

# ESTRUTURA DE DADOS

Prof.<sup>a</sup> Priscilla Abreu

[priscilla.braz@rj.senac.br](mailto:priscilla.braz@rj.senac.br)



# Estrutura de dados



## Roteiro de Aula

- Objetivo da aula
- Revisão
- Introdução a Estrutura de Dados
  - Tipos estruturados

# Estrutura de dados

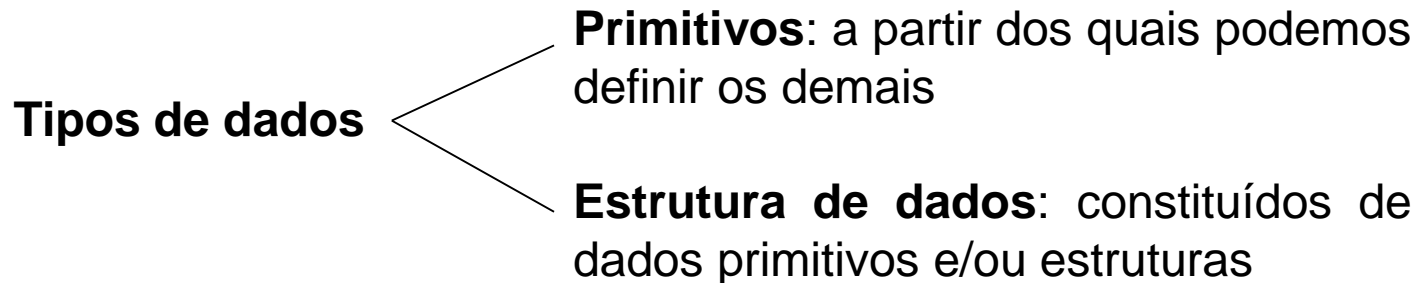


## Objetivo da aula

Revisar os conceitos básicos envolvendo o uso de estruturas heterogêneas.

# RELEMBRANDO...

## INTRODUÇÃO

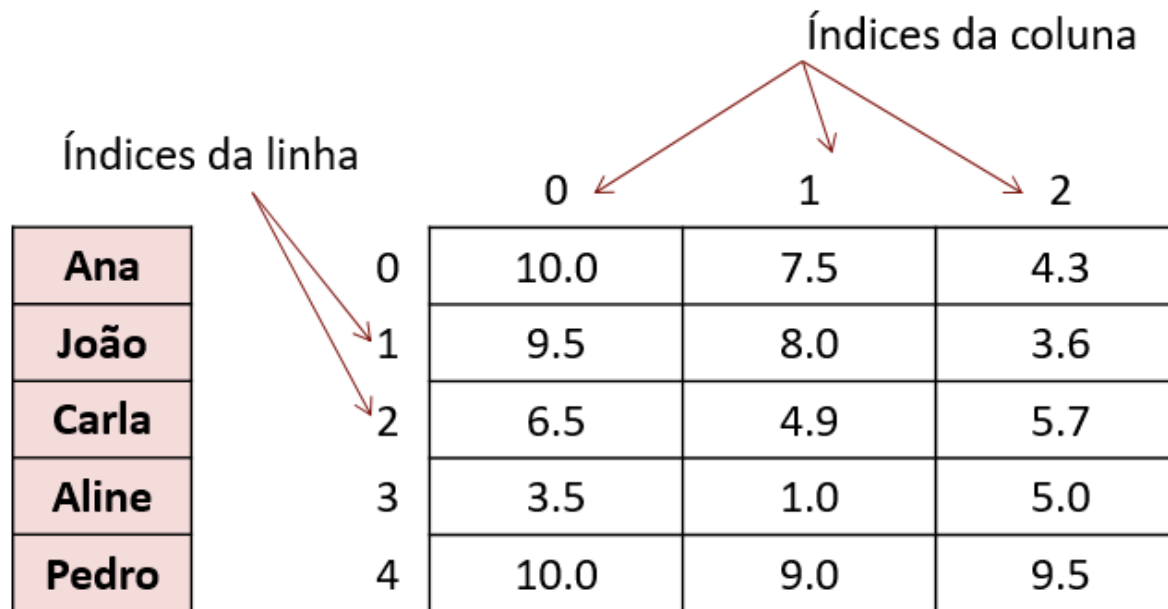


- Tipos primitivos
  - inteiro, real, lógico (boolean), caracter
- Estrutura de dados
  - Conjunto de informações agrupadas de uma forma coerente (com alguma relação entre elas)
    - Ex.: lista de chamada da turma.

# Estrutura de dados

## MATRIZES

Exemplo: Matriz bidimensional de notas de alunos durante um semestre.



