Unoeste

Algoritmos e Estruturas de Dados

struct

Profª. Me. Carla Plantier Message
Profª. Me. Dalila Espinhosa
2024

Introdução

- Vimos até o momento variáveis que podemos chamar como simples (int, char e float) e compostas homogêneas (vetor e matriz).

- Agora, vamos aprender como criar um registro (estrutura), que na linguagem C, chamamos de struct.

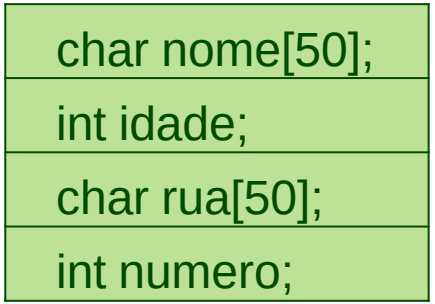
struct

- Uma struct pode ser vista como um **novo tipo de dado**, que é formado por outros tipos de dados.
  - Uma struct deve ser declarada após inclusão das bibliotecas e antes do main().
  - Ela é declarada da seguinte forma:

```
struct nomestruct{
    tipo1 campo1;
    tipo2 campo2;
    ...
    tipoN campoN;
};
```

- Uma struct pode ser vista como um agrupamento de dados.
- Ex.: cadastro de pessoas.
  - Todas essas informações são da mesma pessoa, logo podemos agrupá-las.
  - Isso facilita também lidar com dados de outras pessoas no mesmo programa.

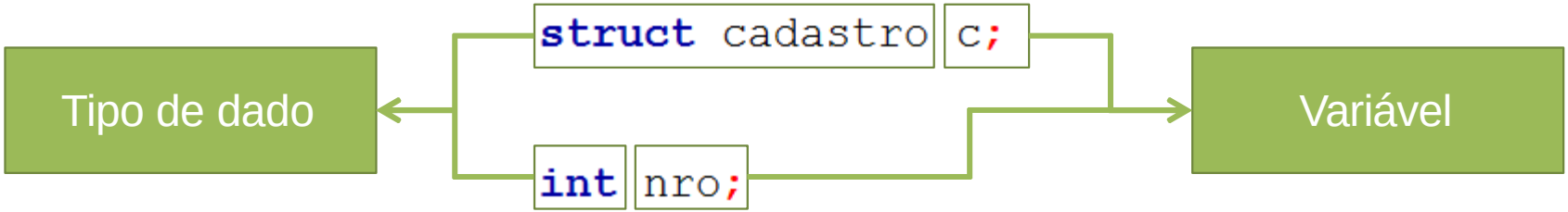
```
struct cadastro{
    char nome[50];
    int idade;
    char rua[50]
    int numero;
};
```



cadastro

Declaração

- Uma vez definida a estrutura, uma **variável** pode ser declarada de modo similar aos tipos já existente: `struct cadastro c;`
- **Obs**: por ser um tipo definido pelo programador, usa-se a palavra **struct** antes do tipo da nova variável.



Exercício

- Declare uma estrutura capaz de armazenar o RA e 2 notas para um dado aluno.

Solução

- O uso de struct facilita na manipulação dos dados do programa. Imagine declarar 4 cadastros, para 4 pessoas diferentes:

```
char nome1[50], nome2[50], nome3[50], nome4[50];
int idade1, idade2, idade3, idade4;
char rua1[50], rua2[50], rua3[50], rua4[50]
int numero1, numero2, numero3, numero4;
```

- Utilizando uma struct, o mesmo pode ser feito da seguinte maneira:

```
struct cadastro{
    char nome[50];
    int idade;
    char rua[50]
    int numero;
};

//declarando 5 cadastros
struct cadastro c1, c2, c3, c4, c5;
```

Como acessar?

- Como é feito o acesso às variáveis da struct?
  - Cada variável da estrutura pode ser acessada com o operador ponto “.”.
  - Ex.:

```
//declarando a variável
struct cadastro c;

//acessando os seus campos
strcpy(c.nome,"João");
scanf("%d",&c.idade);
strcpy(c.rua,"Avenida 1");
c.numero = 1082;
```

- Como nos vetores, uma struct pode ser previamente inicializada:

```
struct ponto {
    int x;
    int y;
};

struct ponto p1 = { 220, 110 };
```

- Como ler os valores das variáveis da struct do teclado?
  - basta ler cada variável independentemente, respeitando seus tipos.

```
struct cadastro c;

gets(c.nome);//string
scanf("%d",&c.idade);//int
gets(c.rua);//string
scanf("%d",&c.numero);//int
```

- Note que cada variável dentro da struct pode ser acessada como se apenas ela existisse, não sofrendo nenhuma interferência das outras.
  - Uma estrutura pode ser vista como um simples agrupamento de dados.
  - Se faço um **scanf** para **estrutura.idade**, isso não me obriga a fazer um **scanf** para **estrutura.numero**

Exercícios

1) Crie uma estrutura para armazenar uma data, como no exemplo 04-03-2024. Depois peça para o usuário informar a data e exibir o ano da data informada.
2) Crie uma estrutura para armazenar o código e preço de um produto. Depois peça para o usuário informar os dados e exibir.
3) Defina uma estrutura para armazenar a ficha de um paciente (código, peso, altura, idade). Preencher e exibir os dados do paciente.
4) Defina uma estrutura para armazenar o cadastro de estoque (código da peça, preço unitário da peça e quantidade em estoque). Cadastrar e exibir os dados da peça.
5) Implemente um programa que leia o nome, o sexo, a idade e a raça de um cão. Use struct para ler os dados. Exibir os dados do cão.

OBS: Implementar os exercícios por meio funções.

Unoeste