79 Martínez Quarta;Z;80 Pinola;Z;78 Casco;L;81 Enzo Pérez;V;82 Palacios;V;83 Nacho Fernández;M;88 De la Cruz;M;89 Borré;A;88 Matías Suárez;A;87	Flamengo: 87.35 de forca River Plate: 84.05 de forca Flamengo eh mais forte

Questão 3

Crie um programa que realize operações (soma, subtração e multiplicação) com matrizes quadradas. A solução deve ler dois conjuntos de 16 valores cada, cada um representando uma matriz 4x4. Após ler os valores em si, deve ser lida uma operação a ser realizada e o resultado deve ser mostrado.

Entrada

A entrada é composta de 2 linhas, cada uma composta de 16 valores inteiros, entre 0 e 10. Os 16 primeiros valores formam a primeira matriz (A). Os outros 16 números irão compor a segunda matriz (B). Após isso, uma string representando uma operação deve ser lida, sendo “soma” para soma, “sub” para subtração e “mult” para a multiplicação.

Saída

A saída deve mostrar a matriz resultante da operação entre A e B, com cada um dos valores formatados em um campo de tamanho 4 justificado à direita, conforme os exemplos abaixo.

Observação: para justificar os valores de saída, siga o tutorial disponível nesse link: https://www.delftstack.com/pt/howto/c/printf-align-columns-in-c/. Um outro exemplo onde essa técnica é usada é esse exercício do Beecrowd: https://www.beecrowd.com.br/judge/pt/problems/view/1435

Exemplo de entrada	**Saída**
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 soma	2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
10 9 8 7 6 5 4 3 2 1 10 9 8 7 6 5 1 2 3 4 5 6 7 8 9 10 1 2 3 4 5 6 sub	9 7 5 3 1 -1 -3 -5 -7 -9 9 7 5 3 1 -1
1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 mult	10 20 30 40 11 22 33 44 12 24 36 48 13 26 39 52