LISTA DE EXERCÍCIOS 1

Exercício 1 No programa abaixo assinale quais são as variáveis locais, variáveis globais e os parâmetros formais.

```
#include <stdio.h>

int a = 100;
int maior (int x, int y);

void main (void)
{
        int b;
        scanf("%d", &b);
        printf("%d", maior(b,t));
}

int maior (int x, int y)
{
        int z;
        if(x > y) z = x;
        else z = y;
        return z;
}
```

Variáveis Locais: _____
Variáveis Globais: _____
Parâmetros Formais: _____

Exercício 2 Escreva um programa que calcule a área de um círculo a partir do valor do raio fornecido pelo usuário. A área do círculo é dada pela fórmula:

$$A = \pi \cdot R^2$$

Exercício 3 Escreva um programa que calcule a média anual de um aluno e imprima se o aluno está aprovado ou não. O programa deverá solicitar as 4 notas parciais ao usuário, calcular a média e, se essa for maior ou igual a 7 deverá imprimir "aprovado". Caso contrário deverá imprimir "reprovado".

Exercício 4 Escreva um programa que calcule as raízes de uma equação do segundo grau. O programa deverá ler as constantes `a`, `b` e `c` do teclado, certificar-se que $b^2 \geq 4ac$ e, então imprimir os possíveis valores para as raízes x_1 e x_2. Para calcular raiz quadrada pode-se usar a função `double sqrt (double)` presente na biblioteca `math.h`.

$$x = \frac{-b \pm \sqrt{b^2 - 4 \cdot a \cdot c}}{2 \cdot a}$$

Exercício 5 Escreva um programa que imprime na tela os dez primeiros números primos.

Exercício 6 Escreva um programa que imprime os N primeiros números da série de Fibonacci. O valor de N deverá ser solicitado ao usuário e lido do teclado. A série de Fibonacci é dada pela fórmula $n_i = n_{i-1} + n_{i-2}$ para $i \geq 2$. Para $i < 2$ ➔ $n = 1$ ($n_0 = n_1 = 1$). Ex: 1, 1, 2, 3, 5, 8, 13, ...

Exercício 7 Descreva o resultado do programa abaixo.

```
#include <stdio.h>

void main (void)
{
    int i;

    for(i=5;i>0;i--)
        printf("%d ", i);

    while(i<20)
    {
        printf("%d ", i++);
        if(i == 3) break;
    }

    do{
        if((i % 2) != 0) continue;
        printf("%d ", i);
    }while(++i < 8);
}
```

Exercício 8 Escreva um programa que faça a conversão da temperatura de Farenheit para Celsius e vice-versa. O programa deverá oferecer as seguintes opções:

```
Escolha a opção:
[1] - Farenheit → Celsius
[2] - Celsius → Farenheit
```

$$C = (F - 32) \cdot \frac{5}{9} \quad \text{e} \quad F = \left(C \cdot \frac{9}{5} \right) + 32$$

Exercício 9 Escreva um programa que imprima na tela a tabuada de um determinado número fornecido pelo usuário. O resultado deve ser apresentado como no exemplo abaixo:

```
n: 6

6 x 1 = 6
6 x 2 = 12
6 x 3 = 18
6 x 4 = 24
...
```

Exercício 10 Escreva um programa que calcula o fatorial de um determinado número fornecido pelo usuário.

Exercício 11 Escreva um programa que leia uma frase digitada no teclado e imprima o número de palavras digitadas. O programa deverá efetuar um loop para ler os caracteres até que um ENTER (13 decimal) seja pressionado.

Exercício 12 Escreva um programa que imprime na tela todos os caracteres ímpares no intervalo fechado [A, B]. O programa deverá solicitar os números A e B do usuário, verificar se A é maior que B e, então imprimir todos os números ímpares contidos no intervalo.

Exercício 13 Escreva um programa que calcula o valor da função exponencial de um determinado número fornecido pelo usuário utilizando a série de MacLaurin. Para calcular x elevado a y pode-se usar a função `double pow (double x, double y)` presente na biblioteca `math.h`.

$$e^x = \sum_0^{\infty} \frac{1}{n!} \cdot x^n \quad \text{ou} \quad e^x = 1 + x + \frac{1}{2} \cdot x^2 + \frac{1}{6} \cdot x^3 + \ldots$$

Exercício 14 Escreva um programa que calcula o valor do coseno de um determinado número fornecido pelo usuário utilizando a série de MacLaurin. Para calcular x elevado a y pode-se usar a função `double pow (double x, double y)` presente na biblioteca `math.h`.

$$\cos(x) = \sum_0^{\infty} \frac{(-1)^n}{(2 \cdot n)!} \cdot x^{2 \cdot n} \quad \text{ou} \quad \cos(x) = 1 - \frac{1}{2} \cdot x^2 + \frac{1}{24} \cdot x^4 - \frac{1}{720} \cdot x^6 + \ldots$$

Exercício 15 Escreva um programa que leia caracteres do teclado até que um ENTER (13 decimal) seja pressionado. Então o programa deverá imprimir um relatório com o número total de caracteres digitados, número total de vogais, número total de consoantes e o número total de caracteres numéricos (0 a 9).

```
#include<conio.h>
#include<stdio.h>
#include<math.h>
void main ()
{
 int x=0,i=0,y=1,z=0,a=0;
 printf("Digite a quantidade de numeros da serie de Fibonacci para ser
visualizada na tela\n");
 scanf("%d",&x);
 if(x!=0)
 {
 printf("%d,",y);
 for (i=2;i<=x;i++)
 {
 z=a+y;
 a=y;
 y=z;
 printf("%d,",z);
 }
 }
 else
 {
 z=1;
 }
 getch();
}
```

```
#include<conio.h>
#include<stdio.h>
#include<math.h>
void main()
{
 int i,p;
 float x,y,m;
 printf("Calculo de media");
 while (p!=0)
 {
 i=1;
 y=x=0;
 printf("\nFavor entrar com as notas bimestrais do aluno\n");
 for (i>0;i<5;i++)
 {
 scanf("%f",&x);
 y=x+y;
 m=y/4;
 }
 if (m<7)
 {
 printf("\nAluno reprovado, sua media eh de %.2f",m);
 printf("\nPara sair do programa digite 0, para calcular outra media digite
1\n");
 scanf("%d",&p);
 clrscr();
 }
 else
 {
 printf("Aluno aprovado, sua media eh de %.2f",m);
 printf("\nPara sair do programa digite 0, para calcular outra media digite
1\n");
 scanf("%d",&p);
 clrscr();
 }
 }
}
```

```
#include<math.h>
#include<stdio.h>
#include<conio.h>
void main ()
{
 int x,i,y;
 while(y!=0)
 {
 printf("Digite o numero para verificar sua tabuada\n");
 scanf("%d",&x);
 printf("A tabuada do numero %d eh:\n\n",x);
 for(i=0;i<=10;i++)
 {
 printf(" %d x %d = %d\n",x,i,x*i);
 }
 printf("\n Para verificar outra tabuada digite 1, caso contrario digite 0\n");
 scanf("%d",&y);
 clrscr();
 }
}
```

