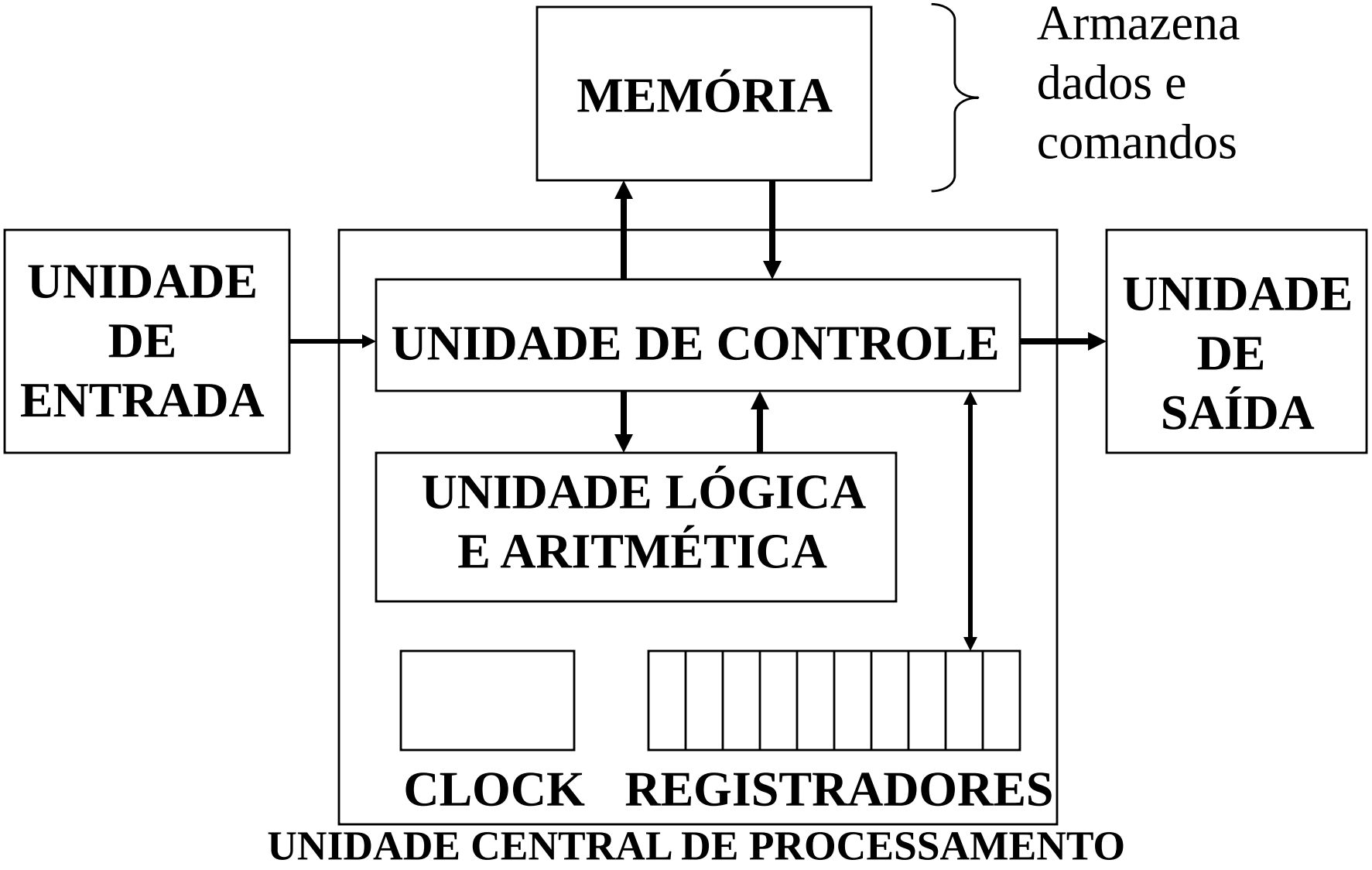
# Introdução à Programação Estruturada Parte 3

