Montagem de Cadastro

Analise o programa Cadastro e monte uma pequena agenda com nomes e telefones.

Sugestães:

- Defina claramente as opçães do programa (exemplo: Adicionar, Buscar, etc).
- Defina as funçães a serem criadas (exemplo: add_nome, find_nome, etc).
- Crie o Algoritmo do Programa e de cada Rotina.
- Divida a Tarefa entre os componentes do grupo.
- Crie um pseudo código e depois (depois mesmo) digite o programa em C.

Lembramos que o Turbo C possui mais recursos que o Classic C, porém erros neste compilador são potencialmente muito mais complicados de serem encontrados e corrigidos do que falhas similares no Classic C.

O pré-processador

O pré-processador é executado antes da compilação propriamente dita. Ele atua como se fosse um processador de textos "inteligente", fazendo substituiçães de textos, inclusães de arquivos e até processamento condicional.

Todas as diretivas para o processador devem ser iniciadas com o sinal #, que deverá estar no início da linha.

#define

Sintaxe:

```
#define <identificador> <definição>
```

Exemplo:

```
#define PI 3.14159
#define quadrado(a) (a)*(a)
```

Assim se no programa existir:

```
y = quadrado(x-2);
```

a expressão será interpretada como y = (x-2) * (x-2);

#include

Sintaxe:

```
#include <arquivo>
```

Exemplo:

```
#include "stdio.h"
```

Devemos observar que existem poucos comandos na linguagem C, e que a maioria das funções deve ser escrita em C. Isto realmente ocorre, porém os produtores dos compiladores C fornecem todas estar funções previamente escritas para seus

compiladores, necessitando apenas serem "incluídas" nos programas pelo programador conforme for necessário.

Nada impede que desenvolvamos novas bibliotecas específicas a nossas necessidades que poderão facilmente ser transportadas entre ambientes (D.O.S. e Unix por exemplo).

#undef

Sintaxe:

```
#undef <identificador>
```

Faz com que o #define atribuído a algum identificador seja cancelado.

#if, #else e #endif

A diretiva if <expressão> faz com que, caso a expressão seja diferente de zero, o código entre a linha do #if até o #else seja incluídos, senão será incluído o trecho entre #else e o #endif.

Exemplo:

```
#define teste 100
#if teste > 50

#else
#endif
```

#ifdef e #ifndef

ifdef <identificador> permite verificar se certo identificador foi definido.

Se existir é retornado Verdadeiro.

ifndef <identificador> permite verificar se certo identificador foi definido.

Se não existir é retornado Verdadeiro.

```
#define pi 3.14159
main()

#ifndef eu
printf("'eu' não foi definido.\n");
#endif

#ifndef pi
printf("Nunca será executado, pois pi existe!\n");
#endif
```

Uniões

Imaginemos um caso onde tenhamos um programa muito grande e que precisemos executá-lo sem que possamos fazer qualquer redução em suas estruturas. Caso pudermos usar o mesmo local de memória para armazenarmos duas variáveis distintas, resolveremos nosso problema.

Exemplo:

```
main()
{
union data
  {
int dia;
int mes;
int ano;
  };
union data d;
d.dia = 30;
printf("%d\n",d.dia);
d.mes = 3;
printf("%d\n",d.mes);
printf("%d\n",sizeof(d));
}
```

Montagem do Cadastro (Aula 25L)

- Digitação do Exemplo Acima.
- Critique a forma de recuperação dos dados previamente digitados.

Teoria/Laboratório - Integrações

Integração C com Clipper

Devido ao fato da linguagem Clipper ter sido escrita em C, a integração das duas linguagens é relativamente simples, porém devemos observar as seguintes condições:

