

# Registros

Prof<sup>a</sup> Ma. Luciane Y. H. Kanashiro

# Introdução

- Vetores e matrizes:
  - Estruturas de dados homogêneas
  - Armazenam vários valores, mas todos de um mesmo tipo (todos int, todos double, todos float, todos char)

# Introdução

- Problemas reais Temos coleções de dados que são de tipos diferentes
- Exemplo: passagem de ônibus, que contém  
número da passagem: int Data: char[],  
origem: char[], destino:char[], horário:char[],  
poltrona:int, Idade: int , nome:char[]

---

Número da passagem: \_\_\_\_\_ Data: \_\_\_\_\_

De: \_\_\_\_\_ Para: \_\_\_\_\_

Horário: \_\_\_\_\_ Poltrona: \_\_\_\_\_ Idade: \_\_\_\_\_

Nome do passageiro: \_\_\_\_\_

---

# Registros

- Tipo de dado heterogêneo
- Conjunto de informações de tipos primitivos diferentes

# Registros

- Agregam vários dados acerca de uma mesma entidade
- Cada dado em um registro é chamado de campo
- Exemplos:
  - carro: cor, marca, ano, placa, chassi
  - pessoa: nome, idade, endereço

# Registros

- Mundo real: existe coleção de dados com tipos diferentes



**FÓRUM DE ARQUITETURA DE MOSSORÓ/RN** 

**FICHA DE INSCRIÇÃO**

Nome: \_\_\_\_\_

Endereço: \_\_\_\_\_

Profissão: \_\_\_\_\_ Fone.: \_\_\_\_\_

E-mail: \_\_\_\_\_

PROFISSIONAL	ASSOCIADO IAB	ESTUDANTE
R\$ 60,00	R\$ 40,00	R\$ 30,00

REALIZAÇÃO:  PROMOÇÃO:  APOIO: 

# Registros

Nº DE CONTROLE						
629702						
BILHETE DE PASSAGEM RODOVIÁRIO					Nº	
SÉRIE ÚNICA					1ª VIA - FISCO	
921067					Número do Bilhete	
DE			PARA			
SAO PAULO			BARRETOS			Cidade de destino da viagem
DATA DA VIAGEM	DIA DA SEMANA	HORA DA VIAGEM	POLTRONA	PLATAFORMA	DISTÂNCIA EM KM	
07/10/11	sexta-feira	15:00	01	19	0505	Número da poltrona
SERVIÇO	LINHA	PREFIXO				
1062	BARRETOS X SAO PAULO	8478				
DATA DA EMISSÃO	HORA DA EMISSÃO	TIPO DE ÔNIBUS	AGÊNCIA	AGENTE		
04/10/11	11:25	CONV.3	0081	DEC6		
FORMA PAGTO.	TARIFA	TAXA DE EMB.	PEDÁGIO	VALOR DA PASSAGEM		
Dinheiro	73,82	4,49	6,44	86,75		
BANDEIRA	Nº CARTÃO	AUTORIZAÇÃO	CONTROLE			
NOME DO TITULAR:						
ASSINATURA DO TITULAR:						
NORMAL						

# Registros

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

<b>MADEIRAS</b>  <b>TELHAS</b>  <b>PORTAS</b>  <b>JANELAS</b>  <b>EUCALIPTO TRATADO</b>  <b>DOBRADIÇAS</b>  <b>FECHADURAS</b>  <b>PORTÕES</b>  <b>PORTEIRAS</b>  <b>MANTA TÉRMICA</b>	 <b>PPIE</b> <small>RÓTULOS ADESIVOS e FORMULÁRIOS</small>		<b>ORÇAMENTO</b> Nº: DATA: HORA:		
	NOME:		VEND.:		
	ENDEREÇO:		TEL.:		
	CNPJ / CPF:		INSC. EST.:		
	QUANTIDADE	CÓDIGO	UNIDADE	DESCRIÇÃO	PREÇO UNITÁRIO
					

OBS.: **SEM VALOR FISCAL**

COND. PAGTO.:

<b>SUB - TOTAL</b>	
<b>DESC.: %</b>	
<b>TOTAL</b>	



# Sintaxe em C

- A palavra reservada *struct* indica ao compilador que está sendo criada uma estrutura
- Uma estrutura deve ser declarada após incluir as bibliotecas e antes da main

```
struct <identificador_struct> {  
    tipo <nome_variável_campo1>;  
    tipo <nome_variável_campo2>;  
    :  
} <variáveis_estrutura>;  
  
struct <identificador_struct> <var1>, <var2>;
```

# Sintaxe em C

- Se o compilador C for compatível com o padrão C ANSI
  - Informação contida em uma struct pode ser atribuída a outra struct do mesmo tipo
  - Não é necessário atribuir os valores de todos os elementos/campos separadamente
  - Por exemplo: `<var1> = <var2>;`
    - Todos os campos de `<var1>` receberão os valores correspondentes dos campos de `var2`

# Sintaxe em C

- Para acessar os campos da struct
  - Utiliza-se o nome da variável struct, seguido de ponto, seguido do nome do campo
  - Por exemplo: `<var1>.<nome_variavel_campo>;`