Material da Prof. Ana Eliza

# Arquitetura do Computador Digital

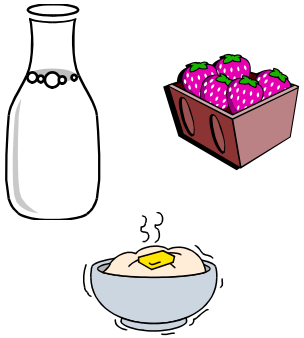


**Dados e comandos, para serem processados, devem estar na memória do computador.**

# Variáveis

↳Exemplo: Receita de Bolo

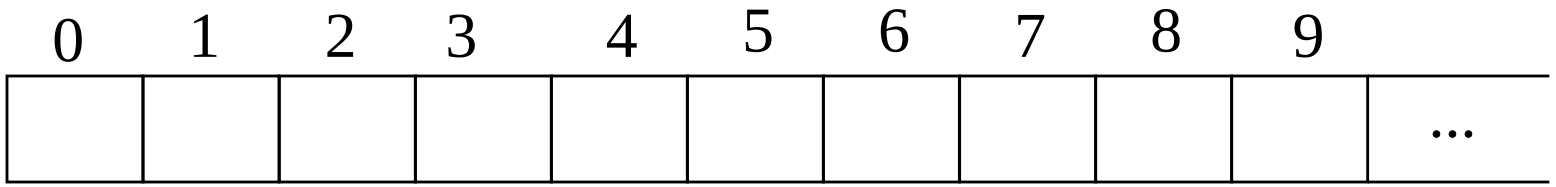
| Dados Iniciais | Algoritmo | Resultado Final |
|---|---|---|
| (***Ingredientes***) | (***Modo de Fazer***) | (***Bolo***) |



**Variáveis** ⇨ "*Recipientes para armazenar dados*"

# Memória

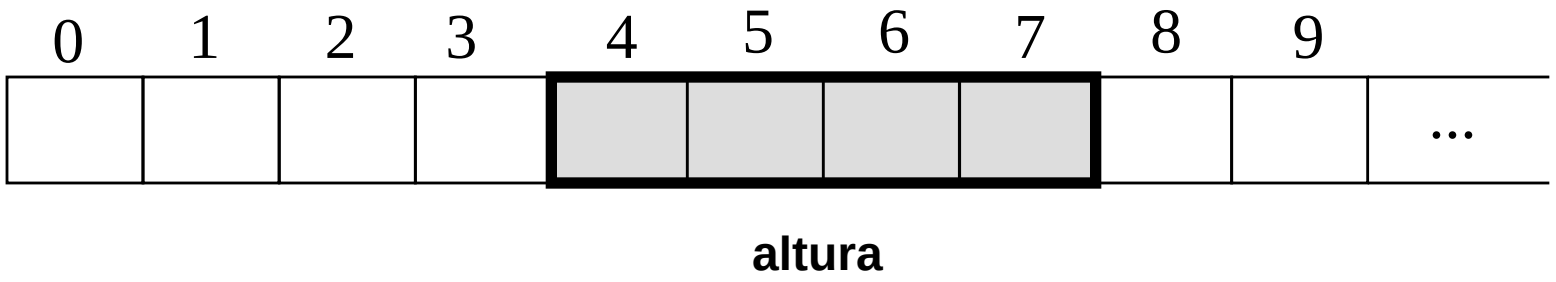
- **Definição**:
  - Uma **memória** é uma **seqüência de células de armazenamento**;



# Variáveis

## Definição:

- Uma **variável** é um "***container***" composto de uma ou mais células de armazenamento (células de memória);



# Variáveis

- **Utilização:**
  - Variáveis são seqüências de células de memória utilizadas para armazenar os dados que são manipulados por um programa.

# Variáveis

- **Tipos de Variáveis:**
  - Variáveis Simples
    - Armazenam um único valor;
  - Variáveis Compostas
    - Armazenam mais de um valor;

# Variáveis

- **Atributos de uma Variável**:
  - **nome**: seqüência de caracteres utilizada para identificar a variável;
  - **tipo**: é o tipo dos dados que serão armazenados na variável;
  - **conteúdo**: é o valor armazenado na variável.
  - **endereço**: é a localização (posição) da variável na memória.

# Tipos de Dados

- O que são tipos de dados?

