# Exercícios de Estrutura

##########################################################################

## Fáceis

**1.** Declare uma estrutura C capaz de guardar os seguintes dados relativos a um prontuário médico:

a) Nome do paciente (string de 50)

b) Endereço (string de 50)

c) Telefone (string de 8)

d) Altura (float)

e) Data de nascimento (string de 8)

f) Peso (float)

Declare um novo tipo C chamado Paciente que deve ser exatamente a estrutura acima.

No main declare duas variáveis do tipo desta estrutura, leia valores e imprima o paciente mais alto.

**2.** Crie uma estrutura para representar uma casa com quantidade de quartos, cor, área e quantidade de portas. No main crie duas casas, faça a leitura para elas e depois imprima na tela. OBS.: Defina o tipo de dados como um ponteiro para a estrutura.

**3.** Faça um programa para trabalhar com Professores. Este programa deverá ler os dados de 10 Professores. No final o programa deve exibir a soma do salário dos Professores.

OBS.: Um Professor deve possuir nome, endereço, salário, idade, sexo, estado civil, data de admissão e quantidade de disciplinas.

**4.** Uma pessoa possui nome, endereço e telefone. Utilizando estrutura, fazer um programa em C que permita a entrada dos dados de 3 pessoas e os imprima em ordem alfabética. Declare um novo tipo C chamado Pessoa que deve ser exatamente a estrutura acima. (Considerar para ordem alfabética somente a primeira letra). Ao

definir o tipo de dados, utilizar ponteiro para a estrutura.

# Médias

**5.** Elaborar um algoritmo que auxilie no controle de uma fazenda de gado. Cada cabeça de gado possui as seguintes informações (Fazer esta questão sem ponteiro e depois com ponteiro):

- código: código da cabeça de gado,
- leite: número de litros de leite produzido por semana,
- alim: quantidade de alimento ingerida por semana - em quilos,
- nasc: data de nascimento - mês e ano,
- abate: 'N" (não) ou 'S' (sim).

O campo nasc. é do tipo struct data que por sua vez, possui dois campos:

- mês
- ano

Declare um novo tipo C chamado Cabeca_de_gado que deve ser exatamente a estrutura acima. No main, leia duas cabeças de gado e depois imprima os valores da vaca que produz mais leite por semana.

**6.** Seja a seguinte estrutura:

```
struct time {
int hora;
int min;
int seg;
}
```

Seja a seguinte declaração:

```
struct time tempo; // variável global
```

Fazer um programa contendo funções para:

a) Receber uma quantidade de tempo em segundos e transformar a mesma em horas, minutos e segundo, preenchendo e imprimindo a variável tempo.

b) Ler a variável tempo (campos hora, min e seg) e devolver a quantidade de horas em segundos.

c) principal: chama as duas funções acima.

**7.** Fazer um programa para simular uma agenda de telefones. Para cada pessoa deve-se ter os seguintes dados:

- Nome
- E-mail
- Endereço (contendo campos para Rua, numero, complemento, bairro, cep, cidade, estado, país)
- Telefone (contendo campo para DDD e número)
- Data de aniversário (contendo campo para dia, mês, ano)
- Observações : Uma linha (string) para alguma observações especial.

a) Definir a estrutura acima.
b) Declarar a variável agenda (vetor) com capacidade de agendar até 100 nomes.
c) Definir a função busca por primeiro nome: Imprime os dados da pessoa com esse nome (se tiver mais de uma pessoa, imprime para todas)
d) Definir a função busca por mês de aniversário: Imprime os dados de todas as pessoas que fazem aniversário nesse mês.
e) Definir a função busca por dia e mês de aniversário: Imprime os dados de todas as pessoas que fazem aniversário nesse dia e mês.
f) Definir a função insere pessoa: Insere por ordem alfabética de nome.
g) Definir a função retira pessoa: retira todos os dados dessa pessoa e desloca todos os elementos seguintes do vetor para a posição anterior.
h) Definir a função imprime agenda com duas opções:

- imprime nome, telefone e e-mail
- imprime todos os dados.

i) O programa deve ter um menu principal oferecendo as opções acima.

# Difíceis

**8.** Seja uma estrutura para descrever os carros de uma determinada revendedora, contendo os seguintes campos:

marca: string de tamanho 15

ano: inteiro

cor: string de tamanho 10

preço: real

a) Escrever a definição da estrutura carro.

b) Declarar o vetor vetcarros do tipo da estrutura definida acima, de tamanho 20 e global.

c) Definir uma função para ler o vetor vetcarros.

d) Definir uma função que receba um preço e imprima os carros (marca, cor e ano) que tenham preço igual ou menor ao preço recebido.

e) Defina uma função que leia a marca de um carro e imprima as informações de todos os carros dessa marca (preço, ano e cor).

f) Defina uma função que leia uma marca, ano e cor e informe se existe ou não um carro com essas características. Se existir, informar o preço