		Índices da coluna		
		0	1	2
Índices da linha	0	10.0	7.5	4.3
	1	9.5	8.0	3.6
	2	6.5	4.9	5.7
	3	3.5	1.0	5.0
	4	10.0	9.0	9.5

# TIPOS ESTRUTURADOS HETEROGÊNEOS

## Definindo tipos de dados...

# Estrutura de dados



## TIPOS DE DADOS

**Imagine a seguinte situação:** Uma loja precisa manter os dados de seus produtos. Para cada produto ela deve manter:

- Código
- Nome
- Preço de compra
- Preço de venda
- Quantidade em estoque

**Como representar esses dados?**  
**Que estrutura pode ser utilizada?**



# Estrutura de dados



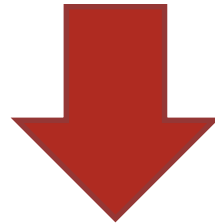
## TIPOS DE DADOS

**Vetores**

**Matrizes**



Estruturas que permitem armazenar diversos elementos, mas apenas elementos do mesmo tipo de dado.



**Tipos estruturados homogêneos**

# Estrutura de dados



## TIPOS DE DADOS

Estrutura de dados que permite agrupar variáveis de tipos de dados diferentes. Geralmente, os elementos de tal estrutura estão logicamente relacionados.

### Sintaxe:

```
struct nome_estrutura{  
    tipo1 var1;  
    tipo2 var2;  
    ...  
    tipon varn;  
} nome_estrutura;
```

# Estrutura de dados



## TIPOS DE DADOS

### Exemplo:

```
struct produto{  
    int cod, qtde;  
    char nome[50];  
    float pcompra, pvenda;  
} produto;
```

### Declaração:

```
struct produto prod;
```

# Estrutura de dados



## TIPOS DE DADOS

### Definição de tipo estrutura

#### Typedef

```
typedef struct produto{  
    int cod, qtde;  
    char nome[50];  
    float pcompra, pvenda;  
};
```

### Declaração:

```
produto prod;
```

# Estrutura de dados



## TIPOS DE DADOS

**Acesso aos campos de uma estrutura:**

`nomeEstrutura.nomeCampo`

**Exemplo:**

```
typedef struct produto{  
    int cod, qtde;  
    char nome[50];  
    float pcompra, pvenda;  
}produto;  
produto prod;
```

```
prod.cod;  
prod.pcompra;
```

# Estrutura de dados



## TIPOS DE DADOS

### Exemplo de leitura e impressão de dados

```
typedef struct produto{  
    int cod, qtde;  
    char nome[50];  
    float pcompra, pvenda;  
} produto;  
produto prod;  
  
scanf("%f",&prod.pcompra);  
printf("Preço de compra: %f", prod.pcompra);
```

# Estrutura de dados



## TIPOS DE DADOS – EXEMPLO

Faça um programa que utilize uma estrutura para armazenar os dados de um produto. Os dados que devem ser armazenados são: código, nome, quantidade em estoque, preço de compra e preço de venda. Leia os dados de um produto e imprima as informações.

# Estrutura de dados



## TIPOS DE DADOS – EXEMPLO

```
#include <stdio.h>
typedef struct produto{
    int cod, qtde;
    char nome[50];
    float pcompra, pvenda;
}produto;
```



# Estrutura de dados



## TIPOS DE DADOS – EXEMPLO

```
int main(){
    produto prod;
    printf("Informe os dados do produto: \n");
    printf("Código: ");
    scanf("%d", &prod.cod);
    printf("Nome: ");
    fflush(stdin);  fgets(prod.nome, 50, stdin);
    printf("Quantidade em estoque: \n");
    scanf("%d",&prod.qtde);
    printf("Preço de compra: \n");
    scanf("%f",&prod.pcompra);
```

## TIPOS DE DADOS – EXEMPLO

```
printf("Preço de venda: \n");  
scanf("%f", &prod.pvenda);  
printf("Impressão... \n");  
printf("Código: %d\n", prod.cod);  
printf("Nome: %s", prod.nome);  
printf("Quantidade em estoque: %d\n", prod.qtde);  
printf("Preço de compra %.2f\n", prod.pcompra);  
printf("Preço de venda: %.2f\n", prod.pvenda);  
}
```

# Estrutura de dados



## TIPOS DE DADOS

**Continuando no exercício anterior...**

**Uma loja não tem um único produto!!**

**É preciso cadastrar vários produtos e não apenas UM!**

**O que fazer???**

# Estrutura de dados



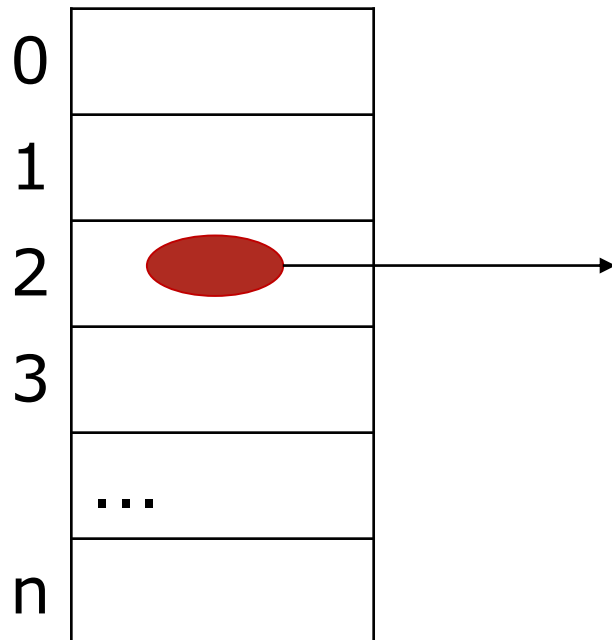
## TIPOS DE DADOS

Para armazenar os dados de vários produtos, a solução será utilizar vetor de registros;

Em cada posição do vetor haverá um registro com seus respectivos elementos.

