Aula 2

# Lógica de Programação de Algoritmos

Prof. Sandro de Araújo

## Conversa Inicial

- Conceitos básicos de algoritmos – dados, tipos de dados, variáveis, constantes, operadores

- Esta aula apresenta a seguinte estrutura de conteúdo:

1. Definição de dados
2. Tipos de dados
3. Variáveis
4. Constantes
5. Operadores

- O objetivo desta aula é apresentar os principais conceitos de dados, tipos de dados, variáveis, constantes e operadores de atribuição, aritméticos, relacionais e lógicos

## Definição de dados

- O dado é uma sequência de símbolos quantificados ou quantificáveis
- São valores fornecidos pelo usuário do programa, podendo ser obtidos com base em processamentos, arquivos, banco de dados ou outros programas

- Para garantir a integridade do resultado obtido com processamento, os dados devem ser classificados de acordo com o tipo do valor a ser armazenado na variável, evitando problemas ocasionados pelo fornecimento de valores inadequados à operação realizada
- Número inteiro com inteiro
- Número flutuante com flutuante

Tipos de dados

- Ao desenvolver um algoritmo, é necessário que se tenha conhecimento prévio do tipo de dado que será utilizado para resolver o problema proposto. Então, escolhe-se o tipo de dado adequado para a variável que representará esse valor

- Alguns tipos são formados por números inteiros e reais que suportam operações matemáticas como adição, subtração, multiplicação, entre outros. Esses tipos são particularmente importantes, visto que o computador trabalha naturalmente com números

- Também podemos considerar as letras como um tipo de dado em que poderíamos definir operações como escrever, ler, concatenar, entre outros

- A maioria das linguagens de programação tipifica os dados em um grupo chamado de tipos primitivos:
  - Numéricos
  - Lógicos
  - Literais ou caracteres

- Tipos primitivos numéricos:
  - Dividem-se em dois grupos: inteiros e reais
    - Os inteiros:
      - Positivos e negativos
      - Não apresentam parte fracionária
      - Exemplo: -357, -23, 0, 98, 237

- Tipos primitivos numéricos:
  - Dividem-se em dois grupos: inteiros e reais
  - Os reais:
    - Positivos e negativos
    - Apresentam parte fracionária
    - Exemplo: - 23.45, -5.6, 0.0, 32.55, 222.02

- Tipos primitivo lógicos:
  - São também chamados de booleanos e podem assumir os valores verdadeiro ou falso – o número 0 para falso e 1 para verdadeiro

- Tipos primitivos literais ou caracteres:
  - São dados formados por um único caractere ou por uma cadeia de caracteres. Esses caracteres podem ser:
    - O alfabeto, com maiúsculos e minúsculos
    - Os números, que não poderão ser usados para cálculos, pois não são valores
    - Os caracteres especiais, como @,#,$,?, +

Variáveis

## Variável

- Uma variável pode ser entendida como uma
  - Posição identificada na memória:
    - Que contém dados
    - Que pode ser modificada durante a execução do programa
    - Pode assumir qualquer valor de um conjunto de valores

## Exemplos de variáveis

- A altura de uma pessoa
- A cotação do bitcoin
- A velocidade de um carro
- Nesses exemplos, os valores dos dados sofrem alterações ou são dependentes da execução em certo instante ou circunstância

## Identificação das variáveis

- Toda variável deve receber um nome ou identificador e estar de acordo com algumas regras:
  - Não utilizar espaços entre as letras
    - Por exemplo, em vez de nome do cliente, o correto seria nome_do_cliente ou nomeDoCliente
    - O caractere "*underline*", representado por "_", pode ser utilizado para substituir o espaço entre as letras

## Identificação das variáveis

- Não iniciar o nome da variável com algarismos (números)
  - Por exemplo: não usar 2valor; o correto seria valor2

## Identificação das variáveis

- Não utilizar palavras reservadas, isto é, palavras que são utilizadas nos algoritmos para representar ações específicas. Por exemplo:
  - se (if na linguagem C) é uma palavra usada para representar uma condição ou teste lógico

## Identificação das variáveis

- var: palavra para representar a área de declaração de variáveis (em pseudocódigo)
  - Cada linguagem de programação tem sua sintaxe para declaração de variável

## Identificação das variáveis

- **Exemplo: na linguagem de programação C devemos listar primeiro o tipo, depois o nome da variável**

| Sintaxe em pseudocódigo | Sintaxe em linguagem C |
|---|---|
| <indicador de variáveis><br>var<br><nome_da_variável> : <tipo><br>matricula_aluno : inteiro | <tipo><br><nome_da_variável>;<br>int matricula_do_aluno; |
| Onde:<br>• var Indica onde as variáveis serão declaradas<br>• matricula_do_aluno é o nome da variável<br>• inteiro é o tipo da variável | Onde:<br>• int é o tipo da variável (inteiro)<br>• matricula_do_aluno é o nome da variável |