1. As variáveis C tem uma tipologia mais rica (não suportada pelo Clipper), de forma que a passagem de variáveis entre Clipper e C tem que levar em consideração esta característica. Felizmente a Nantucket (produtora original do Clipper) providenciou os fontes em C, extend.h e nandef.h, que devem ser compilados juntamente com as rotinas em C a serem utilizadas pelo Clipper.
2. A Rotina deve ser declarada sem utilizar a palavra main() e a função dever ser precedida pela palavra CLIPPER (Exemplo: CLIPPER apaga()).
3. A biblioteca LLIBCR.LIB (da Microsoft), deve ser linkeditada juntamente com as bibliotecas CLIPPER, EXTEND.
4. Para rotinas profissionais, sugerimos a utilização do linkeditor BLINKER (da BLINK), ou o RTLINK (da Nantucket), que criam executáveis mais eficientes que àqueles criados pelo TLINK (Borland), LINK (Microsoft) ou PLINK (Phoenix), embora todos estes possam ser utilizados.
5. Devido ao fato do Clipper ter sido escrito em C, mais especificamente no Microsoft C, preferencialmente este deve ser o compilador a gerar o código objeto para ser posteriormente ligado ao código Clipper. Outro compilador poderá eventualmente gerar código que resulte em alguma incompatibilidade imprevista.

Exemplo: Inverter Cores de Setor da Tela

Programa em Clipper

```
TelaLs = 04
TelaCe = 14
TelaLi = 23
TelaCd = 67
Tela = SaveScreen(TelaLs,TelaCe,TelaLi,TelaCd)
Atributo = "RB/N"
C_CHCOLOR(@Tela,Atributo)
Return
```

Função em C

```
#include <dos.h>
#include <stdio.h>
#include <conio.h>
#include <bios.h>
#include <mem.h>
#include <string.h>
#include <nandef.h>
```

```
#include <extend.h>
/* SO ACEITA LETRAS MAIUSCULA */
#define B       0x01
#define G       0x02
#define R       0x04
#define W       0x07
#define BB      0x10
#define BG      0x20
#define BR      0x40
#define BW      0x70
#define I       0x08
#define BLI     0x80
#define MAXATRIB 20
#define VIDEO   0xB800

CLIPPER c_chcolor()
{
register int i,j;
unsigned int len_tel;
char     *tela,atr ;
if ( PCOUNT != 2 || !ISCHAR(1) || !ISCHAR(2) ) {
        printf( "parametro incorreto em c_chcolor \n" ) ;
        exit( 2 ) ;
    }
tela    = _parc( 1 );
len_tel = _parcsiz( 1 );
atr     = _atr_color( _parc(2) , _parclen(2) );
for ( i=1 ; i<len_tel ; i=i+2 )
        tela[ i ] = atr ;
retni(0) ;
return   ;
}
/* str_atr - 'WNBGR*+/WNBGR*+' retorna o caracter atributo da cor */
int _atr_color( char *str_atr , int len_atr )
{
register int i ;
char atr=0x00 ;
int  charpos ;

if (( charpos = _charpos( str_atr, '/' , len_atr )) != 0 )  {
        for ( i=1+charpos ; i < len_atr && str_atr[ i ] != ',' ; i++ ) {
                switch(str_atr[i])      {
                        case 'B' : atr = atr|BB  ; break ;
                        case 'G' : atr = atr|BG  ; break ;
                        case 'R' : atr = atr|BR  ; break ;
                        case 'W' : atr = atr|BW  ; break ;
                        case '*' : atr = atr|BLI ; break ;
                        case '+' : atr = atr|I   ;
                }
    }
  }

for (i = 0;i < len_atr && str_atr[i] != '/' && str_atr[i] != ',';i++){
        switch(str_atr[i]) {
             case 'B' : atr = atr|B   ; break ;
           case 'G' : atr = atr|G   ; break ;
           case 'R' : atr = atr|R   ; break ;
           case 'W' : atr = atr|W   ; break ;
                        case '*' : atr = atr|BLI ; break ;
                        case '+' : atr = atr|I   ;
                  }
          }
        return( atr );
}


/* retorna a posicao do caracter na string */
int _charpos( char *string , char charac , int len_str )
{
register int charpos ;
for( charpos = 0 ; charpos < len_str ; charpos++ )
        if( string[charpos] == charac )
                return( charpos ) ;
        return( 0 ) ;
}
```

Integração C com Btrieve

Gerenciadores de Arquivo como o Btrieve (Novell) ou Gerenciadores de Bancos de Dados como o Oracle (Oracle), podem ser acessados por programas escritos em C, desde que estes recebam/passem parâmetros da forma especificada pelo gerenciador de Arquivo/Banco de Dados. De forma a exemplificar a utilização de C integrado a gerenciadores, segue trecho de programa em C, abrindo um Arquivo de Dados (Btrieve) e escrevendo no mesmo um registro. Obviamente, é necessário profundo conhecimento em Gerenciadores, para a elaboração de programas utilizando-se deste recurso.