LISTA DE EXERCÍCIOS 2

Exercício 1 Escreva um programa que calcule a média anual de um aluno e imprima se o aluno está aprovado ou não. O programa deverá solicitar as 4 notas parciais ao usuário, armazenar os valores em um vetor, calcular a média e, se essa for maior ou igual a 7 deverá imprimir "aprovado". Caso contrário deverá imprimir "reprovado".

Exercício 2 Escreva um programa que leia uma frase digitada no teclado, armazene em uma string e imprima o número total de palavras digitadas. O programa poderá utilizar a função `gets` da biblioteca `stdio.h` para ler a string de entrada.

Exercício 3 Escreva um programa que lê 20 inteiros do teclado, armazena esses valores em um vetor e verifica se há valores repetidos nesse vetor. O resultado deverá ser impresso na tela e se houver itens repetidos deve-se mostrar em que posições eles se encontram.

Exercício 4 Escreva um programa que leia uma frase do teclado, armazene em um vetor e imprima um relatório com o número total de caracteres digitados, número total de vogais, número total de consoantes e o número total de caracteres numéricos (0 a 9).

Exercício 5 Escreva um programa que calcula o determinante de uma matriz 3x3. O programa deverá solicitar ao usuário item por item da matriz, calcular o determinante e imprimir na tela a matriz fornecida pelo usuário e o valor do determinante.

Exercício 6 Escreva um programa que recebe uma string do usuário, organiza os caracteres em ordem alfabética (Ex: "UNICENP" → "CEINNPU") e, então imprime na tela.

Exercício 7 Considere que você não possui disponível a biblioteca de funções para manipulação de strings. Escreva uma função que retorne o número total de caracteres de uma string.

```
int strlen (char str[]);
```

Exercício 8 Escreva uma função que concatena strings. A string str2 deve ser copiada no fim de str1.

```
void strcat (char str1[], char str2[]);
```

Exercício 9 Escreva um programa que possibilite ao professor armazenar a nota de 3 turmas com 30 alunos cada uma. O programa deverá solicitar as notas de cada aluno e imprimir um relatório contendo a nota mais baixa, a nota mais alta e a média de cada um das 3 turmas.

Exercício 10 Escreva um programa que leia do teclado 10 strings de entrada, armazene-as em uma matriz de strings e imprima um relatório contendo o número de caracteres da menor string, o número de caracteres da maior string e a média do número de caracteres de todas as strings.

```
#include<conio.h>
#include<math.h>
#include<stdio.h>
void main ()
{
  int x=0,i,y=0,z=0,n=0,c=0;
  char frase[999];
  printf("Digite a frase\n");
  gets(frase);
  y=strlen(frase);
   for(i=0;i<y;i++)
   {
    if ((frase[i]=='!') || (frase[i]=='?') || (frase[i]==',') || (frase[i]=='.')
|| (cont[x]==':') || (cont[x]==';'))
    c=c+1;
    if ((frase[i]=='a') || (frase[i]=='A'))
    x=x+1;
    if ((frase[i]=='e') || (frase[i]=='E'))
    x=x+1;
    if ((frase[i]=='i') || (frase[i]=='I'))
    x=x+1;
    if ((frase[i]=='o') || (frase[i]=='O'))
    x=x+1;
    if ((frase[i]=='u') || (frase[i]=='U'))
    x=x+1;
    if (frase[i]==' ')
    z=z+1;
    if (frase[i]=='0')
    n=n+1;
    if (frase[i]=='1')
    n=n+1;
    if (frase[i]=='2')
    n=n+1;
    if (frase[i]=='3')
    n=n+1;
    if (frase[i]=='4')
    n=n+1;
    if (frase[i]=='5')
    n=n+1;;
    if (frase[i]=='6')
    n=n+1;
    if (frase[i]=='7')
    n=n+1;
    if (frase[i]=='8')
    n=n+1;
    if (frase[i]=='9')
    n=n+1;
    }
   printf("\n A quantidade de caracteres digitados eh: %d\n A quantidade de
 vogais da frase eh: %d\n A quantidade de consoantes eh: %d\n A quantidade de
 numeros eh: %d",y,x,y-x-z-n-c,n);
   getch();
   }
```

LISTA DE EXERCÍCIOS 3

Exercício 1 Escreva um programa que lê um inteiro do teclado e imprime na tela o valor da própria variável e o endereço de memória que ela ocupa.

Exercício 2 Escreva um programa que imprime na tela o valor da variável do tipo ponto-flutuante que está armazenada no endereço 0x100000. Implemente no Borland C++ Builder e explique o resultado obtido na prática.

Exercício 3 Escreva um programa que declara uma variável global do tipo inteiro, uma variável local do tipo inteiro e imprime na tela a distância que essas variáveis estão uma da outra na memória, isto é, a diferença entre o endereço de cada uma delas.

Exercício 4 Escreva um programa que declara uma string local inicializada com a frase "CENTRO UNIVERSITARIO POSITIVO". Declare um ponteiro, do tipo char, que receba o endereço do décimo elemento da string criada. Com auxílio do ponteiro, escreva na posição apontada o caracter '*' e imprima na tela a string utilizada.

Exercício 5 Explique o comportamento do seguinte programa.

```
#include <stdio.h>
#include <conio.h>

void main (void)
{
    int x;
    int    * p1;
    float  * p2;
    double * p3;

    x = 100000;
    p1 = &x;
    p2 = &x;
    p3 = &x;

    printf("%d - %d - %d\n", p1, p2, p3);
    printf("%d - %f - %lf\n", *p1, *p2, *p3);
}
```

Exercício 6 Escreva um programa que declara dois vetores de inteiros e, com auxílio de ponteiros, realiza a troca dos valores de cada elemento dos vetores.

