Mecatrônica

STRUCT - Programação II

Autor Prof.: Delcino Picinin Júnior

Struct

Struct também conhecidas como Registros, definem tipos de dados que agrupam variáveis sob um mesmo tipo de dado.

A ideia de usar uma **struct** é permitir que, ao armazenar os dados de uma mesma entidade, isto possa ser feito com uma única variável. Por exemplo, se for preciso armazenar a altura, o peso e a idade de uma pessoa, pode-se criar uma **struct** chamada Pessoa e agrupar os dados em um único tipo de dado, conforme o exemplo a seguir.

Código 1: Exemplo de Struct em C

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

typedef struct {
    float Peso, Altura;
    int Idade;
    char Nome[20];
} Pessoa;

void main(){
    Pessoa P1, P2,VetorPessoas[10];

    P1.Idade = 15;
    P1.Peso = 60.5;
    P1.Altura = 1.75;
    strcpy(P1.Nome,"Luiza");

    VetorPessoas[4].Idade = 23;
    VetorPessoas[4].Peso = 75.3;
    VetorPessoas[4].Altura = 1.89;
    strcpy(VetorPessoas[4].Nome,"Marcia");
```

Passagem de Structs por Parâmetro

Para passar uma struct por valor basta declará-la como um dos parâmetros, como no exemplo a seguir

Código 2: Exemplo Struct por cópia

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

typedef struct {
    float Peso, Altura;
    int Idade;
    char Nome[20];
} Pessoa;

void ImprimePessoa(Pessoa P);

void main(){
    Pessoa P1, P2,VetorPessoas[10];

    P1.Idade = 15;
    P1.Peso = 60.5;
    P1.Altura = 1.75;
    strcpy(P1.Nome,"Luiza");

    VetorPessoas[4].Idade = 23;
    VetorPessoas[4].Peso = 75.3;
    VetorPessoas[4].Altura = 1.89;
    strcpy(VetorPessoas[4].Nome,"Marcia");

    P2 = VetorPessoas[4];

    ImprimePessoa(P2);
}


void ImprimePessoa(Pessoa P)
{
  printf("Nome: %s Idade: %d Peso: %f Altura: %f\n",P.Nome, P.Idade, P.Peso, P.Altura);
}
```

Passagem de Structs por Referência

Para passar uma struct por referência, deve-se passar um ponteiro para a struct, como no exemplo a seguir.

Código 3: Exemplo Struct por cópia

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

typedef struct {
    float Peso, Altura;
    int Idade;
    char Nome[20];
} Pessoa;

void ImprimePessoa(Pessoa P);
void Atribui1(Pessoa *P);
void Atribui2(Pessoa *P);

void main(){
    Pessoa P1, P2;

    Atribui1(&P1);
    ImprimePessoa(P1);
    Atribui2(&P2);
    ImprimePessoa(P2);
}

void Atribui1(Pessoa *P){
    (*P).Idade = 15;
    (*P).Peso = 60.5;
    (*P).Altura = 1.75;
    strcpy((*P).Nome,"Luiza");
}
void Atribui2(Pessoa *P){
    P->Idade = 25;
    P->Peso = 63.5;
    P->Altura = 1.67;
    strcpy(P->Nome,"Flávia");
}


void ImprimePessoa(Pessoa P)
{
   printf("Nome: %s Idade: %d Peso: %f Altura: %f\n",P.Nome, P.Idade, P.Peso, P.Altura);
}
```

Retorno de Struct

Como uma struct define um tipo de dado, este tipo pode ser retornado em uma função, da mesma forma que ocorre com qualquer outro tipo de dado.

No exemplo a seguir, a função retorna uma struct que conterá, em seus campos, os dados que foram recebidos por parâmetro.

Código 4: Exemplo Struct por cópia

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

typedef struct {
    float Peso, Altura;
    int Idade;
    char Nome[20];
} Pessoa;

void ImprimePessoa(Pessoa P);
Pessoa SetPessoa();


void main(){
    Pessoa P1;

    P1=SetPessoa();
    ImprimePessoa(P1);
}

Pessoa SetPessoa(){
    Pessoa P;
    P.Idade = 15;
    P.Peso = 60.5;
    P.Altura = 1.75;
    strcpy(P.Nome,"Luiza");
    return P;
}

void ImprimePessoa(Pessoa P)
{
   printf ("Nome: %s Idade: %d Peso: %f Altura: %f\n",P.Nome, P.Idade, P.Peso, P.Altura);
}
```