**9.** Seja uma estrutura contendo dados dos funcionários de uma empresa. Para cada funcionário tem-se os seguintes dados:

nome: string de tamanho 20

CPF: vetor de 11 inteiros

numpeças: inteiro

salário: real

Seja os seguintes trechos de programa:

```
// declaração de variáveis globais:

struct funcionário vetfunc[50] ;
int nfunc;
main( ) // definição da função principal:

{
leitura ( );
calc_salario( );
tot_folha( );
printf(" Total de peças fabricadas no mês = %d \n", tot_peças( ) );
```

```
op_maior_sal( );
}

void leitura( ) // definição da função leitura:
{
int i,j;
printf("Digite número de funcionários (<= 50): ");
scanf("%d", &nfunc);
for (i = 0; i < nfunc; i++){

 le_CPF( );
while (ver_CPF( ) == 0){

 printf(" \n Erro na digitação do CPF. \n"):
le_CPF( );
}

le_outros_dados( );
}
}
```

Escrever a definição das funções que estão faltando. 

Observações:
Obs.1: A função leitura() chama três funções:

- le_CPF(): executa a leitura dos 11 dígitos que compõe o CPF
- ver_ CPF(): verifica se o CPF é valido, se for retorna o valor 1, caso contrário retorna 0.
- le_outros_dados(): Executa a leitura dos campos nome e num_peças.

Para verificar a validade do CPF (ou CIC):
Um número de CPF é seguido de dois dígitos denominados dígitos de controle. Estes dígitos são gerados a partir dos dígitos que compõem o número de um CPF e acompanham este número como sufixo. Digitando-se um número de CPF é possível computar os seus dígitos de controle e compará-los com os fornecidos ao sistema. Se os dígitos computados não batem com os dígitos fornecidos, então o número do CPF é falso ou ocorreu um erro de digitação. Se eles batem, então as chances de que o número seja correto são muito altas.

Seja CPF = x[0]x[1]...x[8]x[9]x[10], onde x[i] representa um dígito do CPF para 0 <= i <=8 e x[i] um dígito de controle para 9 <= i <= 10.

Exemplo:

Seja o CPF: 1 0 3 1 2 4 9 2 1 X[9] X[10].
O dígito de controle X[9] é obtido da seguinte maneira:

multiplicar os dígitos da esquerda para a direita por um número começando de 1 e incrementado de 1 (de X[0] até x[8]):

Ex.: 1*1, 0*2, 3*3, 1*4, 2*5, 4*6, 9*7, 2*8, 1*9

Somam-se as parcelas obtidas:

Ex.: 1 + 0 + 9 + 4 + 10 + 24 + 63 + 16 + 9 = 136

Obtem-se o resto da divisão inteira desta soma por 11: 138 % 11 = 4
4 corresponde ao dígito X[9]

O dígito de controle X[10] é obtido da seguinte maneira:
multiplicar os dígitos da esquerda para a direita por um número começando de 9 e decrementado de 1 (de X[0] até x[8]):

Ex.: 1*9, 0*8, 3*7, 1*6, 2*5, 4*4, 9*3, 2*2, 1*1

Somam-se as parcelas obtidas:

Ex.: 9 + 0 + 21 + 6 + 10 + 16 + 27 + 4 + 1 = 94

Obtem-se o resto da divisão inteira desta soma por 11: 94 % 11 = 6
6 corresponde ao dígito X[10]

 *Se o resto da divisão for igual a 10, deve-se considerar como 0. Ex.: 98%11 = 10  0*

A função deve verificar se os dígitos de controle fornecidos pelo usuário, obedecem a regra acima.

Obs.2: A função calc_salario() é responsável por calcular o campo salário de cada funcionário, através da seguinte regra que considera o número de peças fabricadas:

- Para número de peças menor ou igual a 30: recebe salário mínimo (R$230,00)
- Para número de peças entre 31 e 45: salário mínimo mais 3% do salário por peça acima das iniciais.
- Para número de peças acima de 45: salário mínimo mais 5% do salário por peça acima das 30 iniciais.

Obs.3: A função tot_folha( ) calcula e imprime o valor da folha de pagamento da empresa.

Obs.4: A função tot_peças( ) retorna o número total de peças fabricadas na empresa.

Obs.5: A função op_maior_sal( ) imprime os dados do operário de maior salário.

**10.** Seja um algoritmo para controlar os produtos do estoque de um supermercado. Para cada produto, tem-se os seguintes campos:

nome: string de tamanho 15
setor: caracter
quantidade: inteiro
preço: real //preço por unidade do produto

a) Escrever a definição da estrutura produto.
b) Declarar o vetor estoque do tipo da estrutura definida acima, de tamanho 100 e global.
c) Definir uma função para ler o vetor estoque.
d) Definir uma função que receba um setor e devolva o número de diferentes produtos desse setor.
e) Definir uma função que calcule e devolva o total de capital investido em produtos do supermercado.
f) Definir a função principal.