Exemplo: Busca de uma Chave (Função 9 do Btrieve)

```
strcpy(chave,"AMARELO"); /* string */
```

```
funcao = 9; /* inteiro */

buffer = 128; /* inteiro */

canal = fp; /* ponteiro de arquivo */

/* retorno é uma variável previamente criada para acomodar o tamanho de

um registro */

status = 0; /* inteiro */

call btrv(funcao,canal,buffer,chave,retorno,status);
```

Status deverá assumir valor 4 se a chave não existir, 0 se existir e um número qualquer entre 1 e 99 indicando que tipo de erro ocorreu.

Biblioteca DBE e ODBC

A recente tendência de mercado em voltar a utilizar linguagens de 3ª geração (principalmente C, BASIC, COBOL e Pascal), que não possuem intrinsecamente gerenciadores de Base de Dados como as linguagens xBase (Clipper, dBase, Fox, etc), criou novamente mercado para os gerenciadores de arquivo como os que existiam até meados da década de 80 (MicroB, KISSBasic, Btrieve).

De fato o Btrieve, por ser o mais utilizado e também por pertencer a uma empresa bastante poderosa (Novell, que adquiriu a SoftCraft produtora original deste programa), foi o único gerenciador que "sobreviveu" a era xBase. Com o retorno das linguagens acima citadas e com o desinteresse da Novell em adequar o Btrieve aos novos dialetos surgidos das linguagens de 3ª geração, criou-se um mercado promissor que Borland (DBE) e Microsoft (ODBC) estão agora disputando. Em favor da Microsoft temos o fato de seu pacote interagir muito bem com o Visual BASIC e o Visual C, pacotes de grande utilização no mercado americano (principalmente o primeiro). Os pontos positivos da Borland residem no suporte de seu gerenciador ao acesso por registro, além do acesso via SQL (Structed Query Language), que é a linguagem padrão (sic) de acesso aos Bancos de Dados mais utilizados (Oracle, SyBase, Informix e Ingres), além da perfeita interação de seu programa com o Borland C++, a versão de linguagem C mais utilizada para desenvolvimento nos Estados Unidos.

Projeto em C (Aulas 26L)

1. Elabore sistema para Controle de Estoque de Peças, contendo arquivo de produtos e de movimentos.
2. Elabore Jogo Senha. O computador deverá permitir que dois humanos joguem entre si, atuando apenas como validador do resultado e indicando quantos dígitos estão no local correto/próximo (bom/ótimo).

APÊNDICE I - BIBLIOGRAFIA E PROGRAMA DO CURSO

Bibliografia:

Básica

- Diagramas de Blocos - Dirceu D. Salvetti

EDUSP

- Turbo C - Herbert Schildt

Makron Books

Complementares

- Linguagem C - Thelmo J.M. Mesquita

Érica

- Linguagem Algorítmica - Dirceu D. Salvetti

EDUSP

- Programação Sistemática - Niklaus Wirth

Campus

Programa do Curso:

