INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SUL-RIO-GRANDENSE
Campus Passo Fundo

ALGORITMOS II

Prof. Adilso Nunes de Souza

REGISTRO

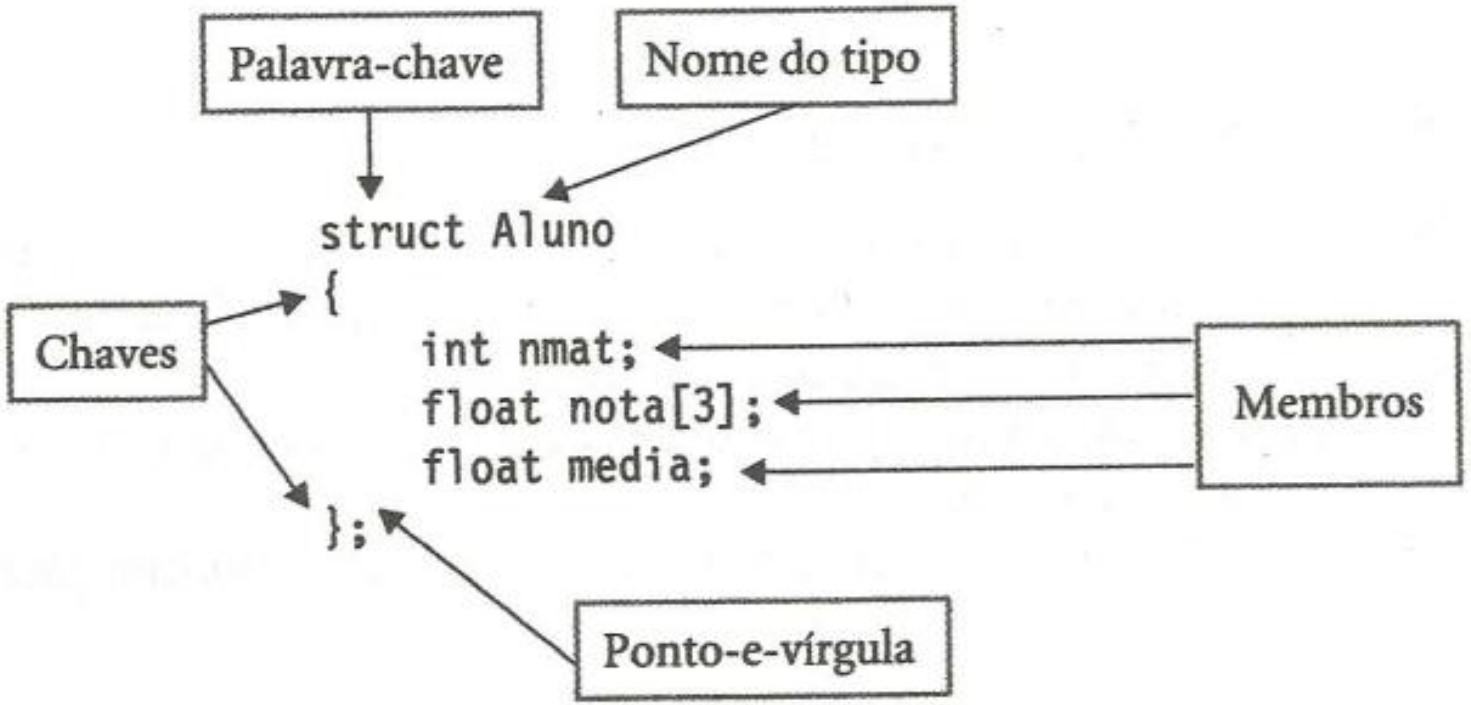
- Registros (struct) são estruturas de dados capazes de agregar várias informações de tipos diferentes.
- Agrupamento de variáveis sob um nome específico, sendo conhecido como tipo de dados agregado ou conglomerado
- Cada informação contida em um registro é chamada de campo ou membro da estrutura.
- São conhecidos como variáveis compostas heterogêneas.

REGISTRO

- Para utilizar uma variável do tipo registro o primeiro passo é declará-la, isto significa especificar o nome de seus campos com seus respectivos tipos.
- Em C/C++ um registro é declarado usando a palavra reservada struct, sendo sempre finalizado com um ponto e vírgula.
- Sintaxe:

```
struct nome_do_registro
{
    tipo nome_do_campo;
};
```

EXEMPLO


Palavra-chave
Nome do tipo
struct Aluno
{
Chaves
int nmat;
float nota[3];
float media;
Membros
};
Ponto-e-vírgula

EXEMPLO

```
struct cadastro
{
    char nome[50];
    char endereco[100];
    int idade;
    float salario;
};
```

REGISTRO

- A partir da estrutura definida, o programa poderá considerar que existe um novo tipo de dado, que poderá ser utilizado.

  cadastro x; //a variável x e do tipo cadastro.
- O registro criado só poderá ser utilizado dentro do bloco onde foi definido;
- Para que um registro seja visível em todo o programa o mesmo deve ser declarado na mesma região das variáveis globais.

