EXERCÍCIOS DE ESTRUTURAS (STRUCT)

1. Implementar uma função para calcular o tempo em minutos entre o início e o fim de um evento, utilizando a seguinte struct:

```
struct horaMinuto {
   int hora;
   int minuto;
};
```

O protótipo da função é:

```
int duracao (struct horaMinuto inicio, struct horaMinuto fim)
```

2. Implementar uma função para imprimir o conteúdo da struct Pessoa abaixo:

```
struct Endereco {
   int numero;
   char rua[50];
   char bairro[30]
   char cidade[40]
};

struct Pessoa {
   char nome[40];
   char sexo;
   struct Endereco end;
};
```

O protótipo da função é: `void imprimir(struct Pessoa pessoa)`

3. Implementar uma função para imprimir os dados de uma pessoa utilizando as seguintes estruturas

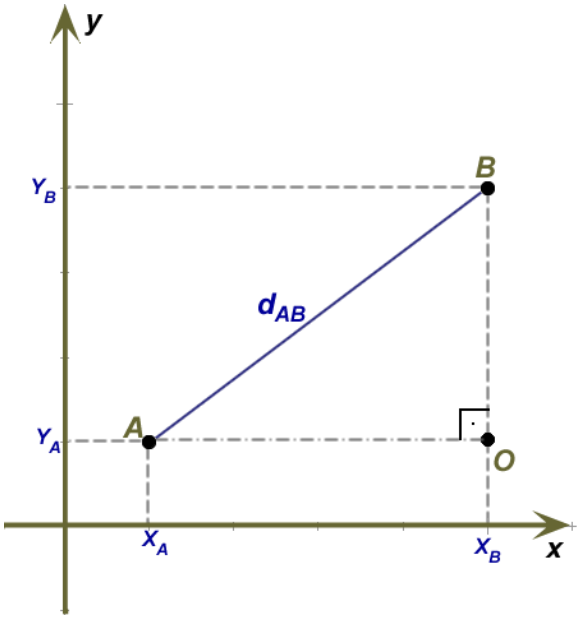
```
struct Data {
      int dia;
      int mes;
      int ano;
};

struct Pessoa {
      char nome[40];
      char sexo;
      struct Data nascimento;
};
```

O protótipo da função é: void imprimirPessoa(struct pessoa)

Escreva um main() para testar a função.

4. A distância entre dois pontos é dada pelo cálculo: $D = \sqrt{(xb - xa) + (yb - ya)}$



Utilizando a struct Ponto abaixo, faça uma função que calcule e retorne se um ponto está dentro (1) ou fora (0) de um círculo.

```
struct ponto {
      int x;
      int y;
};
```

O protótipo da função é:

```
int dentroDoCirculo(struct ponto centro, int raio, struct ponto p)
```

5. Crie uma função para retornar 0 ou 1, indicando se um ponto está dentro ou fora de um triângulo. Utilize a seguinte estrutura:

```
struct triangulo {
   struct ponto p1;
   struct ponto p2;
   struct ponto p3;
};
```

O protótipo da função é:

```
int dentroDoTriangulo(struct triangulo trian, struct ponto p)
```

6. (para casa. Escrever o algoritmo antes) Implementar uma função para calcular a quantidade de dias percorridos entre duas datas, armazenadas em uma struct Data. O protótipo da função é:

```
int duracao (struct Data inicio, struct Data fim)
```

7. (para casa. Escrever o algoritmo antes) Implementar uma função para dizer se uma data está ou não dentro de um período. Caso verdadeiro, a função retorna 1, senão retorna 0. O período é representado pela struct:

```
struct Periodo {
   struct Data inicio;
   struct Data fim;
};
```

O protótipo da função é:

```
int duracao (struct Periodo periodo, struct Data data
```

8. Implementar uma função para ler os dados da struct Pessoa, utilizando um ponteiro para esta struct. Utilizar a função para imprimir a pessoa e verificar o resultado.

9. Centro de gravidade de um corpo é um ponto central em relação a todos os pontos compostos por este corpo. Levando isto em consideração, implemente uma função que calcule e retorne o centro de gravidade para um conjunto de pontos. Utilize o seguinte protótipo: struct ponto calcularCentroDeGravidade(struct corpo c). A struct corpo é definida como:

```
struct corpo {
      struct ponto [50];
      int quantidade;
};
```

Esta estrutura suporta no máximo 50 pontos, e a quantidade indica quantos pontos o corpo possui. Um corpo pode ser composto de no mínimo 1 ponto e no máximo 50.

