Introdução à Programação II

Variáveis Compostas Heterogêneas

Material da Prof. Ana Eliza

Variáveis Compostas Heterogêneas

- **Estruturas**
  - São variáveis compostas heterogêneas, ou seja, cujos componentes podem ser de tipos diferentes.
  - São também chamados de **registros**.

Variáveis Compostas Heterogêneas

- **Definição da Estrutura (sintaxe)**

```
struct rótuloDaEstrutura {
        tipoCampo1 nomeCampo1;
        tipoCampo2 nomeCampo2;
        ...
        tipoCampoN nomeCampoN;
};
```

- **Declaração das Variáveis (sintaxe)**

```
struct rótuloDaEstrutura nomeVariavel1,..., nomeVariavelN;
```

Variáveis Compostas Heterogêneas

- **Definição da Estrutura (exemplo)**

```
struct tipoPessoa {
        char nome [30];
        int idade;
        float peso;
      };
```

- **Declaração das Variáveis (exemplo)**

```
struct tipoPessoa pessoa1, pessoa2;
```

Variáveis Compostas Heterogêneas

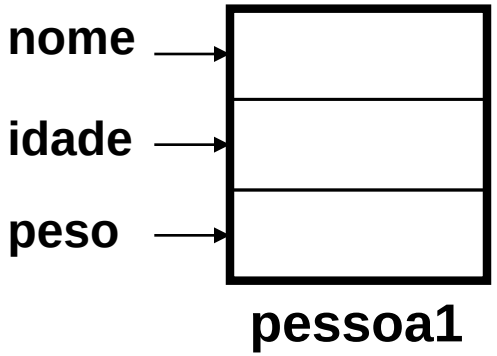
- **Estruturas**
  - O nome da variável composta (ex: pessoa1) identifica o "conjunto" como um todo.
  - Cada componente (campo) é identificado individualmente através de seu nome.
  - Sintaxe de acesso a um campo:

    nomeVariavel.nomeCampo
  - Exemplo:

    pessoa1.nome

    pessoa1.idade

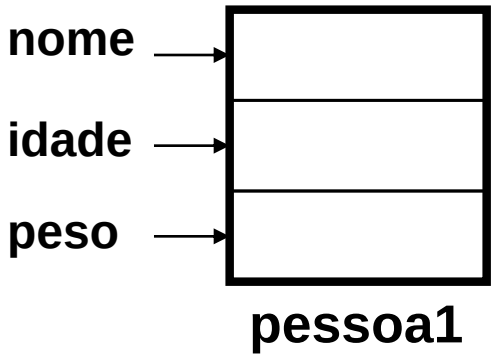
    pessoa1.peso



Variáveis Compostas Heterogêneas

- **Estruturas – Preenchimento**
  - O preenchimento de uma estrutura é feito campo a campo.
  - Exemplo:

```
pessoa1.idade = 10;
scanf("%f", &pessoa1.peso);
strcpy(pessoa1.nome,"Maria");
```



Variáveis Compostas Heterogêneas

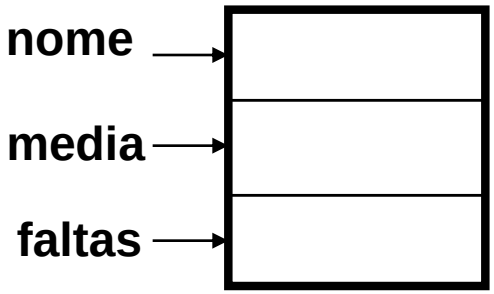
- **Estruturas – Acesso**
  - O acesso a uma estrutura para consulta ou exibição é feita campo a campo.
  - Exemplo:

