ESTRUTURAS e TEMPO

```
//Criando uma estrutura
struct data
{
   int dia;
   int mes;
   int ano;
};
//Criando uma variável de uma estrutura e atribuindo valores
struct data
{
   int dia;
   int mes;
   int ano;
};
struct data nascimento;

nascimento.dia = 13;
nascimento.mes = 5;
nascimento.ano = 2000;

//Criando um vetor de uma estrutura mais complexa
struct Funcionario
{
   char nome[64];
   int idade;
   char ssan[11]; //seguro social
   int categoria;
   float salario;
   int num_funcionario;
   struct data
   {
      int dia;
      int mes;
      int ano;
   } emp_datas[3];
} equipe[100];
//Atribuição em vetor de estruturas

equipe[10].emp_datas[0].dia = 17;
equipe[10].emp_datas[0].mes = 7;
equipe[10].emp_datas[0].ano = 1998;
```

```
//Passando uma estrutura a uma função (estrutura.c)
#include <stdio.h>

struct Forma
{
   int tipo;
   int cor;
   float raio;
   float area;
   float perimetro;
};
void exibe_estru(struct Forma forma)
{
   printf("forma.tipo %d\n", forma.tipo);
   printf("forma.cor %d\n", forma.cor);
   printf("forma.raio %f\n forma.area %f\n forma.perimetro %f\n",
      forma.raio, forma.area, forma.perimetro);
 }
void main(void)
{
   struct Forma circulo;

   circulo.tipo = 0;
   circulo.cor = 1;
   circulo.raio = 5.0;
   circulo.area = 22.0/7.0*circulo.raio*circulo.raio;
   circulo.perimetro = 2.0 * 22.0 / 7.0 * circulo.raio;
   exibe_estru(circulo);
}
// Alterando uma estrutura dentro de uma função (altera.c)
#include <stdio.h>

struct Forma {
  int tipo;
  int cor;
  float raio;
  float area;
  float perimetro;
};
void muda_estru(struct Forma *forma){
   (*forma).tipo = 0;
   (*forma).cor = 1;
   (*forma).raio = 5.0;
   (*forma).area = 22.0/7.0 * (*forma).raio * (*forma).raio;
   (*forma).perimetro = 2.0*22.0/7.0 * (*forma).raio;
}
```

```
void main(void)
{
   struct Forma circulo;

   muda_estru(&circulo);
   printf("circulo.tipo %d\n", circulo.tipo);
   printf("circulo.cor %d\n", circulo.cor);
   printf("circulo.raio %f\n circulo.area %f\n circulo.perimetro %f\n",
      circulo.raio, circulo.area, circulo.perimetro);
}
// Inicializando uma estrutura (inicializa.c)
#include <stdio.h>

void main(void)
{
   struct Forma
  {
      int tipo;
      int cor;
      float raio;
      float area;
      float perimetro;
   } circulo = {0, 1, 5.0, 78.37, 31.42};

   printf("circulo.tipo %d\n", circulo.tipo);
   printf("circulo.cor %d\n", circulo.cor);
   printf("circulo.raio %f\n circulo.area %f\n circulo.perimetro %f\n",
       circulo.raio, circulo.area, circulo.perimetro);
}
//Trabalhando com <time.h>: Estruturas
struct tm
{
  int tm_sec;          /* Seconds.        [0-60] (1 leap second) */
  int tm_min;          /* Minutes.        [0-59] */
  int tm_hour;         /* Hours.          [0-23] */
  int tm_mday;         /* Day.            [1-31] */
  int tm_mon;          /* Month.          [0-11] */
  int tm_year;         /* Year            - 1900.  */
  int tm_wday;         /* Day of week.    [0-6] */
  int tm_yday;         /* Days in year.   [0-365]       */
  int tm_isdst;        /* Horário de verão [-1/0/1]*/
}
```

```
// Trabalhando com <time.h>: Estruturas (tempo.c)
#include <stdio.h>
#include <time.h>

void main(void){
   time_t hora_atual;
   time_t hora_inicio;

   printf("Prestes a fazer uma pausa de 5 segundos\n");
   time(&hora_inicio); // Pega o tempo inicial em segundos
   do {
      time(&hora_atual); }
   while ((hora_atual - hora_inicio) < 5);
   printf("Acabado\n");
}
// Converte data e hora para string
#include <stdio.h>
#include <time.h>

void main(void)
{
   time_t hora_atual;

   time(&hora_atual);  // Pega o tempo em segundos
   printf("A data e hora atuais são: %s", ctime(&hora_atual));
}
//Uniões
struct Empdatas
{
    int dias_trabalhados;
    struct UltData
   {
      int dia;
      int mês;
      int ano;
   } ultimo_dia;
};       //  4 * int = 8 bytes
union Empdatas
{
   int dias_trabalhados;      //  dias_trabalhados ou UltData
   struct UltData
   {
      int dia;
      int mês;
      int ano;
   } ultimo_dia;
}; //  3 * int = 6 bytes
```

```
//Economizando memória com uniões  (uniao.c)
#include <stdio.h>

void main(void)
{
   union FuncionariosDatas
   {
      int dias_trabalhados;
      struct Data
     {
        int mes;
        int dia;
        int ano;
     } ultimo_dia;  // Data - requer 12 bytes
  } emp_info;

  union Numeros
  {
    int a;
    float b;
    long c;
    double d;              // double - requer 8 bytes
  } valor;

  printf("Tamanho de FuncionariosDatas %d bytes\n", sizeof(emp_info));
  printf("Tamanho dos Números %d bytes\n", sizeof(valor));
}
```