Exercício 7 Escreva uma função que receba como argumentos dois inteiros e, com auxílio de ponteiros, faça a troca dos dois valores sem que haja perda de informação após a execução do programa retornar da função.

Exercício 8 Escreva um programa que declare um vetor de caracteres para armazenar uma string digitada pelo usuário. Com auxílio de ponteiros, verifique o número de caracteres digitados pelo usuário, lembrando que o sinalizador do final da string é o caracter '\0'.

Exercício 9 Escreva um programa que declare um vetor de caracteres para armazenar uma string digitada pelo usuário. Com auxílio de ponteiros, imprima a string digitada pelo usuário de trás para frente.

Exercício 10 Escreva um programa que declare um vetor de caracteres para armazenar uma string digitada pelo usuário. Com auxílio de ponteiros, imprima a string digitada pelo usuário criptografando da seguinte maneira. Ao encontrar o caracter 'Z' substituir por 'P' e vice-versa. O mesmo deve ocorrer para os pares 'E' e 'O', 'N' e 'L', 'I' e 'A', 'T' e 'R'.

Exercício 1 Escreva um programa que lê um inteiro do teclado e
imprime na tela o valor da própria variável e o endereço de memória que ela ocupa.

```
#include <math.h>
#include <stdio.h>
#include <conio.h>
void main ()
{
int a,*p;
printf("digite o valor de a\n");
scanf(" %d",&a);
p=&a;
printf(" o valor de a eh: %d e sua memoria eh:
%d",a, p);
getch();
}
```

Exercício 2 Escreva um programa que imprime na tela o valor da variável do tipo ponto-flutuante que está
armazenada no endereço 0x100000. Implemente no Borland C++ Builder e explique o resultado obtido na
prática.

```
#include <math.h>
#include <stdio.h>
#include <conio.h>
void main ()
{
float a;
float *p=1245064;
a=*p;
printf("%f",a);
getch();
}
```

Exercício 3 Escreva um programa que declara uma variável global
do tipo inteiro, uma variável local do tipo inteiro e imprime na
tela a distância que essas variáveis estão uma da outra na memória,
isto é, a diferença entre o endereço de cada uma delas.

```
#include <math.h>
#include <conio.h>
#include <stdio.h>
int x,*p;
void main ()
{
int y,*q;
int z;
printf("Digite o 1 ponteiro\n");
scanf("%d",&x);
printf("Digite o 2 ponteiro\n");
scanf("%d",&y);
z=&x-&y;
printf ("A distancia entre as memorias eh de %d
(%d - %d)", 4*z,&x,&y);
getch();
}
```

Exercício 4 Escreva um programa que declara uma string local inicializada com a frase "CENTRO UNIVERSITARIO POSITIVO". Declare um ponteiro, do tipo char, que receba o endereço do décimo elemento da string criada. Com auxílio do ponteiro, escreva na posição apontada o caracter '*' e imprima na tela a string utilizada.

```
void main ()
{
 char *Ft;
 char t;
 char F[30]="CENTRO UNIVERSITARIO
POSITIVO";
 Ft=&F[9];
 t='*';
 *Ft=t;
 printf("a string utilizada eh %s\n",F);
 printf("a memoria gravada foi %d\n",Ft);
 printf("a nova string eh %s\n",F);
 getch ();
 }
```

Exercício 5 Explique o comportamento do seguinte programa.
R: qdo vc declara um ponteiro do tipo inteiro, ele só vai ler
variáveis do tipo inteiro...e vice-versa....neste caso todos
os ponteiros leêm os msm endereços de memória...já na hora de
fazer a impressão do conteúdo, apenas o ponteiro 1 irá funcionar
pois trata-se de uma variável do tipo inteiro.

```
#include <stdio.h>
#include <conio.h>

void main(void)
{
int x;
int *p1;
float *p2;
double *p3;
x=100000;
p1=&x;
p2=&x;
p3=&x;
printf ("%d - %d - %d\n", p1,p2,p3);
printf ("%d - %f - %lf\n", *p1,*p2,*p3);
getch ();
}
```

Exercício 6 Escreva um programa que declara dois vetores de inteiros e,
com auxílio de ponteiros, realiza a troca dos valores de cada elemento dos vetores.

```
#include <stdio.h>
#include <math.h>
#include <conio.h>
#define tamanho 5
void troca_valor();
main()
{
        int matriz_1[tamanho], *pti_1;
        int matriz_2[tamanho], *pti_2;
        int indice;
        pti_1=matriz_1;
        pti_2=matriz_2;
        for(indice=0; indice < tamanho;
indice++)
        {
                printf("valor para matriz
1 \n", indice);

scanf("%d",&matriz_1[indice]);
        }
        for(indice=0; indice < tamanho;
indice++)
        {
                printf("valor para matriz
2 \n", indice);

scanf("%d",&matriz_2[indice]);
        }
        for(indice=0; indice < tamanho;
indice++)
        {

troca_valor(&matriz_1[indice],
&matriz_2[indice]);
        }
}
void troca_valor(campo1, campo2)
{
        int *campo1, *campo2;
        {
                int temporario;
                temporario=*campo1;
                *campo1=*campo2;
                *campo2=temporario;
                getch();
        }
}
```

Exercício 8 Escreva um programa que declare um vetor de
caracteres para armazenar uma string digitada pelo usuário.
Com auxílio de ponteiros, verifique o número de caracteres
digitados pelo usuário, lembrando que o sinalizador do final
da string é o caracter '\0'.

```
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>

void main()
{
        char frase[80];
        int *p;
        int i;
        p = malloc(2);

        printf("digite uma frase: ");
        gets(frase);

        for(i=0; i<=80; i++)
        {
                if(frase[i]=='\0')
                {
                        *p = i;
                        break;
                }
        }
        printf("%d", *p);
        getch();
}
```

Exercicio 9 Escreva um programa que declare um vetor de caracteres para
armazenar uma string digitada pelo usuário. Com auxílio de ponteiros,
imprima a string digitada pelo usuário de trás para frente.

```
#include<conio.h>
#include<stdio.h>
#include<math.h>

void main ()
{
char A[99], *t;
int i;
printf("digite uma frase:\n");
gets(A);
for(i=strlen(A)-1;i>=0;i--)
{
t=A[i];
printf ("%c",t);
}
getch();
}
```