REGISTRO

- Como registro é um novo tipo é necessário ter variáveis deste novo tipo.
- Formas possíveis de declaração:

➢ Declara a estrutura, atribuindo um nome e depois utiliza este nome para criar as variáveis do tipo da estrutura.

```
struct numero
{
   int num;
   float res;
};
numero x;
```

REGISTRO

- Ao declarar o registro já definir a variável ou variáveis deste tipo.

```
struct numero
{
   int num;
   float res;
} x, y, z;
```

REGISTRO

- Se for necessário apenas uma variável da estrutura, o nome da estrutura pode ser omitido, definindo diretamente a variável.

```
struct
{
   int num;
   float res;
} x;
```

- OBS: desta forma não será possível criar outras variáveis do tipo da estrutura, todas necessitam ser definidas durante a declaração da estrutura.

REGISTRO

- A inicialização dos campos da struct é semelhante à inicialização de um array

```
struct numero
{
   int num;
   string nome;
   float res;
   float nota[3];
};
```

```
main()
{
   numero x = {5, "Pedro da Silva", 8.45, 1.5, 6.9, 8.0};

   cout << x.num << endl;
   cout << x.nome << endl;
   cout << x.res << endl;
   cout << x.nota[0] << endl;
   cout << x.nota[1] << endl;
   cout << x.nota[2] << endl;
}
```

MANIPULAÇÃO DOS DADOS DA ESTRUTURA

- Para inserir valores para uma variável do tipo registro, basta indicar o nome da variável ponto(.) e o nome do campo declarado no registro.
- Sintaxe:

```
x.num = 8;
cin >> x.num;
cout << x.num;
```

MANIPULAÇÃO DOS DADOS DA ESTRUTURA

- A informação contida em uma variável do tipo da estrutura pode ser atribuída a outra variável do mesmo tipo.

```
struct
{
   int num;
   float res;
}x, y;
x.num = 12;
cout <<  x.num << endl;
y = x;
cout <<  y.num << endl;
```

- A variável y passa a conter os mesmos valores da variável x

MANIPULAÇÃO DOS DADOS DA ESTRUTURA

- Esta instrução de atribuição entre duas variáveis do tipo struct não pode ser usada por array, os quais devem ser atribuídos elemento por elemento.

```
struct alunos
{
   int mat;
   string nome;
   float nota[3];
   float media;
};
```

```
main()
{
   alunos vet[10], aux[10];
    vet[0].mat = 11;
    vet[1].media = 8.0;

    aux = vet; // operação inválida
}
```

STRUCT COM FUNÇÃO

- Para manipular uma variável que foi definida do tipo de uma struct, devemos declarar os parâmetros na função, também sendo do mesmo tipo.

STRUCT COM FUNÇÃO

```
void ler(cad &a)
{
      cout << "Informe o nome: ";
      gets(a.nome);
      fflush(stdin);

      cout << "Informe a idade: ";
      cin >> a.idade;
      fflush(stdin);

      cout << "Informe o salario: ";
      cin >> a.sal;
      fflush(stdin);
}
```

```
struct cad
{
   char nome[50];
   int idade;
   float sal;
};

main()
{
   cad teste;
   ler(teste);
}
```

STRUCT COM FUNÇÃO

- Em caso de vetor

```
struct cad
{
   char nome[50];
   int idade;
   float sal;
};

main()
{
   cad teste[3];
   ler(teste);
}
```

```
void ler(cad vet[])
{
   int x;
   for(x = 0; x < 3; x++)
   {
      cout << "Informe o nome: ";
      gets(vet[x].nome);
      fflush(stdin);

      cout << "Informe a idade: ";
      cin >> vet[x].idade;
      fflush(stdin);

      cout << "Informe o salario: ";
      cin >> vet[x].sal;
      fflush(stdin);
   }
}
```

STRUCT COM FUNÇÃO

- Em caso de vetor com um campo de vetor na struct
  - Ver exemplo 11

- Em caso de vetor controlando quais posições serão ocupadas e com menu
  - Ver exemplo 12

STRUCT ANINHADAS

- A linguagem C++ permite a definição de estruturas aninhadas, isto significa que é possível que algum membro de um estrutura seja definido como sendo outra estrutura previamente declarada e no mesmo escopo.
  - Ver exemplo 13 e 14

REFERÊNCIAS

- ARAÚJO, Jáiro – Dominando a Linguagem C. Editora Ciência Moderna.
- PEREIRA, Silvio do Lago. Estrutura de Dados Fundamentais: Conceitos e Aplicações, 12. Ed. São Paulo, Érica, 2008.
- LORENZI, Fabiana. MATTOS, Patrícia Noll de. CARVALHO, Tanisi Pereira de. Estrutura de Dados. São Paulo: Ed. Thomson Learning, 2007.
- VELOSO, Paulo. SANTOS, Celso dos. AZEVEDO, Paulo. FURTADO, Antonio. Estrutura de dados. Rio de Janeiro: Ed. Elsevier, 1983 27ª reimpressão.