- Apresentação (Aula 01T)
- Linguagem C (Aula 02T) - HELLO.C
- C- Uma Visão Geral e Instruções de Entrada e Saída (Aula 03T)
- Instruções de Entrada e Saída (Aula 04T) - QUADR.C
- Tomada de Decisão (Aula 05T) - MAIOR.C
- Tipos de Dados (Aula 06T) - FAT1.C até FAT5.C
- Variáveis e Operadores (Aula 07T) - DOSTIME.C, TIPOS.C, FAHRCELC.C, REPRES.C, INC1.C e INC2.C
- Tomadas de Decisão - Parte II (Aula 08T) - SWITCH.C
- Loops (Aula 09T) - PRIMO1.C até PRIMO3.C e FOR.C
- Comandos Desestruturadores (Aula 10T) - ALO_VEJA.C
- Matrizes (Aula 11T)
- Ordenação (Aula 12T) - ORDENA.C
- Ponteiros - Apresentação (Aula 13T)
- Ponteiros - Conceitos de Endereços (Aula 14T) - PONT_7.C e PONT_1.C
- Ponteiros - Conceitos Avançados (Aula 15T) - PONT_3.C, PONT_4.C, PONT_2.C, PONT_5.C, INV1.C e INV2.C
- Ponteiros - Pilhas (Aula 16T) - PILHA.C e PILHA2.C
- Ponteiros - Conceitos Complementares (Aula 17T)
- Ponteiros x Matrizes e Entradas e Saídas - Arquivos (Aula 18T) - PONT_8.C, PONT_9.C e PONT_6.C
- Operadores e Funções String (Aula 19T) - MAIUSC.C, MES.C e NUMEROLO.C
- Entradas e Saídas em Dispositivos (Aulas 20T) - IMP.C e ATOI.C
- Operações com Arquivo (Aula 21T) - REGS.C e REGS2.C
- Operações com Arquivo - Continuação (Aula 22T) - CAD1.C, ESCARQ.C, ESCARQ2.C e LESEQ.C
- Operações com Arquivo - Continuação (Aula 23T) - TYPE.C e TYPE2.C
- Operações com Arquivo - Conceitos de Chaves (Aula 23T)
- Teoria - Integrações Clipper e Btrieve (Aulas 24T)
- Montagem de Cadastro (Aula 25T)
- Integrações (Aula 26T)

Marcas Registradas citadas nesta apostila

Os produtos citados anteriormente, pertencem aos seus respectivos fabricantes,

a saber:

- Turbo C, Turbo Pascal, Borland C++, DBE - Borland Internacional Co.

- Visual BASIC, Visual C, ODBC - Microsoft Co.
- Fox- Fox Software Inc. (Microsoft)
- Clipper- Nantucket Co. (Computer Associates)
- Ingress- ASK (Computer Associates)
- Oracle- Oracle Inc.
- Btrieve- SoftCraft System, Inc. (Novell)
- SyBase- SyBase Co.
- Informix- Informix Co.
- dBase- Ashton Tate, Inc. (Borland)

Apêndice II - Lista de Exercícios Bimestrais

Lista de Exercícios de TAPD - 1º. Bimestre

- 1- Elaborar programa que construa os "n primeiros números da série de Fibonacci (0 1 1 2 3 5 8 ...).
- 2- Elaborar programa que imprima os números primos entre os números "a" e "b", fornecidos pelo usuário.
- 3- Crie programa que calcule a somatória de uma seqüência de números dados.
- 4- Traduzir o algoritmo a seguir para C, simular e dizer qual problema foi resolvido.
- declare i,lim,a como inteiras
- inicio
- iniciar i com 1, ler a, ler lim
- enquanto i < lim faça
- i recebe i + 1
- a recebe a - 1
- imprima a
- final

- 5- Certo usuário resolveu testar se um número inteiro qualquer "n" era divisível (resto = 0) pelos seguintes números: 3 e 5. Simule para 15 e 19.
- 6- Traduzir o algoritmo a seguir para C, simular e dizer qual problema foi resolvido.
- declare i,num como inteiras
- inicio
- solicitar e ler num
- iniciar i com 1
- enquanto num > 0 faça
- i recebe num * i
- num recebe num - 1
- final

- 7- Elabore um programa que totalize uma série de "n" números inteiros informados pelo teclado e imprima sua média.
- 8- Codifique o algoritmo abaixo em C.

```
declare i,j,k
leia k
se k > 100
        para i de 1 até 4 faça
                leia j
                se j > 7
                        imprima j
                senão
                        leia j
                se j = 10
                        imprima k
                senão
                imprima j
```

- Simule para: 200 3 5 6 8 e também para: 20 10
- 9- Codifique o algoritmo abaixo em C.

```
declare i,j,k,r
leia j,k
para i de j até k faça
        r = resto(i/2)
        if r = 0 então
                imprima i
```

Simule para: 3 12

- 10- Elabore programa que apresente a série abaixo apresentada:
- 0 4 7 10 13 16 ....
- O usuário deve informar o total de número a serem apresentados.