Exercício 10 Escreva um programa que declare um vetor de caracteres
para armazenar uma string digitada pelo usuário. Com auxílio de
ponteiros, imprima a string digitada pelo usuário criptografando da
seguinte maneira. Ao encontrar o caracter 'Z' substituir por 'P' e
vice-versa. O mesmo deve ocorrer para os pares 'E' e 'O', 'N' e 'L',
'I' e 'A', 'T' e 'R'.

```
#include <stdio.h>
#include <conio.h>

void main(void)
{
char TXT[80], c, *p;
int i;
printf("Usuario, entre com a string\n");
gets(TXT);
for (i=0;i<strlen(TXT);i++)
  {
  p=&TXT[i];
  switch (*p)
    {
    case 'Z': *p='P'; break;
    case 'P': *p='Z'; break;
    case 'E': *p='O'; break;
    case 'O': *p='E'; break;
    case 'N': *p='L'; break;
    case 'L': *p='N'; break;
    case 'I': *p='A'; break;
    case 'A': *p='I'; break;
    case 'T': *p='R'; break;
    case 'R': *p='T'; break;
    }
  }
printf("%s",TXT);
getch();
}
```

Exercicio 7 Escreva uma função que receba como argumentos dois inteiros e, com
auxílio de ponteiros,
faça a troca dos dois valores sem que haja perda de informação após a execução do
programa retornar da
função.

```
#include<stdio.h>
#include<math.h>
#include<conio.h>
void main()
{
int x,y,z,*p1,*p2;
printf("digite o primeiro inteiro\n");
scanf("%d",&x);
printf("digite o segundo inteiro\n");
scanf("%d",&y);
p1=&x;
p2=&y;
z=*p1;
x=*p2;
y=z;
printf("%d %d",x,y);
getch();
}
```

LISTA DE EXERCÍCIOS 4

Exercício 1 Escreva uma função que imprime na tela a tabuada de um determinado número fornecido como argumento. O protótipo da função e como deverá ser apresentado o resultado estão logo abaixo:

```
void imprime_tabuada (int i);
```

```
6 x 1 = 6
6 x 2 = 12
6 x 3 = 18
6 x 4 = 24
...
```

Exercício 2 Escreva uma função que retorna o maior valor entre dois inteiros.

```
int max (int a, int b);
```

Exercício 3 Escreva uma função que calcula a área de um círculo a partir do valor do raio fornecido como argumento. A declaração da função e a fórmula da área do círculo são:

```
float calcula_area (float raio);
```

$$A = \pi \cdot R^2$$

Exercício 4 Escreva um programa que calcula a área de um círculo utilizando a função `calcula_area` escrita no exercício anterior. O programa deverá solicitar ao usuário o valor do raio, chamar a função `calcula_area` e então imprimir o valor da área na tela.

Exercício 5 Escreva uma função que retorne a média dos valores contidos em uma matriz unidimensional. Como argumento deve-se passar a matriz e o número de elementos.

```
float media (float m[], int n);
```

Exercício 6 Descreva objetivamente o que o seguinte programa faz.

```
#include <stdio.h>

void swap (int a, int b);

void main (void)
{
    int a = 10, b = 20;

    printf("%d e %d", a, b);
    swap(a, b);
    printf("%d e %d", a, b);
}

void swap (int a, int b)
{
    int tmp;
    tmp = a;
    a = b;
    b = tmp;
}
```

Exercício 7 Escreva uma função que realize o swap entre duas variáveis `a` e `b`.

```
void swap (int * a, int * b);
```

Exercício 8 Escreva duas funções: uma que converta um valor de temperatura em Celsius para Farenheit, e outra que converta um valor de temperatura em Farenheit para Celsius. Seguem os protótipos das funções e as devidas fórmulas de conversão.

```
float Celsius_to_Farenheit (float x);

float Farenheit_to_Celsius (float x);
```

$$C = (F - 32) \cdot \frac{5}{9} \quad \text{e} \quad F = \left(C \cdot \frac{9}{5} \right) + 32$$

Exercício 9 Escreva uma função que calcula as raízes de uma equação do segundo grau. A função deverá ler os argumentos `a`, `b` e `c`, certificar-se que $b^2 \geq 4ac$ e, então colocar os possíveis valores para as raízes x_1 e x_2 nos argumentos `px1` e `px2`. Para calcular raiz quadrada pode-se usar a função `double sqrt (double)` presente na biblioteca `math.h`.

```
void raizes (float a, float b, float c, float * px1, float * px2);
```

$$x = \frac{-b \pm \sqrt{b^2 - 4 \cdot a \cdot c}}{2 \cdot a}$$

Exercício 10 Escreva uma função recursiva que calcula o fatorial de um inteiro passado como argumento da função. Pressupõe-se que:

$$n! = n \cdot (n-1)!$$

```
int fatorial (int n);
```

Exercício 11 Descreva objetivamente o que o seguinte programa faz.

```
#include <stdio.h>

void XYZ (void);

void main (void)
{
      printf("Digite uma frase: ");
      XYZ();
}

void XYZ (void)
{
      char c;
      if((c = getch()) != '?') XYZ();
      putchar(c);
}
```

LISTA 04

Exercício 1
Escreva uma função que imprime na tela a tabuada de um determinado número fornecido como argumento. O protótipo da função e como deverá ser apresentado o resultado estão logo abaixo:

```
#include <conio.h>
#include <stdio.h>
void imprime_tabuada (int i);
void main()
{
   int x;
   printf(" digite um valor \n");
   scanf("%d",&x);
   imprime_tabuada(x);
   getch();
}
void imprime_tabuada (int i)
{
   int a;
   for(a=1; a<=10; a++)
   {
   printf("%d x %d = %d \n", i,a,i*a);
   }
}
```

Exercício 2
Escreva uma função que retorna o maior valor entre dois inteiros.

```
int max (int a, int b);

#include <stdio.h>
#include <conio.h>

int maior_num (int a, int b);
void main (void)

{
    int a, b, num_maior;
    printf("digite os 2 numeros para
serem comparados");
    scanf("%d",&a);
    scanf("%d",&b);
    num_maior = maior_num(a,b);
    printf("O numero maior eh: %d",
num_maior);
    getch();
}
int maior_num (int a, int b)
{
    if (a>b) return a;
    return b;
}
```

Exercício 3
Escreva uma função que calcula a área de um círculo a partir do valor do raio fornecido como argumento. A declaração da função e a fórmula da área do círculo são:

```
float calcula_area (float raio);
{
     flaot a;
     a=3,14*raio*raio;
     return (a);
}
```