  Conjunto de Valores

  +

  Conjunto de Operações

- Ou seja, o tipo de dados de uma variável determina que valores podem ser armazenados nela e que operações podem ser realizadas com esses valores.

# Tipos de Dados Simples

- Tipos de dados simples da linguagem C:
  - ***int*** ⇨ Números inteiros
    - Conjunto de valores: -2.147.483.648 ... 2.147.483.647
  - ***float*** ⇨ Números reais
    - Conjunto de valores: valores com 06 (seis) dígitos de precisão
  - ***char*** ⇨ Caracteres
    - Conjunto de valores: ‘0’...’9’,‘A’...’Z’,’a’...’z’, ‘!’, ‘?’, ‘@’, ‘#’, ...

# Declaração de Variáveis

- A declaração de uma nova variável provoca:
  - A alocação (“reserva”) de uma seqüência de células de memória em quantidade suficiente para armazenar o **tipo** de dados declarado;
  - A associação do **nome** dado na declaração ao bloco de memória alocado.

# Declaração de Variáveis

- **Sintaxe**

  tipoDeDados  nomeDaVariável;

- **Exemplos**

  **int** idade;

  **float** altura, peso;

  **char** sexo;

# Nomes de Variáveis

- Regras (Linguagem **C**):
  - ✓ O primeiro caracter deve ser uma letra;
  - ✓ Os caracteres seguintes devem ser letras, dígitos ou o caracter *underline* ( _ );
  - ✓ Não deve haver espaços em nomes de variáveis;
  - ✓ Não deve haver símbolos de pontuação em nomes de variáveis.

# Nomes de Variáveis

## Exemplos

- ✓ Nomes válidos:

  idade
  salario
  fone_comercial
  email2

- ✓ Nomes inválidos:

  2prova
  preço
  salário
  ano atual
  valor$

# Nomes de Variáveis

- Importante:
  - ✓ A linguagem C é "***case sensitive***", ou seja, faz diferença entre caracteres maiúsculos e minúsculos.
- Exemplos:
  - ✓ **Idade ≠ idade ≠ IDADE**
  - ✓ **salario ≠ Salario**

# Comando de Atribuição

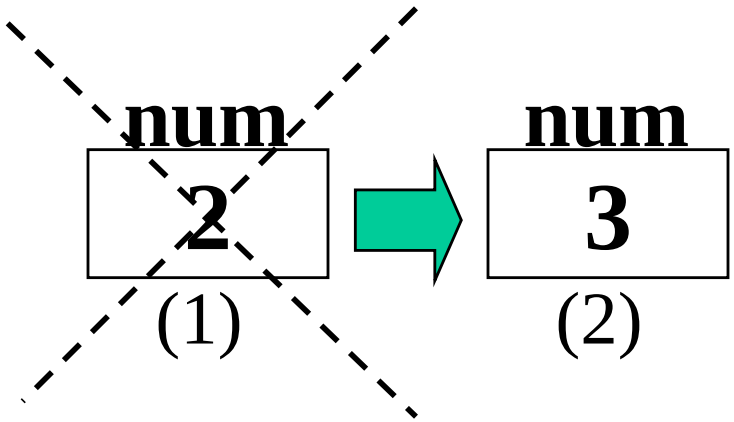
- **Atribuição** significa **armazenamento**
- O comando de atribuição ( = ) é utilizado para armazenar uma informação em uma variável.
- Modo de Funcionamento do Comando:
  - Avalia a expressão situada no lado direito do comando de atribuição;
  - Armazena o valor resultante na variável situada do lado esquerdo do comando de atribuição.