Elabore Algoritmos, Programas em C e Simulações.

Lista de Exercícios de TAPD- 2º. Bimestre

- 1- Ordene de forma crescente o conteúdo da matriz "m" com 10 elementos.
- 2- Dado o programa a seguir

```
main()
{
int i,j,l;
printf("Entre com 2 números: ");
scanf("%d%,&i); scanf("%d",&j);
if (i > j)
        l = i + j;
else
        l = i;
printf("%d",l);
}
```

O que faz este programa

Simule para i = 2, j = 3 e para j = 5 e i = 2

Caso trocássemos "scanf("%d",&i)" por "scanf(i)", o que aconteceria.

- 3- Traduzir o algoritmo a seguir para C, simular e dizer se este programa imprime sempre o menor número. Justifique ou corrija o programa, caso este esteja errado.

```
Leia a, b, c
Se a < b então
        d <-- a
Se a < c então
        d <-- a
Se b < c então
        d <-- b
imprima d
```

- 4- Elabore um programa que imprima valores de acordo com a tabela abaixo, sem usar o comando if.

```
i < 20 --> 3                i = 20 --> 4 5 6
i > 20 e i < 50 --> 5 6     i = 50 --> 6
i > 50 e i < 75 --> 7 8     i >= 75 --> 9
i = 90 --> 8
```

- Sugestão: Use switch

- 5- Dado o Programa

```
main()
{
int x,y;
for (x = 1, y = 2; x + y < 20; x = y++ + x + 1)
        printf("%d",x+y);
}
```

- a- Simule a execução deste programa, o que será impresso

- b- O que ocorreria caso fossem feitas as alterações abaixo,

```
for (x = 1, y = 2; x + y < 20; x = ++y + x + 1)
        printf("%d",x+y);
        y = 2;
```

- Simule.
- c- Rescreva este programa para que o usuário possa decidir, a cada impressão, se deseja ou não prosseguir a execução.

- 6- Traduza para a linguagem C

```
principal
declarar i=0,j=1,m,k como inteiras
        imprima i,j
para k de 1 até 20 faça
        m = i + j
        imprima m
        i = j
        j = m
```

- Simule

7- Dado o Programa a seguir:

```
main()
{
int y,a;
printf("Digite um valor: ");
scanf("%d",&a);
for(y=a;y<=100;y++);
        printf("%d",y);
for(y=100;y>0;y--) {
        printf("%d\n",y);
a=y+5;
}
printf("%d",a);
}
```

Simule, o que será impresso?

O programador cometeu um erro lógico. Identifique-o, corrigindo o programa.

Rescreva o programa, sem usar o comando "for".

- 8- Traduza para a linguagem C

```
principal
declarar i,j,k,t como inteiras
a como matriz de inteiros com 11 elementos
leia i,j
para k de i até j faça
        imprima linha em branco
para t de 0 até 10 faça
        a[t] = k * t
imprima k,t,a[t]
```

- O que faz este programa?

- 9- Dado o algoritmo a seguir

```
Programa X
i,n,j  <-- Inteiro
A <-- Matriz de Inteiros
Limpar a Tela
Leia n (Limite da Série)
j <-- 1
Para i que varia de 1 até n faça
        Se resto (i/4) = 0 então
                A[j] <-- i
                j <-- j + 1
Para i que varia de 1 até j faça
        Imprima A[j]
```

1. O que faz este programa .

2. Traduzir para C.
3. Simulação. Considerar n=17.

10- Calcule e apresente os "n" primeiros números primos solicitados pelo usuário.

Elabore Algoritmos, Programas em C e Simulações.