Exercício 4
Escreva um programa que calcula a área de um círculo utilizando a função calcula_area escrita no exercício anterior. O programa deverá solicitar ao usuário o valor do raio, chamar a função calcula_area e então imprimir o valor da área na tela.

```
#include <stdio.h>
#include <conio.h>
float calc_area (float raio);
void main (void)
{
    float x;
    printf("Entre com o raio");
    scanf("%f",&x);
    printf("A area eh: %f",
                calc_area(x));
    getch();
}
float calc_area (float raio)
{
    float a;
    a=3.14*raio*raio;
    return a;
}
```

Exercício 5
Escreva uma função que retorne a média dos valores contidos em uma matriz unidimensional.Como argumento deve-se passar a matriz e o número de elementos.

```
float media (float m[], int n);
{
    float media;
    while(i<n)
    {
       soma=m[i] + soma;
       i++;
    }
       media = soma/n;
       return (media);
}
```

Exercício 6
Descreva objetivamente o que o seguinte programa faz.

```
#include <stdio.h>
void swap (int a, int b);
void main (void)
{
     int a = 10, b = 20;
     printf("%d e %d", a, b);     //
retorna 10 e 20
     swap(a, b);
     printf("%d e %d", a, b);     //
retorna 10 e 20
}

void swap (int a, int b)          // o
programa nao faz nada
{
     int tmp;
     tmp = a;
     a = b;
     b = tmp;
}
```

Exercício 7
Escreva uma função que realize o swap
entre duas variáveis a e b.

```
void swap (int * a, int * b);
{
    int tmp;
    tmp = *a;
    *a=*b;
    *b=tmp;
}
```

Exercício 8
Escreva duas funções: uma que converta
um valor de temperatura em Celsius
para Farenheit, e outra que converta um
valor de temperatura em Farenheit
para Celsius. Seguem os protótipos das
funções e as devidas fórmulas de
conversão.

```
float Celsius_to_Farenheit (float x);
float Farenheit_to_Celsius (float x);

#include <conio.h>
#include <stdio.h>
#include <math.h>
float celsius_to_farenheit (float x);
float farenheit_to_celsius (float x);
void main()
{
          float temp, celsius, farenheit,
w, y;
          printf(" 1) digite a
temperatura em celsius: \n");
          scanf("%f",&w);
          printf(" 2) digite a
temperatura em farenheit: \n");
          scanf("%f",&y);
          celsius = farenheit_to_celsius
(y);
          farenheit =
celsius_to_farenheit (w);
          printf("\n Resp. 1) em
farenheit e : %.2f \n", farenheit);
          printf("\n Resp. 2) em celsius
e: %.2f \n", celsius);
          getch();
}
float celsius_to_farenheit (float c)
{
          float f;
          f=((c*(9.0/5.0))+32);
          return (f);
}
float farenheit_to_celsius (float f)
{
          float c;
          c=(((f-32)*5)/9.0);
          return (c);
}
```

LISTA DE EXERCÍCIOS 5

Exercício 1 Considere que uma empresa precisa armazenar os seguintes dados de um cliente:

- Nome completo com no máximo 50 caracteres.
- Renda mensal do cliente.
- Ano de nascimento.
- Possui ou não carro.

Defina uma estrutura para armazenar estes dados e escreva um programa que lê estes dados do teclado, armazena-os em uma variável e, então imprima os dados na tela.

Exercício 2 Considerando a mesma estrutura do exercício anterior, escreva um programa que lê os dados de 100 clientes e imprime um relatório contendo:

- Quantos clientes têm carro.
- Quantos clientes nasceram entre 1960 e 1980.
- Quantos clientes têm renda mensal acima da média.

Exercício 3 Escreva um programa que implementa uma estrutura com os seguintes campos:

- Nome do Usuário com no máximo 50 caracteres.
- Senha do Usuário com no máximo 10 caracteres.

Declare uma matriz de 10 elementos do tipo criado. Faça um loop solicitando que o usuário cadastre nome e senha e armazene essas informações na matriz declarada. Após cadastrar os dez registros, fique em loop solicitando nome e senha ao usuário até que esse digite um nome e senha válidos, isto é, nome e senha que estejam previamente cadastrados. A cada tentativa sem sucesso imprima "Acesso Negado".

Exercício 4 Considere a seguinte estrutura:

```
struct Produto
{
    char codigo[20];
    char descricao[20];
    float preco;
}
```

4.1. Escreva uma função que receba uma variável do tipo Produto como argumento e imprima na tela os valores dessa variável.

```
void imprime_produto (Produto p);
```

4.2. Escreva uma função que solicita ao usuário os dados referentes a uma variável do tipo Produto e retorne uma variável do tipo Produto.

```
Produto le_produto (void);
```

4.3. Escreva uma função que solicita ao usuário os dados referentes a uma variável do tipo Produto e preenche essas informações em uma variável referenciada por um ponteiro que é recebido como argumento da função.

```
void le_produto (Produto * p);
```

Exercício 5 Considere a mesma estrutura do exercício anterior e uma estrutura chamada Estoque como a que é mostrada logo abaixo. A estrutura contém um vetor do tipo Produto de 1000 itens e a variável numero_itens, que contém o número de itens cadastrados no estoque até o momento.

```
struct Estoque
{
      Produto prod[1000];
      int numero_itens;
}
```

5.1. Escreva uma função que cadastra um novo produto ao estoque. A função deverá receber um ponteiro para Estoque como argumento e solicitar ao usuário informações a respeito do novo produto a ser cadastrado.

```
void cadastra_produto (Estoque * p);
```

5.2. Escreva uma função que realiza a consulta de preço de um produto do estoque. A função deverá receber um ponteiro para Estoque como argumento, solicitar ao usuário a descrição do produto que se deseja consultar e, então imprimir o preço do produto.
Obs: para comparar duas strings pode-se utilizar a função `strcmp`.

```
void consulta_preco (Estoque * p);
```

5.3. Escreva uma função que realiza a atualização do preço de um produto do estoque. A função deverá receber um ponteiro para Estoque como argumento, solicitar ao usuário a descrição do produto, o novo preço e, então atualizar o preço do produto no estoque.

```
void atualiza_preco (Estoque * p);
```

5.4. Escreva um programa que oferece ao usuário as seguintes funcionalidades mostradas logo abaixo.

```
[ 1 ] - Cadastra Produto
[ 2 ] - Consulta Preço
[ 3 ] - Atualiza Preço
[ 4 ] - Sai do Programa
```

Requisitos:

- O programa deverá permanecer em execução até que a opção 4 seja escolhida.
- O programa deverá trabalhar com 2 estoques distintos, isto é, deverá possuir 2 variáveis globais do tipo Estoque de nomes Matriz e Filial.

  ```
  Estoque Matriz;
  Estoque Filial;
  ```

- Antes de efetuar qualquer operação de cadastro, consulta ou atualização de informações o programa deverá solicitar em qual estoque o usuário quer operar (Matriz ou Filial).

