



SIN211 Algoritmos e Estruturas de Dados

Prof. João Batista Ribeiro

joao42ibatista@gmail.com



Universidade Federal de Viçosa

Slides baseados no material da Prof.^a Rachel Reis



Assuntos da Aula

- Arrays
- Estruturas
- Arrays e Estruturas



Arrays – Tipos

- Array de 1 dimensão → **Vetor**
- Array de 2 dimensões → **Matriz**



Arrays – O que são?

- São variáveis compostas homogêneas, ou seja, conjunto de variáveis do mesmo tipo
- Estrutura de Dados estática
- Ocupam posições consecutivas na memória principal



Arrays – Operações Básicas

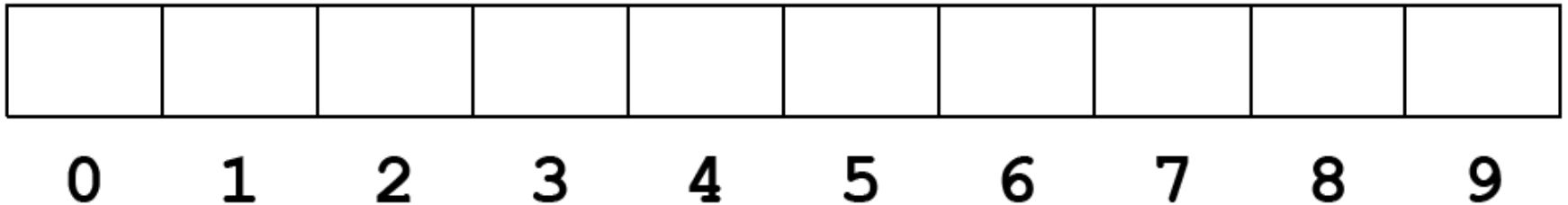
- Declarar
- Inicializar
- Imprimir



Arrays - Declarar

- Ex: Declaração de um vetor de 10 posições
`float vetor1[10];`

vetor1





Arrays - Inicializar

- Atribuindo valores o vetor

```
vetor1[0] = 15;
```

15									
0	1	2	3	4	5	6	7	8	9

```
vetor1[7] = 2.4;
```

15							2.4		
0	1	2	3	4	5	6	7	8	9

Arrays - Imprimir

- Imprimindo os elementos do vetor

15			2.4		3.5		2.4		
0	1	2	3	4	5	6	7	8	9

```
for (pos=3; pos<=7; pos++){  
    printf("Valor %.1f \n",vetor1[pos]);  
}
```

Tela

```
Valor 2.4  
Valor ?  
Valor 3.5  
Valor ?  
Valor 2.4
```

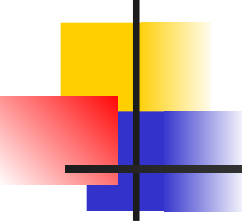
Valor de 'pos'

```
3  
4  
5  
6  
7
```




Estruturas

- Muitas vezes precisamos compor os dados para formar estruturas de dados complexas
- Variáveis compostas homogêneas (Arrays)
 - ✂ Conjunto de variáveis de mesmo tipo
- Variáveis compostas heterogêneas
 - ✂ Conjunto de variáveis de tipos diferentes
 - ✂ Chamadas de
 - **Estruturas (Struct)**
 - **Registros (Record)**



Estruturas – O que são?

- Variáveis compostas heterogêneas (estruturas) são um conjunto de variáveis de tipos diferentes que são logicamente relacionadas.
- Essas variáveis compartilham o mesmo identificador e ocupam posições consecutivas de memória



Estruturas – Exemplo 1

- Estruturas podem ser usadas para armazenar informações relacionadas
- Exemplo 1: Produto

Livro (char[11])	L	i	n	g	u	a	g	e	m		C
Preco (float)	59,9000										
Autor (char[11])	D	.		R	i	t	c	h	i	e	



Estruturas – Exemplo 2

- Exemplo 2: Ficha de cliente (cadastro)

Nome (char[10])	H	e	l	e	n	a				
Idade (int)	30									
Telefone (int)	5555-5555									
Cidade (char[10])	S	a	o		P	a	u	l	o	



Estruturas – Operações

- Definir
- Declarar
- Inicializar
- Imprimir



Estruturas - Definir

```
struct identificacao_da_estrutura {  
    tipo1 nome1;  
    tipo2 nome2;  
    ...  
    tipoN nomeN;  
};
```



Estruturas – Definir

```
struct sEndereco {  
    char rua[40];  
    int numero;  
    char cidade[30];  
    char estado[2];  
    long int CEP;  
};
```

- Este código deve vir no início do programa, após os “*includes*”