# Estrutura de dados

## TIPOS DE DADOS



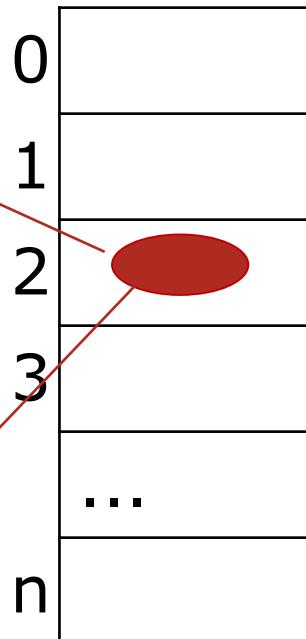
Produto	
Código	Nome
Preço de compra	Preço de venda
Quantidade em estoque	

# Estrutura de dados

## TIPOS DE DADOS

Produto	
Código	Nome
Preço de compra	Preço de venda
Quantidade em estoque	

prod[2].cod  
prod[2].qtde  
prod[2].nome  
prod[2].pcompra  
prod[2].pvenda



```
typedef struct produto{  
    int cod, qtde;  
    char nome[50];  
    float pcompra, pvenda;  
} produto;  
  
int main(){  
    produto prod[50];  
}
```

# Estrutura de dados



## TIPOS DE DADOS – EXEMPLO

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

```
typedef struct produto{  
    int cod, qtde;  
    char nome[50];  
    float pcompra, pvenda;  
};
```

# Estrutura de dados



## TIPOS DE DADOS – EXEMPLO

```
int main(){
    produto prod[5];
    int i;
    for (i=0; i<5;i++){
        printf("Código: ");
        scanf("%d", &prod[i].cod);
        printf("Nome: ");
        fflush(stdin);  fgets(prod[i].nome, 50, stdin);
        printf("Quantidade em estoque: \n");
        scanf("%d",&prod[i].qtde);
    }
}
```



## TIPOS DE DADOS – EXEMPLO

```
printf("Preço de compra: \n");  
scanf("%f",&prod[i].pcompra);  
printf("Preço de venda: \n");  
scanf("%f", &prod[i].pvenda);  
}
```

## TIPOS DE DADOS – EXEMPLO

```
printf("Impressão... \n\n");
for (i=0;i<5;i++){
    printf("Código: %d", prod[i].cod);
    printf("\nNome: "); puts(prod[i].nome);
    printf("Quantidade: %d \n", prod[i].qtde);
    printf("Preço    de    compra:    %5.1f\n",
prod[i].pcompra);
    printf("Preço    de    venda:    %5.1f\n",
prod[i].pvenda);
}
}
```

# Estrutura de dados



## TIPOS DE DADOS – Exercício

Em uma escola é necessário armazenar os dados de diversos alunos. Suponha uma turma com 20 alunos e que os dados de cadastro desejados são: nome, matrícula, quatro notas e a média final. Faça um programa para armazenar esses dados e imprimir a média de cada aluno.

Nome	Matrícula	Nota				Média
		1	2	3	4	

# Estrutura de dados



## TIPOS DE DADOS – Registros

Com registros podemos agrupar dados logicamente relacionados, mas que podem possuir tipos de dados diferentes.

Assim, temos a possibilidade de representar os dados de forma mais ampla, melhorando a qualidade dos nossos algoritmos.

# **FUNÇÕES E PROCEDIMENTOS (REVISÃO)**

# Introdução à Programação



## REVISANDO...

O QUE SÃO:

- FUNÇÕES E PROCEDIMENTOS?
- PARÂMETROS?
- PASSAGEM POR VALOR?
- PASSAGEM POR REFERÊNCIA?

# Introdução à Programação



## EXERCÍCIO

Faça programa que leia dez números inteiros positivos e informe, para cada um deles, os seus divisores. Para isso, você deve implementar um(a) procedimento ou função que recebe, por parâmetro, um valor inteiro e positivo e apresenta esses divisores do valor.

# Introdução à Programação



## EXERCÍCIO

Faça um programa que leia dois valores positivos ( $a$ ,  $b$ ) e calcule e exiba a potência  $a^b$ .

Você deve fazer a leitura dos valores utilizando um procedimento chamado “leitura” e a potência deve ser calculada por uma função “calcular\_potencia”, que recebe os dois valores por parâmetro e retorna o resultado da potência, **sem o uso de funções da biblioteca <math.h>**.

O procedimento de leitura e a função da potência devem ser chamados na função main, que exibirá os dados.

**Utilize apenas variáveis locais!**



# DÚVIDAS?