Exercício 1- Considere que uma empresa precisa armazenar os seguintes dados de um cliente:
- Nome completo com no máximo 50 caracteres.
- Renda mensal do cliente.
- Ano de nascimento.
- Possui ou não carro.

Defina uma estrutura para armazenar estes dados e escreva um programa que lê estes dados do teclado, armazena-os em uma variável e, então imprima os dados na tela.

```
#include <stdio.h>
#include <conio.h>

struct cliente

{
 char nome[51];
 float renda;
 int ano;
 char carro;
};

void main (void)

{
 cliente ficha;
 printf ("Digite seu nome");
 gets (ficha.nome);
 printf ("Digite sua renda");
 scanf ("%f",&ficha.renda);
 printf ("Digite o ano de nascimento")
 Scanf ("%d",&ficha.ano);
 printf ("Possui carro? S para sim e N para não.");
 ficha.carro=getchar();
 printf ("nome= %s", ficha.nome);
 printf ("renda= %.2f", ficha.renda);
 printf ("ano de nascimento= %d", ficha.ano);
 printf ("possui carro= %c", ficha.carro);
 getch();
}
```

Exercício 2- Considerando a mesma estrutura do exercício anterior, escreva um programa que lê os dados de 100 clientes e imprime um relatório contendo:
- Quantos clientes têm carro.
- Quantos clientes nasceram entre 1960 e 1980.
- Quantos clientes têm renda mensal acima da média.

```
#include <stdio.h>
#include <conio.h>

void main (void)

{
 cliente ficha [100];
 int i, c, a;
 for (i=0;i<100;i++)
 {
 printf ("Digite seu nome");
 gets (ficha[i].nome);
 printf ("Digite sua renda");
 scanf ("%f",&ficha[i].renda);
 printf ("Digite o ano de nascimento")
 Scanf ("%d",&ficha[i].ano);
 printf ("Possui carro? S para sim e N para não.");
 ficha[i].carro=getchar();
 crlscr();
 }
 for(i=0;i<100;i++)
 {
 if (ficha[i].carro='s')
 {
 c++;
 }
 if (ficha[i].ano>1960)&&(ficha[i].ano<1980)
 {
 a++;
 }
 }
}
```

Exercício 4
Considere a seguinte estrutura:

```
struct Produto
{
        char codigo[20];
        char descricao[20];
        float preco;
}
```

4.1. Escreva uma função que receba uma variável do tipo Produto como argumento e imprima na tela os valores dessa variável.

```
void imprime_produto (Produto p);

{
        printf("%s", p.codigo);
        printf("%s", p.descricao);
        printf("%s", p.preço);
}
```

4.2. Escreva uma função que solicita ao usuário os dados referentes a uma variável do tipo Produto e retorne uma variável do tipo Produto.

```
Produto le_produto (void);

{
        produto tmp;
        printf("digite o codiogo");
        gets(tmp.codigo);
        printf("digite a descrição");
        gets("tmp.descrição);
        printf("digite o preço");
        scanf("%f",&tmp.preço);
        return(tmp);
}
```

4.3. Escreva uma função que solicita ao usuário os dados referentes
a uma variável do tipo Produto e preenche essas informações em uma
variável referenciada por um ponteiro que é recebido como argumento
da função.

```
void le_produto (Produto * p);

{
        printf("digite o codigo");
        gets(p->codigo);
        printf("digite a descrição");
        gets(p->descrição);
        printf("digite o preço");
        scanf("%f",&p->preço);
        return(tmp);
}
```

Exercício 5 Considere a mesma estrutura do exercício anterior e uma
estrutura chamada Estoque como a que é mostrada logo abaixo. A
estrutura contém um vetor do tipo Produto de 1000 itens e a variável
numero_itens, que contém o número de itens cadastrados no estoque até
o momento.

```
struct Estoque
{
        Produto prod[1000];
        int numero_itens;
}

struct estoque
{
        produto prod[1000]
        int numero_itens;
}
void cadastra_produto (estoque *p)
{
        int i=p->numero_itens;
        printf("codigo");
        gets(p->prod[i].codigo);
        printf("descrição");
        gets(p->prod[i].descirçao);
        printf("preço");
        scanf("%f",&p->prod[i].preço);
        p->numero_itens++;
}
```