Estrutura – Declarar/Inicializar

```
int main( ){  
    //cria variavel ender1 como struct sEndereco  
    struct sEndereco ender1;  
  
    strcpy(ender1.rua, "Rua 7 de Setembro");  
    ender1.numero = 405;  
    strcpy(ender1.cidade, "Goiania");  
    strcpy(ender1.estado, "GO");  
    ender1.CEP = 06599604;  
}
```




Estrutura - Definir, Declarar, Inicializar, Imprimir

```
#include <stdio.h>
#include <stdlib.h>

struct sRetangulo{
    float altura, largura;
};

int main(void){
    struct sRetangulo ret1;

    ret1.altura = 10;
    printf("Digite o valor da largura: ");
    scanf("%f", &ret1.largura);


    printf("Altura: %.1f", ret1.altura);
    printf("Largura: %.1f", ret1.largura);

    return 0;
}
```



Estrutura - Definir

```
typedef struct identificacao_da_estrutura {  
    tipo1 nome1;  
    tipo2 nome2;  
    ...  
    tipoN nomeN;  
} id_da_estrutura;
```



Estrutura – Definir, Declarar, Inicializar, Imprimir

```
#include <stdio.h>

typedef struct Hora {
    int hora;
    int minuto;
    int segundo;
} THora;

int main(void) {
    THora H;
    H.hora = 10;
    H.minuto = 15;
    H.segundo = 30;
    printf("%d:%d:%d", H.hora, H.minuto, H.segundo);
    return 0;
}
```



Estruturas Aninhadas

- Estruturas em que um ou mais de seus membros também sejam estruturas.

```
typedef struct rotulo_estrutural {  
    tipo1 nome1;  
    tipoN nomeN;  
} id_estrutural;
```

```
typedef struct rotulo_estrutura2 {  
    id_estrutural nome;  
    tipoN nomeN;  
} id_estrutura2;
```



Estruturas Aninhadas - Exemplo

```
#include <stdio.h>
#include <string.h>

typedef struct sHora{
    int hora;
    int minuto;
    int segundo;
} Hora;

typedef struct sRelogio{
    Hora H;
    char modelo[10];
} Relogio;
```



Estruturas Aninhadas

```
#include <stdio.h>
#include <string.h>
```

```
typedef struct sHora{
    int hora;
    int minuto;
    int segundo;
} Hora;
```

```
typedef struct sRelogio{
    Hora H;
    char modelo[10];
} Relogio;
```

```
int main(void){
    Relogio r1;

    r1.H.hora = 10;
    r1.H.minuto = 15;
    r1.H.segundo = 30;

    strcpy(r1.modelo, "Cassio");

    printf("Modelo: %s\n", r1.modelo);
    printf("%d:%d:%d", r1.H.hora,
            r1.H.minuto,
            r1.H.segundo);

    return 0;
}
```



Arrays e Estruturas

- É possível combinar arrays e estruturas para criação de diferentes estruturas de dados.
- Podemos ter uma estrutura contendo um membro do tipo array, ou;
- Criar um array cujo os elementos sejam estruturas



Declarando Arrays de Estruturas

- → Dada a estrutura lista abaixo:

```
struct lista{  
    char titulo[30];  
    char autor[30];  
    int regnum;  
    double preco;  
};
```

- Declare um vetor com 50 elementos do tipo **lista**



Exercícios

1. Faça um programa para ler um vetor de 12 posições e em seguida ler também dois valores X e Y quaisquer correspondentes a duas posições no vetor. Ao final seu programa deverá escrever a soma dos valores encontrados nas respectivas posições X e Y
2. Leia um vetor de 16 posições e troque os 8 primeiros valores pelos 8 últimos e vice-versa. Escreva ao final o vetor obtido
3. Leia um vetor de 10 posições e em seguida um valor X qualquer. Seu programa deverá fazer uma busca do valor de X no vetor lido e informar a posição em que foi encontrado ou se não foi encontrado
4. Leia um vetor de 10 posições e em seguida um valor X qualquer. Seu programa deverá fazer uma busca do valor de X no vetor lido. Caso encontre o valor deverá ser removido do vetor.



Exercícios

5. Considere um cadastro de produtos de um estoque, com as seguintes informações para cada produto:
- Código de identificação do produto: representado por um valor inteiro
 - Nome do produto: com até 50 caracteres
 - Quantidade disponível no estoque: representado por um número inteiro
 - Preço de venda: representado por um valor real
- (a) Defina uma estrutura em C, denominada produto, que tenha os campos apropriados para guardar as informações de um produto
- (b) Crie um conjunto de 10 produtos e peça ao usuário para entrar com as informações de cada produto
- (c) Encontre o produto com o maior preço de venda
- (d) Encontre o produto com a maior quantidade disponível no estoque