EXERCÍCIOS COM PONTEIROS E ALOCAÇÃO DE MEMÓRIA

Instrução para a resolução das questões:

- ***Não utilize variáveis estáticas, ou seja, utilize somente ponteiros e alocação dinâmica de memória para armazenar as informações necessárias de cada programa.***

1. [**PONTEIRO E MALLOC**] Escreva um programa em C que leia 2 números reais e imprime a média deles.

2. [**PONTEIRO E MALLOC**] Escreva um programa em C que leia 3 números inteiros e informa o tipo de triangulo formado pelos 3 números.
   - *Equilátero* : 3 lados iguais
   - *Isósceles* : 2 lados iguais
   - *Escaleno* : todos os lados diferentes

3. [**PONTEIRO E MALLOC**] Escreva um programa em C que leia um número inteiro e determina se é um número perfeito.
   Um número é perfeito se a soma dos seus divisores é igual ao próprio número. Por exemplo, o número 6 é perfeito, pois a soma dos seus divisores - 1 + 2 + 3 – é igual a 6.
   A lógica do programa para determinar se um número é perfeito deve ser implementada com uma função.

```
int perfeito(int *n)
```

4. [**PONTEIRO COM VETOR]** Escreva um programa em C que leia um número inteiro N e, em seguida, aloque memória suficiente para armazenar N números inteiros.
   Escreva também uma função que receba os N números inteiros e retorne o maior número dentre eles.

```
int maior(int *vetor)
```

   Obs: *NÃO use a notação de vetor (colchetes) da linguagem C para manipulação dos elementos do vetor, utilize somente aritmética de ponteiros.*

5. [**PONTEIRO COM STRUCT**] Escreva um programa em C que leia duas datas (dia, mês e ano) e calcule a diferença de dias entre as duas datas.
   Utilize um tipo de dado heterogêneo (***struct***) para representar a data.
   O cálculo da diferença de dias entre as datas deve ser implementado como uma função.

```
int diferencaDias(struct Data *data1, struct Data *data2)
```

6. **[PONTEIRO COM STRUCT]** Escreva um programa em C que define uma struct para representar um ponto cartesiano (x, y), leia dois pontos e calcule a distância entre os pontos.

   O cálculo da distância entre os dois pontos deve ser implementado como uma função.

   Utilize um tipo de dado heterogêneo (struct) para representar o ponto cartesiano (ponto2D).

```
float distancia(struct Ponto2D *ponto1, struct Ponto2D *ponto2)
```