## Identificação das variáveis

- **Não utilizar caracteres especiais, como acentos, símbolos (?, /, :, @, # etc.), ç, entre outros**
  - **Exemplo:**
    - **m@tricula, #nome_aluno, $serviço, numeração, entre outros**

## Identificação das variáveis

- **Não utilizar nomes iguais para representar variáveis diferentes**
  - **Utilizar duas variáveis com o mesmo nome em um mesmo escopo não é possível**

## Identificação das variáveis

- **Ser conciso e utilizar nomes coerentes**
  - **Vou criar uma variável para guardar a matrícula do aluno e denominei como:**
    - **altd3241 : interio**
  - **altd3241 não faz sentido para quem vai ler o código**
  - **Já matricula_do_aluno faz todo o sentido para quem o lê**

## Constantes

## Constante

- Uma constante segue as mesmas regras de variável, mas com a certeza de que o dado ou valor não será alterado durante a execução do programa e de que será sempre o mesmo, sendo obrigatória a atribuição de um valor no momento da declaração

## Constante

- Um exemplo de uma constante matemática é o número PI, que é um valor fixo de aproximadamente 3,1415 , e que continuará assim até o final da execução

## Operadores

## Operadores

- Os operadores são utilizados para representar expressões de cálculo, comparação, condição e atribuição. Para a construção de algoritmos temos os seguintes tipos de operadores:
  - De atribuição
  - Aritméticos
  - Relacionais
  - Lógicos

## Operadores de atribuição

- Um dos operadores mais utilizados na programação é o operador de atribuição, representado no pseudocódigo pela seta ←
  - nomeDaVariavel ← expressão

## Operadores de atribuição

- Exemplos:
  - nomeDoCliente ← “Joãozinho da Silva”
  - resultado ← a + 5
  - valor ← 3.5

## Operadores de atribuição

- Para a linguagem C, o sinal de atribuição é representado pelo "=",
  nomeDaVariavel = expressão
  - Exemplos:
    - nomeDoCliente = "Joãozinho da Silva"
    - resultado = a + 5
    - valor = 3.5

## Operadores de atribuição

- Chamamos de operadores aritméticos o conjunto de símbolos que representa as operações básicas da matemática

| OPERADOR | NOTAÇÃO ALGORÍTMICA |
|---|---|
| Incremento | Utiliza-se uma expressão. Exemplo: a ← a + 1 |
| Decremento | Utiliza-se uma expressão. Exemplo: a ← a - 1 |
| Adição | + |
| Subtração | - |
| Multiplicação | * |
| Divisão | / |
| Exponenciação | ^ ou ** Exemplo: pot ← 2**3 ou 2^3, raiz ← 4**(½) ou raiz ← 4^(½) |
| Módulo | Mod Exemplo: resto ← a mod b |

## Operadores lógicos

- Chamamos de operadores aritméticos o conjunto de símbolos que representa as operações básicas da matemática

| Operador | Representação algorítmica | Notação para C | Descrição para linguagem C |
|---|---|---|---|
| E | .e. | && | Realiza a operação E, exemplo: (x >= 0 && x <= 8) |
| OU | .ou. | \|\| | Realiza a operação OU, exemplo: (a == 'G' \|\| b != 33) |
| NÃO | .não. | ! | Realiza a operação OU, exemplo: !(x == 11) |

## Operadores lógicos

- Como resultado, dessas operações teremos como retorno:
  - O valor UM (1), se a expressão relacional for verdadeira
  - o valor ZERO (0), se a expressão relacional for falsa

## Operadores lógicos

- **Na tabela verdade é expresso o conjunto de possibilidades existentes para a combinação de variáveis ou expressões e operadores lógicos**

| a | b | a .e. b (a && b) | a .ou. b (a \| \| b) | .não. a |
|---|---|---|---|---|
| falso | falso | falso | falso | verdadeiro |
| falso | verdadeiro | falso | verdadeiro | verdadeiro |
| verdadeiro | falso | falso | verdadeiro | falso |
| verdadeiro | verdadeiro | verdadeiro | verdadeiro | falso |


- **GUEDES, S. Lógica de programação algorítmica. São Paulo: Pearson, 2014.**
- **MEDINA, M.; FERTING, C. Algoritmos e programação: teoria e prática. São Paulo: Novatec, 2006.**
- **PUGA, S.; RISSETTI, G. Lógica de programação e estruturas de dados. São Paulo: Pearson, 2016.**