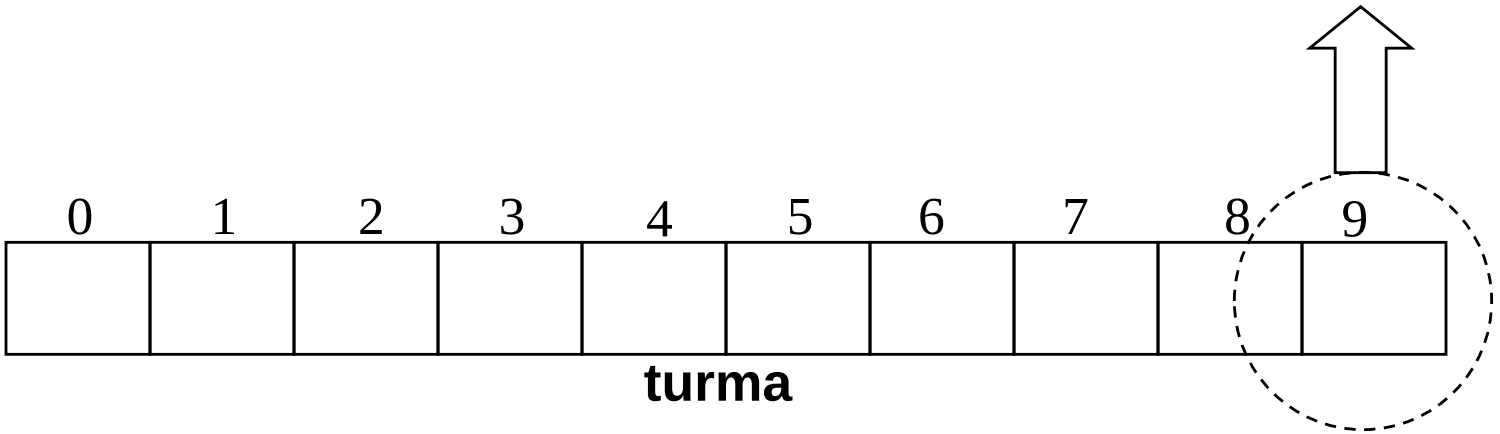
```
printf("%f", pessoa1.peso);
if (pessoa1.idade >= 18)
   printf("%s é maior de idade.",pessoa1.nome);
```

Variáveis Compostas Heterogêneas

- ## Vetores de Estruturas

```
struct tipoAluno {
        char nome [30];
        float media;
        int faltas;
};
 struct tipoAluno turma [10];
```





Variáveis Compostas Heterogêneas

- **Vetores de Estruturas – Preenchimento**
  - Exemplo

```
for (i = 0; i < 10; i++)
{
    printf("Informe o nome: ");
    gets(turma[i].nome);
    printf("Informe a média: ");
    scanf("%f",&turma[i].media);
    printf("Informe a qtd. de faltas: ");
    scanf("%i",&turma[i].faltas);
}
```

Variáveis Compostas Heterogêneas

- **Vetores de Estruturas – Exibição**
  - Exemplo

```
for (i = 0; i <= 9; i++)
{
    printf("Dados do aluno %i: \n",i+1);
    printf("Nome: %s \n", turma[i].nome);
    printf("Média: %f \n", turma[i].media);
    printf("Faltas: %i \n", turma[i].faltas);
}
```

Variáveis Compostas Heterogêneas

- **Estruturas como parâmetro de funções e procedimentos**

Exemplo 1:

```
void exibir (struct tipoAluno  al) { // passagem por valor
    printf("Nome: %s \n", al.nome);
    printf("Media: %f \n", al.media);
    printf("Faltas: %i \n", al.faltas);
}
```

Chamada:

```
exibir (aluno);
```

Variáveis Compostas Heterogêneas

- **Estruturas como parâmetro de funções e procedimentos**

Exemplo 2:

```
void preencher (struct tipoAluno  * al) { // por referência
    printf("Informe o nome: "); gets((*al).nome);
    printf("Informe a media: "); scanf("%f", &(*al).media);
    printf("Informe as faltas: "); scanf("%i", &(*al).faltas);
}
```

Chamada:

```
exibir (&aluno);
```

Variáveis Compostas Heterogêneas

- **Estruturas como parâmetro de funções e procedimentos**

Exemplo 3:

```
void preencher (struct tipoAluno  * al) { // por referência
    printf("Informe o nome: "); gets(al->nome);
    printf("Informe a media: "); scanf("%f",&al->media);
    printf("Informe as faltas: "); scanf("%i",&al->faltas);
}
```

Chamada:

```
exibir (&aluno);
```

Variáveis Compostas Heterogêneas

- **Estruturas como retorno de funções**

Exemplo 3:

```
struct tipoResp {
     int posMaior, posMenor;
};
...
struct tipoResp funcao_busca (int numeros [100]) {
    // função para retorna o posição do maior e do menor número
    struct tipoResp resp;
     ...
     return resp;
}
int main ( ) {
     struct tipoResp res;
     int nums [10];
     ...
     res = funcao_busca (nums);
```