Lista de Exercícios de TAPD - 3º. Bimestre

- 1- Elabore programa que atribua um valor constante em uma variável "b" para uma variável "a", ambas inteiras, obrigatoriamente através de ponteiros.
- 2- Crie uma função que acumule em uma variável "x", passada como parâmetro pelo programa principal (seu endereço naturalmente) um valor digitado pelo operador na própria função.
- 3- Crie programa que acumule em uma variável "x" passada do programa principal, recursivamente, 5 vezes o número 5.
- 4- Elabore função que troque os valores das variáveis "a" e "b" passadas pelo programa principal.
- 5- Elabore programa que armazene números em uma pilha e recupere-os a ordem do usuário.
- 6- Considere o programa abaixo:

```
main()
char c,*pc,d;
{
c = 'A';
*pc = c;
d = &pc;
printf("%c",d);
}
```

O programador deseja transferir o valor da variável "c" para "d" usando ponteiros.

O que ocorre quando executado o programa acima, simule.

Programa correto, se necessário.

Comente a utilização do ponteiro neste programa, na versão acima.

- 7- Considere o programa abaixo:

```
main()
{
char a = '1', b = '9';
Troca(&a,&b);
printf("%c e %c",a,b);
}
```

- Escrever "Troca()", de forma que sejam impressos "9 e 1"
- 8- Considere o programa a seguir:

```
main()
int i,k,*pi,*pk;
 {
i = 2; k = 0;
puts("Qual será o valor de k? ");
*pk = i;
pk = &k;
printf("%d",k);

}
```

Simule a execução deste programa

Qual é o valor de k após a execução. Justifique.

Se trocarmos "*pk = i" por "pk = &i" o que ocorreria.

Caso seja necessário, corrija este programa para que no seu final "k" ter o valor de "i".

- 9- Sabendo que o programa principal lê um número, escreva função que mostre na tela a tabuada deste número digitado e retorne ao programa principal o valor da somatória das parcelas da tabuada.

- 10- Monte programa que acesse números inteiros previamente armazenados em uma pilha, porém propositadamente cause um estouro inferior da pilha. Qual será o resultado apresentado?

Elabore Algoritmos, Programas em C e Simulações.

Lista de Exercícios de TAPD - 4º. Bimestre

- 1- Monte programa que transfira os dados do arquivo DADO.DAT, não necessariamente existente, para o arquivo DADO.BAK.
- 2- Elabore programa que crie um arquivo chamado DADO.DAT e escreva um registro neste arquivo.
- 3- Dado um Nome qualquer, inverta-o.
- 4- Crie programa que exiba os dados do arquivo DADO.DAT, caso este arquivo exista. Se não existir envie mensagem "Arquivo Inexistente" ao operador.
- 5- Crie programa que exiba os dados do arquivo DADO.DAT, caso este arquivo exista. Se não existir envie mensagem "Arquivo Inexistente" ao operador.
- 6- Elabore Programa que crie e armazene registros em arquivo a ser acessado randômicamente.
- 7- Crie Programa que altere registros criados previamente.
- 8- Elabore Programa que liste em tela registros criados previamente.
- 9- Crie Lista de Registros usando ponteiros como apontadores.
- 10- Crie Registros ordenados de forma alfabética com ponteiros.

Elabore Algoritmos, Programas em C e Simulações.

Apêndice III - Respostas de Alguns Exercícios Propostos

- Aula 1L - Exercício 1

```
main()
{
printf("Hello, world!\n");
}
```

- Aula 7L - Exercício 6

```
main()
{
char ac[80];
int format, n;
char *acpoint;
for(n = 1; n < 8; n++){
        format = n;
        dostime(ac, format);
        printf("Acessando a funcao 'dostime' no formato,;
                        %d que representa: ",format);
        puts(ac);
        }
}
```

- Aula 8L - Exercício 6

```
main()
{
char c;
printf("Digite 1, 2 ou 3 ou outro caracter:");
c = getch();
```

```
switch © {
        case '1':
                puts("\nVoce Escolheu 1");
        case '2':
                puts("\nVoce Escolheu 2");
                break;
        case '3':
                puts("\nVoce Escolheu 3");
                break;
        default:
                puts("\nVoce Escolheu algo diferente de 1,2 ou 3");
        }
puts("Fim!");
}
```

