Instituto Superior de Engenharia de Coimbra

Licenciatura em Engenharia Informática – 19/20

Programação

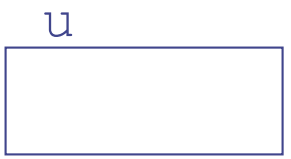
3A: *Unions* e *Enumerations*

Francisco Pereira (xico@isec.pt)

Unions

- Estruturas especiais em que apenas existe espaço para armazenar informação num dos campos

```
union{
   int i;
   float x;
   char s[3];
} u;
```



Espaço partilhado pelos campo `i, x, s`

Em cada instante apenas um pode guardar informação

Unions

- As uniões (unions) são constituídas por um ou mais campos.
- Os campos sobrepõem-se, existindo, em cada instante, apenas um campo ativo.
- Vantagem:
  - Permitem flexibilizar as estruturas

Propriedades das Unions

- Propriedades específicas
  - Não regista qual é a variável ativa
    - Solução: utilizar uma “etiqueta”
- Propriedades idênticas às das estruturas:
  - Declaração de tipos e definição de variáveis
  - Utilização em funções, relação com ponteiros e utilização do operador de atribuição (=)

Exemplo: Catálogo de publicações

- Armazenar informação relativa a diversas publicações
- Informação importante:
  - Tipo de publicação: Jornal ou Livro
  - Título;
  - Se for um livro:
    - Autor
  - Se for um jornal:
    - Data (dia, mês e ano)

Catálogo de publicações: Estrutura

Instituto Superior de Engenharia de Coimbra

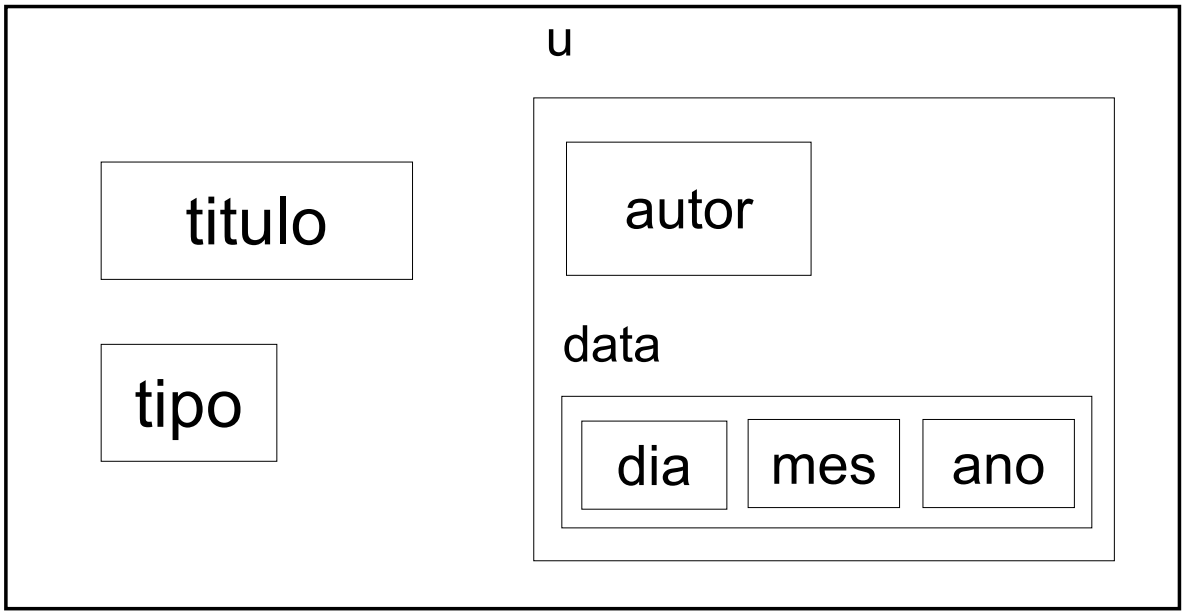
Etiqueta:
0: Jornal
1: Livro

Secção variável

```
#define MAX_T 50

struct pub{
   char titulo[MAX_T];
   int tipo;
   union{
        struct{
              int dia, mes, ano;
        } data;
        char autor[MAX_T];
   } u;
};
```

Catálogo de publicações: Estrutura




```
#define MAX_C 100

struct pub catalogo[MAX_C];
int total=0; /*numero de publicacoes armazenadas*/
```

Exemplo 1: Procura

- Procurar jornais publicados em 2005:

```
int i;

for(i=0; i<total; i++)
      if(catalogo[i].tipo == 0 &&
         catalogo[i].u.data.ano == 2005)
            puts(catalogo[i].titulo);
```

Exemplo 2: Insere

```
int insere_pub(struct pub *tab, int total)
{
   if(total == MAX_C)
   {
       puts("Nao existe espaço livre");
       return total;
   }
   else
   {
       printf("Titulo: ");
       scanf(" %49[^\n]", tab[total].titulo);
       printf("Tipo de publicacao: (0: Jornal - 1: Livro)");
       scanf(" %d", &tab[total].tipo);
...
```

Exemplo 2: Insere (cont.)

```
...
        if(tab[total].tipo == 0)
        {
             printf("Data: ");
             scanf("%d%d%d", &tab[total].u.data.dia,
                    &tab[total].u.data.mes,
                    &tab[total].u.data.ano);
        }
        else
        {
             printf("Autor: ");
             scanf(" %49[^\n]", tab[total].u.autor);
        }
        return total+1;
   }
}
```

Enumerações

- Criação de um tipo definido por enumeração:
  - Domínio discreto e limitado
    - Utilidade: Clarifica o código
  - Programador define completamente o domínio:
    - Conjunto de valores constantes

- Exemplo:
  - Criar um tipo por enumeração para os naipes de um jogo de cartas

Enumerações

- Criação do tipo:
  - `enum naipe{COPAS,OUROS,ESPADAS,PAUS};`

Tipo

Gama de valores

- Declaração de variáveis:
  - `enum naipe x, y;`

Enumerações

- Os nomes das constantes são sinónimos para valores inteiros
  - Por defeito:
    - `enum naipe{COPAS,OUROS,ESPADAS,PAUS};`

  - Outros valores especificados na declaração:
    - `enum naipe{COPAS=10,OUROS,ESPADAS,PAUS=20};`

Exemplo

```
enum naipe {COPAS, OUROS, ESPADAS, PAUS};
enum naipe x, y=COPAS;
int i;

x = PAUS;
y++;
i = x + 10;
if(x==ESPADAS || x==PAUS)
   printf("Naipe nao e vermelho\n");
switch(y){
   case COPAS:
   case OUROS: printf("Naipe vermelho\n"); break;
}
printf("%d\t%d\t%d\n", i, x, y);
```