# Comando de Atribuição

- **Exemplos (C/C++):**

(1) num = 2;

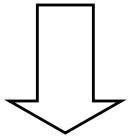
(2) num = num + 1;



# Comandos de Entrada de Dados

- **Comando *scanf***
  - É utilizado para comunicação entre o usuário e o programa;
  - Permite que o usuário forneça dados ao programa.
  - **Formato**:

    **scanf(“%**tipo_da_variável**”,&**nome_da_variável);

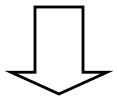


*“Armazena o dado digitado pelo usuário do programa na variável indicada.”*

# Comandos de Entrada de Dados

- **Exemplo**:

  **scanf**(“**%**f”,**&**altura);

  “*Armazena o valor digitado pelo usuário do programa na variável* ***altura****.”*

- **OBS:**

  %i ou %d => int

  %f => float

  %c => char

# Comandos de Entrada de Dados

- **Exemplos:**

```
scanf ("%i",&idade);
scanf ("%f",&altura);
scanf ("%c",&sexo);
```

# Comandos de Saída de Dados

- **Comando *printf***
  - São utilizados para comunicação entre o programa e o usuário;
  - Permite que o programa forneça informações ao usuário.
  - **Formato**:

    **printf**(“**%**tipoDaVariável”,nomeDaVariável);

    **printf** (“Seqüência de caracteres”);

    **printf** (“Seqüência de caracteres”, listaDeVariáveis);

# Comandos de Saída de Dados

## Exemplos:

**printf** (“%i”,idade);

**printf** (“Qual é a sua idade?”);

**printf** (“A idade de Maria é %i”,idade);

**printf** (“A idade de Maria é %i e ela tem %f de altura”,idade,altura);

# Estrutura Básica de um Programa em Linguagem C

```
Cabeçalho (bibliotecas)

int main ()
{
   declaraçõeS de variáveis

   lista de comandos
}
```

# Exemplo de um Programa Simples em Linguagem C

```
#include <stdio.h>          ⇨ Inclusão de biblioteca
int main ( )
{
   int num1, num2, soma;    ⇨ Declaração de variáveis
   printf (“Digite um número inteiro: ”);
   scanf (“%i”,&num1);
   printf (“Digite outro número inteiro : ”);
   scanf (“%i”,&num2);
   soma = num1 + num2;
   printf(“A soma de %i e %i eh %i. “, num1,num2,soma);
}
```

# A Biblioteca stdio.h

- “stdio” vem de “**st**andar**d i**nput/**o**utput”;
- A biblioteca stdio.h contém as funções de entrada e saída de C padrão (ANSI C);
- Exemplos de funções da biblioteca “stdio.h”:
  - scanf
  - printf

# Operadores Aritméticos

+ : Adição

- : Subtração

* : Multiplicação

/ : Divisão

% : Resto da divisão de inteiros

# Funções Matemática

- **Biblioteca da funções matemática**

  ***math.h***

- O padrão ANSI C define 22 funções matemáticas que se dividem nas seguintes categorias:
  - Funções trigonométricas
  - Funções hiperbólicas
  - Funções exponenciais e logarítmicas
  - Funções diversas

# Funções Matemática

- **Exemplos de funções matemática**

**pow (x,y)**: Calcula $x^y$

**sqrt (x)**: Calcula a raiz quadrada de x

**ceil (x)**: Informa o menor inteiro maior que x

**floor (x)**: Informa o maior inteiro menor que x

# Funções Aritméticas

- **Biblioteca**

  ***math.h***

  Função **floor (x)** : Informa o maior valor inteiro menor que x

  Exemplo:

  float x,y,z,k;

  ...

  x = 7.8;

  y = floor (x); ⇨ o valor de y será 7

  z = -7.8;

  k = floor(z); ⇨ o valor de k será -8

# Funções Aritméticas

- **Biblioteca**

  ***math.h***

  Função **ceil (x)** : Informa o menor valor inteiro maior que x

  Exemplo:

  float x,y,z,k;

  ...

  x = 7.3;

  y = ceil (x); ⇨ o valor de y será 8

  z = -7.8;

  k = ceil (z); ⇨ o valor de k será -7

# Expressões Aritméticas

- **Ordem de Prioridade**

| | **Prioridade** |
|---|---|
| Parênteses dos mais internos para os mais externos | 1 - Maior |
| Funções matemáticas | 2 |
| *, / , % | 3 |
| +, - | 4 - Menor |

# Expressões Aritméticas

- **OBSERVAÇÕES**:
  - Os critérios de prioridade são seguidos no cálculo de uma expressão aritmética;
  - Operadores de mesma prioridade ⇨ a avaliação é feita da esquerda para a direita.