- Aula 9L - Exercício 3

```
main()
{
char ac[80];
int j,i,ini,fim,n,nao;
double r;
cls();
printf("Digite extremo inferior: "); scanf("%d",&ini);
printf("\nDigite extremo superior: ");scanf("%d",&fim);
dostime(ac, 2);
puts(ac);
for(i=ini;i<=fim;i++) {
        nao = 1;
        if (i % 2 == 0)
                nao = 0;
        else {
                j = 3;
                r = i;
                r = sqrt( r );
                while (j<=r) {
                        if (i % j == 0) {
                                        nao = 0;
                                        break;
         }
                        j =j + 2;
     }
        }
        if (nao || i == 2)
                printf("%d ",i);
        }
printf("\n");
dostime(ac, 2);
puts(ac);
}
```

- Aula 23L - Exercício 2

```
struct date {
int day;
int month;
char mon_name[12];
};
main()
{
static struct date months[3] = { {5,1,"JANEIRO"}, {4,2,"FEVEREIRO"},
                {10,3,"MARCO"} };
int a;
struct date *p_months;
p_months=&months[1];    /* observe que o 1o. esta' armazenado     */
                        /* na posicao 0 */
                        /* da matriz, sempre usada em C. */
                         Portanto Fevereiro sera' o 2o. mes */
printf("Nome do Mes No. %d e': %s.",;
p_months->month, p_months->mon_name);
}
```

- Aula 24L - Exercício 2

```
main()
{
FILE *fo;
char c, dest[25];
int i;
char *cp;
double d;
printf("Arquivo a ser Criado: ");
gets(dest);
if((fo = fopen(dest,"w")) == NULL) {
        printf("Nao posso criar %s\n",dest);
        exit();
        }
c = 'A';
i = 999;
cp = "String criada para exemplo.";
d = 123.456;
```

```
fprintf(fo,"%c %d %s %lf\n", c, i, cp, d);
fclose(fo);
puts("Resposta Correta (no arquivo):A 999 String criada ;
                                para exemplo. 123.456000");
}
```

- Aula 24L - Exercício 3

```
main()
{
FILE *ler;
int i;
char x;
char name[50];
double d;
if((ler = fopen("TESTE.TST","r")) == NULL) {
        puts("Nao Posso Abrir TESTE.TST");
        exit();
        }
fscanf(ler,"%c %d %s %lf", &x, &i, name, &d);
printf("%c %d %s %lf", x, i, name, d);
}
```

- Aula 24L - Exercício 4

```
main()
{
int ret;
FILE *fi, *fo;
char c, source[50], dest[50];
printf("Nome do Arquivo: ");
gets(source);
fi = fopen(source, "w+");
ret = fputs("abcdefghijklmnopqrstuvwxyz", fi);
fclose(fi);
puts("Feito");
}
```

- Aula 24L - Exercício 5

```
/* programa leitor de arquivo, semelhante ao type do DOS */
#define MAXLEN 80
main()
{
FILE *fi;
char source[25], line[MAXLEN], *p;
printf("Nome do Arquivo: ");
gets(source);
if((fi = fopen(source, "r")) == NULL) {
        printf("\tNao posso abrir %s\n",source);
        exit();
        }
p = line;
while(p != NULL) {
        p = fgets(line,MAXLEN,fi);

        printf(line);
        }
fclose(fi);
puts("\n{FEITO}");
}
```

- Lista 1 - Exerc. 1 (Fibonnaci)

```
main()
{
int n,i,j,l,p;
printf("Digite um numero: ");
scanf("%d",&n);
i=0;
j=1;
printf("%d %d ",i,j);
for(l=1;l<=n-2;l++) {
  p=i+j;
  printf("%d ",p);
  i=j;
  j=p;
  }
}
```

- Lista 3 - Exerc. 2 (Passagem de Valores e Variáveis)

```
main()
{
int x;
soma(&x)
```

```
printf("\nO valor digitado foi %d\n",x);
}


soma(z)
int *z;
{
int y;
printf("Digite um Valor: ");
scanf("%d",&y);
*z=y;
```