Exercício 4 e 5

```
// CONSTANTS ////////////////////////////////////

#include<stdio.h>
#include<conio.h>
#include<string.h>

// STRUCTS //////////////////////////////////////////

struct produto {
  char codigo[20];
  char descricao[20];
  float preco;
};

struct estoque {
  produto prod[1000];
  int numero_itens;
};

estoque matriz;
estoque filial;

// PROTOTYPES ////////////////////////////////////

void imprime_produto(produto p);
produto le_produto(void);
void cadastra_produto(estoque *p);
estoque *almoxarifado(void);
void consulta_preco(estoque *p);
void atualiza_preco(estoque *p);

// FUNCTIONS ////////////////////////////////////

void main(void){
 int menu;
 estoque *almox;
 while(menu!=4) {
  printf("+----------------------------------------+\n");
  printf("|              P R O D U T O S           |\n");
  printf("|                                        |\n");
  printf("| [ 1 ] Cadastra Produto                 |\n");
  printf("| [ 2 ] Consulta Preco                   |\n");
  printf("| [ 3 ] Atualiza Preco                   |\n");
  printf("| [ 4 ] Sair do Programa                 |\n");
  printf("|                                        |\n");
  printf("+----------------------------------------
+\n\n");
  printf("Digite a opcao desejada: ");
  scanf("%d", &menu);
  switch (menu) {
   case 1: clrscr();
    almox=almoxarifado();
    cadastra_produto(almox);
    getch();
    break;
   case 2: clrscr();
    almox=almoxarifado();
    consulta_preco(almox);
    getch();
    break;
   case 3: clrscr();
    almox=almoxarifado();
    atualiza_preco(almox);
    getch();
    break;
   case 4: printf("\nObrigado por utilizar nossos servicos.
Volte sempre!!!");
    getch();
  }
  clrscr();
 }
}

void imprime_produto(produto p) {
 printf("\n\nCodigo: %s",p.codigo);
 printf("\nDescricao: %s",p.descricao);
 printf("\nPreco: %.2f",p.preco);
}

produto le_produto(void) {
 produto tmp;
 printf("\n\nDigite o codigo: ");
 scanf("%s", tmp.codigo);
 printf("\nDigite descricao: ");
 scanf("%s", tmp.descricao);
 printf("\nDigite o preco: ");
 scanf("%f",&tmp.preco);
 return(tmp);
}

void cadastra_produto(estoque *p) {
 p->prod[p->numero_itens]=le_produto();
 p->numero_itens++;
}
```

```
estoque *almoxarifado(void) {
 int menu2;
 estoque *tmp,
 printf("+----------------------------------------------------+\n");
 printf("|    SELECAO DE ESTOQUE                              |\n");
 printf("|                                                    |\n");
 printf("| [ 1 ] Estoque Matriz                               |\n");
 printf("| [ 2 ] Estoque Filial                               |\n");
 printf("|                                                    |\n");
 printf("+----------------------------------------------------
+\n\n");
 printf("Digite a opcao desejada: ");
 scanf("%d",&menu2);
 switch (menu2) {
   case 1: tmp=&matriz;
     break;
   case 2: tmp=&filial;
 }
 return(tmp);
}
void consulta_preco(estoque *p) {
 int i;
 produto tmp;
 clrscr;
 printf("Entre com a descricao do produto: ");
 scanf("%s",&tmp.descricao);
 for(i=0;i<=p->numero_itens;i++) {
  if(!strcmp(tmp.descricao,p->prod[i].descricao)) {
    printf("\n\nO preco do produto %s e R$ %.2f .",p-
>prod[i].descricao,p->prod[i].preco);
    i=0;
    break;
  }
 }
 if(i!=0) printf("\nProduto nao cadastrado");
}
void atualiza_preco(estoque *p) {
 int i;
 produto tmp;
 printf("Entre com a descricao do produto:");
 scanf("%s",&tmp.descricao);
 for(i=0;i<=p->numero_itens;i++) {
  if(!strcmp(tmp.descricao,p->prod[i].descricao)) {
    printf("Entre com o novo preco do produto");
    scanf("%f",&tmp.preco);
    p->prod[i].preco=tmp.preco;
    imprime_produto(p->prod[i]);
    i=0;
    break;
  }
 }
 if(i!=0) printf("\n\nEste produto nao existe no
almoxarifado selecionado!");}
```

```
// CONSTANTS ////////////////////////////////////

#include<stdio.h>
#include<conio.h>
#include<string.h>

// STRUCTS ////////////////////////////////////////

struct produto {
 char codigo[20];
 char descricao[20];
 float preco;
};

struct estoque {
 produto prod[1000];
 int numero_itens;
};

estoque matriz;
estoque filial;

// PROTOTYPES ////////////////////////////////////

void imprime_produto(produto p);
produto le_produto(void);
void cadastra_produto(estoque *p);
estoque *almoxarifado(void);
void consulta_preco(estoque *p);
void atualiza_preco(estoque *p);

// FUNCTIONS ////////////////////////////////////

void main(void){
 int menu;
 estoque *almox;
 while(menu!=4) {
  printf("+----------------------------------------+\n");
  printf("|          PRODUTOS                      |\n");
  printf("|                                        |\n");
  printf("| [ 1 ] Cadastra Produto                 |\n");
  printf("| [ 2 ] Consulta Preco                   |\n");
  printf("| [ 3 ] Atualiza Preco                   |\n");
  printf("| [ 4 ] Sair do Programa                 |\n");
  printf("|                                        |\n");
  printf("+----------------------------------------
+\n\n");
  printf("Digite a opcao desejada: ");
  scanf("%d", &menu);
  switch (menu) {
   case 1: clrscr();
    almox=almoxarifado();
    cadastra_produto(almox);
    getch();
    break;
   case 2: clrscr();
    almox=almoxarifado();
    consulta_preco(almox);
    getch();
    break;
   case 3: clrscr();
    almox=almoxarifado();
    atualiza_preco(almox);
    getch();
    break;
   case 4: printf("\nObrigado por utilizar nossos servicos.
  Volte sempre!!!");
    getch();
  }
  clrscr();
 }
}

void imprime_produto(produto p) {
 printf("\n\nCodigo: %s",p.codigo);
 printf("\nDescricao: %s",p.descricao);
 printf("\nPreco: %.2f",p.preco);
}

produto le_produto(void) {
 produto tmp;
 printf("\n\nDigite o codigo: ");
 scanf("%s", tmp.codigo);
 printf("\nDigite descricao: ");
 scanf("%s", tmp.descricao);
 printf("\nDigite o preco: ");
 scanf("%f",&tmp.preco);
 return(tmp);
}

void cadastra_produto(estoque *p) {
 p->prod[p->numero_itens]=le_produto();
 p->numero_itens++;
}

estoque *almoxarifado(void) {
 int menu2;
 estoque *tmp;
 printf("+----------------------------------------+\n");
 printf("|   SELECAO DE ESTOQUE                   |\n");
 printf("|                                        |\n");
 printf("| [ 1 ] Estoque Matriz                   |\n");
 printf("| [ 2 ] Estoque Filial                   |\n");
 printf("|                                        |\n");
 printf("+----------------------------------------
+\n\n");
 printf("Digite a opcao desejada: ");
 scanf("%d",&menu2);
 switch (menu2) {
  case 1: tmp=&matriz;
   break;
  case 2: tmp=&filial;
 }
 return(tmp);
}

void consulta_preco(estoque *p) {
 int i;
 produto tmp;
 clrscr;
 printf("Entre com a descricao do produto: ");
 scanf("%s",&tmp.descricao);
 for(i=0;i<=p->numero_itens;i++) {
  if(!strcmp(tmp.descricao,p->prod[i].descricao)) {
   printf("\n\nO preco do produto %s e R$ %.2f .",p-
>prod[i].descricao,p->prod[i].preco);
   i=0;
   break;
  }
 }
 if(i!=0) printf("\nProduto nao cadastrado");
}

void atualiza_preco(estoque *p) {
 int i;
 produto tmp;
 printf("Entre com a descricao do produto:");
 scanf("%s",&tmp.descricao);
 for(i=0;i<=p->numero_itens;i++) {
  if(!strcmp(tmp.descricao,p->prod[i].descricao)) {
   printf("Entre com o novo preco do produto");
   scanf("%f",&tmp.preco);
   p->prod[i].preco=tmp.preco;
   imprime_produto(p->prod[i]);
   i=0;
   break;
  }
 }
 if(i!=0) printf("\n\nEste produto nao existe no
almoxarifado selecionado!");
}
```

LISTA DE EXERCÍCIOS 6

Exercício 1 Escreva um programa que cria um arquivo texto do nome "teste.txt" e escreve a string "Unicenp - Centro Universitário Positivo" nas dez primeiras linhas do arquivo.

