Exercício de Revisão de Ponteiros em C

Implemente funções em C para cada um dos problemas abaixo, em um mesmo arquivo ("seu_nome.c"). Faça com que a função *main* chame adequadamente cada uma das funções.

Envie o arquivo "seu_nome.c" pelo Classroom. Use alocação dinâmica sempre que possível!!!

1. Implemente uma função que calcule a área da superfície e o volume de uma esfera de raio r. A área da superfície e o volume são dados, respectivamente, por $4\pi r^2$ e $4\pi r^3/3$. Essa função deve obedecer ao seguinte protótipo: `void calc_esfera (float r, float* area, float* volume);`

2. Implemente uma função que calcule as raízes de uma equação do segundo grau, do tipo ax2 + bx + c = 0. Essa função deve obedecer ao seguinte protótipo: `int raizes (float a, float b, float c, float* x1, float* x2);`

3. Implemente uma função que receba como parâmetro um vetor de números inteiros (vet) de tamanho n e retorne quantos números pares estão armazenados nesse vetor. Essa função deve obedecer ao protótipo: `int pares (int n, int* vet);`

4. Implemente uma função que receba como parâmetro um vetor de números inteiros (vet) de tamanho n e inverta a ordem dos elementos armazenados nesse mesmo vetor. Essa função deve obedecer ao protótipo: `void inverte (int n, int* vet);`

5. Implemente uma função que permita a avaliação de polinômios. Cada polinômio é definido por um vetor que contém seus coeficientes. Por exemplo, o polinômio de grau 2, 3x2 + 2x+ 12, terá um vetor de coeficientes igual a vet[] = {12, 2, 3}. A função deve obedecer ao seguinte protótipo: `double avalia (double* poli, int grau, double x)`, onde poli é o vetor de coeficientes; grau é o grau do polinômio; x é o valor da variável.

DESAFIO:

Implemente uma função que receba um vetor de inteiros (vet) de tamanho n. Essa função deve alocar dinamicamente um outro vetor também de tamanho n que contenha os endereços dos valores do vetor de inteiros de forma ordenada crescente, ficando a primeira posição do vetor de ponteiros o endereço do menor valor até a última posição, que conterá o endereço do maior valor. Essa função deve obedecer ao protótipo: `int** inverte2 (int n, int* vet);`