1) Structs

Uma estrutura é um agregado de várias variáveis, possivelmente de tipos diferentes. Em C define-se uma estrutura utilizando-se o termo *struct*.

Frequentemente há a necessidade dos programas modelarem conjuntos de dados que não são homogêneos como, por exemplo, informações a respeito de um estudante que poderiam incluir o nome, o número de matrícula e a média das notas. Um tipo de dados para esse conjunto poderia usar uma *string* de caracteres para o nome, um número inteiro para o número da matrícula, um número real para a média das notas.

A criação de uma estrutura em C tem a forma geral:

```
struct nome_estrutura {
   tipo_1  nome_campo_1;
   tipo_2  nome_campo_2;
   ...
   tipo_n nome_campo_n;
} variaveis_estrutura;
```

O termo `nome_estrutura` é o nome dado para a estrutura, definida por seus diversos campos. A ocorrência de `variáveis_estrutura` é opcional, sendo utilizadas para definir nomes de variáveis do tipo da estrutura, como mostra o exemplo:

```
struct aluno{
      char nome[40];
      int numero;
      float media_notas;
} a1, a2;
```

2) Operações em Estruturas

As operações principais sobre estruturas são atribuição e referência aos campos da estrutura. As referências aos campos individuais são especificadas por meio do operador ponto (.) e do nome de cada campo:

```
struct aluno{
  char nome[40];
  int numero;
  float media_notas;
} a1,a2;

main(){
      strcpy(a1.nome , "Joao Marcelo");
      a1.numero = 3342;     // referência ao campo número
      a1.media_notas = 87.8;
      printf("Nome: %s",a1.nome);
      …
      a2=a1;     // atribuição
      printf("Nome: %s",a2.nome);
      …
}
```

Como mostra o trecho de programa anterior, podem-se atribuir duas variáveis de tipo estrutura, desde que sejam do mesmo tipo. Neste caso, todos os campos de primeira serão copiados na segunda.

Pode-se também ter um ponteiro para uma estrutura. Um exemplo de declaração poderia ser:

```
struct aluno *aluno_especial;
```

Um ponteiro para uma estrutura funciona como um ponteiro para qualquer outro tipo de dado em C: pode-se apontar para uma variável *struct* já existente ou alocar memória (usando, por exemplo, a função malloc:

```
int main(){
      struct aluno *aluno_especial;
      aluno_especial = (struct aluno *) malloc (sizeof (struct
aluno));
      strcpy((*aluno_especial).nome, "Carlos Pontes");
      …
}
```

Exercícios:

1) Escreva um programa para criar uma *struct* chamada Ponto, contendo as coordenadas inteiras x e y do plano. Declare 2 pontos, leia as coordenadas x e y de cada um e calcule e imprima a distância entre os pontos.

2) Usando a *struct* aluno definida na seção anterior, crie um programa que armazena num vetor os dados de todos os alunos de uma turma (20 alunos no máximo) e retorne os cálculos:

- Média geral da turma
- Listagem de todos os nomes de alunos que não obtiveram média igual ou superior a 60.

3) Crie um programa para armazenar uma agenda de contatos pessoais usando estrutura. Faça funções para (i) pesquisar por nome, (ii) pesquisar por telefone e (iii) listar contatos. Um contato deve ser composto por nome, endereço, telefone e e-mail. Use alocação dinâmica para o armazenamento do nome do contato. Para definir e armazenar a lista de contatos, use um vetor de estrutura, também alocado dinamicamente.

Bom trabalho!!