Exercício 2 Escreva um programa que solicita ao usuário duas strings como parâmetros de entrada. Uma string conterá o nome de um arquivo e a outra conterá um texto qualquer. O programa deverá abrir o arquivo indicado pelo usuário e adicionar o texto fornecido no fim do arquivo.

Exercício 3 Escreva um programa que executa as seguintes instruções:

- Solicita que o usuário forneça um nome de arquivo do tipo texto;
- Abre o arquivo texto;
- Conta o número de ocorrências das letras 'A', 'C', 'G' e 'T' presentes nele;
- Imprime o número de ocorrências dos caracteres solicitados na tela;
- Fecha o arquivo.

Exercício 4 Faça um programa que realiza backups de arquivos. O programa deverá solicitar ao usuário o nome de um arquivo e então, gerar uma cópia deste adicionando ao nome do arquivo a extensão (.BAK).
Obs: Pode-se usar para concatenar strings a função strcat.

Exercício 5 Escreva um programa que realiza a comparação de arquivos. O programa deverá solicitar ao usuário o nome de dois arquivos e então, comparar os arquivos, caracter a caracter, e imprimir se os arquivos são iguais ou há diferenças entre eles.

Exercício 6 Considere um arquivo de dados do tipo texto com o seguinte conteúdo:

```
3
CHAPINHA
7.5 6.0 8.5 4.5
MARZOLINHA
7.5 3.5 5.5 8.5
NINJA
5.0 6.0 7.0 8.8
```

O arquivo acima é apenas um exemplo. Nestes arquivos de alunos a primeira linha contém o número de alunos no arquivo. As linhas seguintes contêm os seguintes dados:

- nome do aluno com no máximo 50 caracteres;
- notas dos quatro bimestres.

Escreva um programa que imprime os nomes dos alunos aprovados, isto é, os que possuem a média igual ou superior a 7.0.

```
#include<stdio.h>
#include<conio.h>

Void main()
{
FILE *p;
int i;
p=fopen("test.txt","w");
if(p==NULL)
{
printff("Erro na aberura do arquivo);
return;
}
for (i=0;i<10;i++);
{
fputs("unicenp\n",p);
}
fclose(p);
}
```

```
#include <stdio.h>
#include <string.h>

void main()

{
   char nome[50];
   FILE *n_ori, *n_bak;
   printf("Nome do arquivo a ser feito o backup\n");
   gets(nome);
   if (( f_origem = fopen (file, "r")) == NULL )
   {
    printf("Erro na abertura do arquivo!");
    getch();
    exit(1);
   }
   n_bak=fopen(strcat(nome,".bak"),"w");
   while(!feof(n_ori))
   {
    fputc(fgetc(n_ori), n_bak);
   }
   fclose(n_ori);
   fclose(n_bak);
}
```

LISTA DE EXERCÍCIOS 7

Exercício 1 Escreva um programa que abre um arquivo texto de nome "msg.txt" e gera uma cópia criptografada do mesmo com nome "msg_cript.txt". A criptografia consiste em substituir letras organizadas aos pares conforme a tabela abaixo. Por exemplo, ao se encontrar a letra 'E' no arquivo original deve-se substituí-la por 'O' e vice-versa.

Z	E	N	I	T
P	O	L	A	R

Exercício 2 Escreva um programa que solicita que o usuário digite um texto qualquer, leia e armazene este texto em uma string temporária e então salve essa string em um arquivo binário de nome "texto.dat".

Exercício 3 Escreva um programa que solicita que o usuário digite 3 valores inteiros, leia estes valores e salve-os em um arquivo binário de nome "inteiros.dat".

Exercício 4 Considere a seguinte estrutura:

```
struct Produto
{
    char codigo[20];
    char descricao[20];
    float preco;
};
```

Escreva um programa que leia do teclado as informações de uma variável Produto e adicione essas informações em um arquivo binário chamado "produto.dat".

Exercício 5 Escreva um programa que declara uma matriz global de 1000 elementos do tipo Contato descrita logo abaixo:

```
struct Contato
{
    char nome[100];
    char email[100];
};
```

5.1 Escreva uma função que recebe uma variável do tipo Contato como parâmetro de entrada e adiciona essa variável em um arquivo binário chamado "contatos.dat".

```
void adiciona_contato (Contato x);
```

5.2 Escreva uma função que lê as variáveis Contato do arquivo e armazena-as em uma matriz que é passada como parâmetro da função. A função deverá ler as variáveis até que o fim do arquivo seja alcançado.

```
void carrega_contatos (Contato m[]);
```

```
#include <stdio.h>
#include <string.h>
#include <conio.h>

void main()

{
//   char nome[50];
   char tmp;
   FILE *n_ori, *n_crip;
   if (( n_ori = fopen ("msg.txt", "r")) == NULL )
   {
    printf("Erro na abertura do arquivo!");
    getch();
    exit(1);
   }
   if (( n_crip = fopen ("msg_cript.txt", "w")) == NULL )
   {
    printf("Erro na criptografia do arquivo!");
    getch();
    exit(1);
   }
   do{
   tmp = getc(n_ori);
   switch(tmp)
      {
      case 'Z': tmp = 'P'; break;
      case 'E': tmp = 'O'; break;
      case 'N': tmp = 'L'; break;
      case 'I': tmp = 'A'; break;
      case 'T': tmp = 'R'; break;
      case 'P': tmp = 'Z'; break;
      case 'O': tmp = 'E'; break;
      case 'L': tmp = 'N'; break;
      case 'A': tmp = 'I'; break;
      case 'R': tmp = 'T'; break;
      }
      if (tmp != EOF)
        fprintf(n_crip, "%c", tmp);
      }while (tmp!=EOF);

      fclose(n_ori);
      fclose(n_crip);
}
```