PONTEIROS PARA STRUCT

1. Criar uma função para alocar e ler os dados de uma pessoa e retornar o ponteiro para o main(). Criar outra função para imprimir os dados da pessoa alocada.
   A assinatura de cada função é:
   Pessoa * lerPessoa();
   void imprimirPessoa(Pessoa * alguem);

```
typedef struct Data {
   int dia;
   int mes;
   int ano;
} Data;

typedef struct Pessoa {
   char nome[40];
   char sexo;
   Data * nascimento;
} Pessoa;
```

2. Implementar uma função para calcular o tempo em minutos entre o início e o fim de um evento, utilizando a seguinte struct:

```
typedef struct horaMinuto {
   int hora;
   int minuto;

} HoraMinuto;
```

   O protótipo da função é:
   `int duracao (HoraMinuto * inicio, HoraMinuto * fim);`

   Criar um main() que declare de maneira normal as duas variáveis e faça a leitura, e passe apenas seus ponteiros para a função.

3. Implementar uma função para alocar e ler a hora/minuto de início e fim de um evento. Atualizar o exercício anterior para utilizar esta função.
   O protótipo da função é: `HoraMinuto * lerMomentoEvento();`

4. A distância entre dois pontos é dada pelo cálculo: $D = \sqrt{(xb - xa) + (yb - ya)}$
   Utilizando a struct Ponto abaixo, faça uma função que calcule e retorne se um ponto está dentro (1) ou fora (0) de um círculo. Criar também uma função para alocar e ler os valores de um ponto.

```
typedef struct ponto {
```

```
   int x;
   int y;
} Ponto;
```

O protótipo de cada função é:

```
Ponto * lerPonto();
int dentroDoCirculo(Ponto * centro, int raio, Ponto * p);
```

5. Implementar uma função para ler do teclado a hora e minuto da seguinte struct:
6.

```
typedef struct horaMinuto {
      int hora;
      int minuto;
} HoraMinuto;
```

O protótipo da função é: `void lerDataHora(HoraMinuto * horaMin);`
Criar um main que aloque a sctuct e chame a função para efetuar os testes. Utilizar a função do exercício 2 para imprimir a duração o conteúdo lido.

7. Implementar a função do exercício anterior utilizando a seguinte protótipo:
   `void lerDataHora(HoraMinuto horaMin);`

8. Criar as funções abaixo para ler os pontos de vários quadrados (que compõem uma região) e um outro ponto, de forma que seja possível informar se o ponto está dentro ou fora de algum quadrado da região. Utilizar as seguintes structs:

```
typedef struct ponto {
   int x;
   int y;
} Ponto;

typedef struct quadrado {
   Ponto * p1;
   Ponto * p2;
} Quadrado;

typedef struct regiao {
   Quadrado * quadrados[10]
} Regiao;
```

A assinatura das funções são:

```
Regiao * criarRegiao(); //aloca uma região
void lerQuadrado(Regiao * regiao); //aloca um quadrado e seus pontos, lê do teclado os pontos e insere o quadrado na região
Ponto * lerPonto(); //aloca o ponto, lê seus valores e retorna seu ponteiro
```

`int estaDentro(Regiao * regiao, Ponto * ponto);` // retorna 1 se o ponto estiver dentro de algum quadrado da região, caso contrário retorna 0.

Observação: criar um main() para ler alguns pontos, podem ser lidos de zero a 10 quadrados. As posições do vetor de quadrados que não possuir um quadrado deve ter o valor NULL.

9. Uma empresa de transporte quer armazenar as informações dos passageiros que viajam em um ônibus. O ônibus é representado por um vetor de ponteiros para passageiros. As estruturas são as seguintes:

```
typedef struct Bagagem {
   int numero;
   int peso;
   int tipo; //1 - mala, 2 - bolsa
} Bagagem;

typedef struct Passageiro {
   char * nome;
   int lugar;
   Bagagem * bagagem;
} Passageiro;

typedef struct Onibus {
   int quantidadePassageiros;
   Passageiro * poltronas [40];
} Onibus;
```

Os passageiros devem ser alocados à medida que são lidos do teclado. Os lugares no ônibus que não possuem passageiros devem possuir o valor NULL (ponteiro nulo). Caso o passageiro possua bagagem ela deve ser alocada, caso contrário ela também deve ser NULL.

O main() deve fazer a leitura do número da poltrona e dos dados do passageiro. O main() deve chamar as seguintes funções para manipular os dados.

```
int estaCheio(Onibus * onibus); //retorna 0 ou 1
int estaDesocupado(Onibus * onibus, int numeroDaPoltrona); //retorna 0
ou 1
int quantidadeDeLugaresLivres(Onibus * onibus);
void incluirPassageiro(Onibus * onibus, Passageiro * pass);
void adicionarBagagem(Passageiro * pass, Bagagem * bag);
```