UNIVALI – EMCT – Kobrasol – Ciência da Computação
Algoritmos e Programação de Computadores 2per

VARIÁVEIS HETEROGÊNEAS – REGISTRO

Um **REGISTRO** é um conjunto de dados logicamente relacionados, mas de tipos arbitrários.
O registro é o caso mais geral de variável composta na qual os elementos do conjunto não precisam ser, necessariamente, homogêneos ou do mesmo tipo.

Declaração de registros em pseudocódigo

```
TIPO
     NomeTipoReg = REGISTRO
          nome_componente1: TIPO
          ...
          nome_componenteN: TIPO
     FIMREGISTRO
```

```
TIPO
     Aluno =  REGISTRO
          nome, matricula: STRING
          sexo: CARACTER
          curso: INTEIRO
          medias_disciplinas: VETOR [1..52] DE REAL
     FIMREGISTRO
```

Declaração de registros em Linguagem C/C++: `struct`

- Há algumas maneiras, mas criar um tipo estruturado é a forma mais moderna e interessante para programar.

```
typedef struct {
     tipo nome_componente1;
     ...
     tipo nome_componenteN;
} NomeTipoStruct;
```

Ex.:

```
typedef struct {
   string nome, matricula;
   unsigned char sexo;
   int curso;
   float medias_disciplinas[52];
} Aluno;
```

```
typedef struct {
   string marca, tipoPlaca;
} Monitor;
```

```
typedef struct {
   int hd;
   Monitor video;
} PersonalComputer;
```

UNIVALI – EMCT – Kobrasol – Ciência da Computação
Algoritmos e Programação de Computadores 2per

Declaração de variável do tipo registro em pseudocódigo

```
VAR
  nomeVariavel: NomeTipoReg
```

Ex.:
```
VAR
  aluno1, aluno2: Aluno
```

Declaração de variável do tipo struct em Linguagem C/C++

```
VAR
  NomeTipoStruct nomeVariavel;
```

Ex.:
```
VAR
  Aluno aluno1, aluno2, alunos_cursoCC[300];
```

Acesso aos elementos

- Como as informações primitivas estarão dentro do registro, é necessário utilizar o operador de membro **"."**

Ex.:
```
LEIA (aluno1.nome)
LEIA (aluno1.matricula)
...
    LEIA (aluno1.medias_disciplinas[i])


LEIA (alunos_cursoCC[pos].nome)
...
LEIA (alunos_cursoCC[pos].medias_disciplinas[i])
```

Atribuição entre variáveis do tipo registro

```
TIPO
    Retangulo = REGISTRO
                    altura, largura: REAL
                FIMREGISTRO
VAR
    a, b: Retangulo
INICIO
    a.largura <- 12.7
    a.altura <- 4.0
    b.largura <- a.largura
    b.altura <- a.altura
FIM
```

Pode-se utilizar atribuição direta: b <- a

Problema em fazer assim: quando há componentes do tipo vetor

Tamanho de variável do tipo struct em Linguagem C/C++

- Para recuperar o tamanho em bytes do tipo usa-se a função `sizeof(tipo)`

Ex.: `sizeof(Retangulo)` -> 8 bytes