# Representação em C

Registro

```
struct participante{  
    char *nome;  
    char endereço[100];  
    char *profissão;  
    char fone[20] ;  
    char email[50];  
  
};
```

FÓRUM DE ARQUITETURA DE MOSSORÓ/RN		
		
FICHA DE INSCRIÇÃO		
Nome: _____		
Endereço: _____		
Profissão: _____		Fone.: _____
E-mail: _____		
PROFISSIONAL	ASSOCIADO IAB	ESTUDANTE
R\$ 60,00	R\$ 40,00	R\$ 30,00
REALIZAÇÃO: 	PROMOÇÃO: 	APOIO: 

# Manipulação

Registro

- Comando para declarar variável com a struct (registro)

```
struct participante p;
```

– Já vimos que para acessar os membros de uma struct deve-se usar nome\_variável.nome\_membro

- Setando informações contidas no registro

```
p.nome = "José";
```

```
strcpy(p.endereco, "rua das flores")
```

Imprimindo informações contidas no registro

```
printf("%s", p.nome);
```

# Manipulação

- Para obter o nome do participante e colocar na string nome da struct se poderia utilizar:

```
/*Exemplo com ponteiro*/
```

```
    p.nome = malloc(5);
```

```
    printf("Digite o nome do participante: ");
```

```
    gets(p.nome);
```

```
    printf("Nome: %s \n", p.nome);
```

```
/*Exemplo com string de char*/
```

```
    printf("Digite o endereco do participante: ");
```

```
    gets(p.endereco);
```

```
    printf("Endereco: %s \n", p.endereco);
```

# Registro de Conjuntos

- Campos podem ser compostos por outros tipos construídos.
- Ex: suponha que o registro tenha um campo que indique o valor pago por tipo de profissional, sendo que cada posição corresponde a um tipo de profissional

```
tipo vetorTipo[3];
```

**FÓRUM DE ARQUITETURA DE MOSSORÓ/RN** 

**FICHA DE INSCRIÇÃO**

Nome: \_\_\_\_\_

Endereço: \_\_\_\_\_

Profissão: \_\_\_\_\_ Fone.: \_\_\_\_\_

E-mail: \_\_\_\_\_

PROFISSIONAL	ASSOCIADO IAB	ESTUDANTE
R\$ 60,00	R\$ 40,00	R\$ 30,00

REALIZAÇÃO:  PROMOÇÃO:  APOIO: 

# Representação

Registro de Conjuntos

- Criar conjunto (Registro)

```
typedef struct tipo_categoria {  
    char categoria[50];  
    float valor;  
}tipo;
```



# Manipulação

Registro de Conjuntos

```
public class Participante{  
    char *nome;  
    char *endereço;  
    char *profissão;  
    char *fone ;  
    char *email;
```

**//Criando ponteiro para armazenar os registros tipo**

```
tipo *t;
```

```
}
```

```
typedef struct tipo_categoria {  
    char categoria[50];  
    float valor;  
}tipo;
```

# Manipulação

Registro de Conjuntos

```
tipo ttipo;
```

- Criando vetor para armazenar

```
tipo vetorTipo[3];
```

- Atribuindo valor aos campos do Conjunto

```
ttipo. Categoria("Estudante");
```

```
ttipo.Valor(30);
```

```
vetorTipo[0] = ttipo;
```

- Imprimindo

```
printf("%s\n", p.t[0].categoria);
```



```
printf("%f\n", p.t[0].valor);
```

- Atribuindo valor ao campo no Registro

```
p.t = vetorTipo;
```

# Conjunto de Registros

- As estruturas homogêneas podem armazenar as estruturas heterogêneas

		...	
---	--	-----	--

0

1

...

N-1

# Manipulação

Conjunto de Registros

- Criando vetor para armazenar os registros

```
struct participante vetor_participantes[100];
```

- Atribuindo valor aos campos do Registro

```
p.nome = "Maria";  
p.email="maria@gmail.com";  
p.endereco = "Rua XV";  
p.fone = "41 99993233";  
p.profissao = "arquiteta";  
p.t = vetorTipo;
```

Atribuindo valor aos campos do vetor

```
Vetor_participante[0] = p;
```

- Imprimindo valor dos campos do vetor

```
printf("%s\n", vetor_participantes[0].nome);  
printf("%f\n", vetor_participantes[0].t[0].valor);
```

# Referências

- [1] ASCENCIO, A. CAMPOS, E. **Fundamentos da programação de computadores: algoritmos, Pascal, C/C++ e Java**. 3 ed. São Paulo: Pearson Prentice Hall, 2007.
- [2]FORBELLONE, André Luiz Villar; EBERSPÄCHER, Henri Frederico. **Lógica de Programação: A Construção de Algoritmos e Estrutura de Dados**. 2ª Ed. São Paulo: Makron Books, 2000.
- [4]XAVIER, Gley Fabiano Cardoso. **Lógica de Programação**. 11ª Ed. São Paulo: SENAC São Paulo, 2007MANZANO, José Augusto N. G.; OLIVEIRA, Jayr Figueiredo de. **Algoritmos: Lógica para Desenvolvimento de Programação de Computadores**. 12ª Ed. São Paulo: Érica, 2001.
- <https://www.inf.pucrs.br/~cnunes/lapro/aulas/structs.pdf>