Pontifícia Universidade Católica
do Rio de Janeiro

Introdução a Computação

Aula 02 – Lógica de Programação

Edirlei Soares de Lima
<elima@inf.puc-rio.br>

visionlab
visualization laboratory

Lógica de Programação

- **Lógica de Programação** é a técnica de criar sequências lógicas de ações para atingir determinados objetivos.
- **Sequências Lógicas** são instruções executadas para atingir um objetivo ou solução de um problema.
- **Instruções** são uma forma de indicar ao computador uma ação elementar a executar.
- Um **Algoritmo** é formalmente uma sequência finita de instruções que levam a execução de uma tarefa.

Lógica de Programação

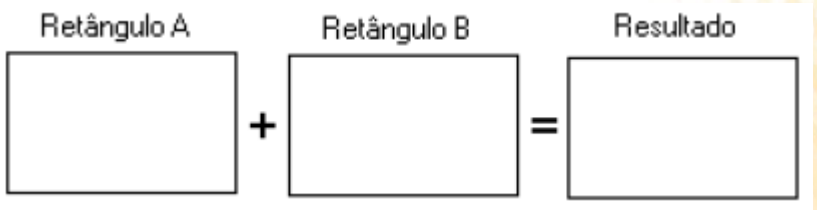
- Até mesmo tarefas simples, podem ser descritas por **sequências lógicas**:

"Chupar uma bala"

1) Pegar a bala;
2) Retirar o papel;
3) Chupar a bala;
4) Jogar o papel no lixo;

"Somar dois números"

1) Escreva o primeiro número no retângulo A.
2) Escreva o segundo número no retângulo B.
3) Some o número do retângulo A com número do retângulo B e coloque o resultado no retângulo C.



visionlab
visualization laboratory

Escrevendo Algoritmos

- Os algoritmos podem ser escritos diretamente em uma linguagem de programação ou simplesmente descritos em pseudocódigo.

- **Pseudocódigo** é uma forma genérica de escrever um algoritmo.

- **Linguagens de programação** são formas padronizadas de comunicar instruções para o computador. São conjuntos de regras sintáticas e semânticas usadas para definir um programa de computador.

Escrevendo Algoritmos

- **Exemplo:** “Ler duas notas e calcular a média”

Pseudocódigo:

```
variáveis
    nota1, nota2, media : real;

início
    leia(nota1);
    leia(nota2);
    media = (nota1 + nota2)/2;
    escreva(media);
fim
```

Linguagem C:

```
float nota1, nota2, media;

void main()
{
    scanf("%f", &nota1);
    scanf("%f", &nota2);
    media = (nota1 + nota2)/2;
    printf("A média é: %f", media);
}
```

visionlab
visualization laboratory

Pseudocódigo

- **Estrutura** de um programa descrito em pseudocódigo:

```
constantes
   <nome> = <valor>;
   …
variáveis
   <nome> : <tipo>;
   …
início
   <comando>;
   …
fim
```

Exemplo:

```
variáveis
     nota1, nota2, media : real;

início
     leia(nota1);
     leia(nota2);
     media = (nota1 + nota2)/2;
     escreva(media);
fim
```

visionlab
visualization laboratory

Escrevendo Algoritmos

Processo Geral de um Algoritmo

- **Entrada**: O algoritmo recebe os dados de entrada.
- **Processamento**: Os procedimentos para se chegar ao resultado final são executados.
- **Saída**: O resultado final é mostrado.

Variáveis e Constantes

- **Variáveis** e **constantes** são os elementos básicos manipulados por um programa.

- **Constante** é um valor fixo que não se modifica ao longo da execução de um programa.

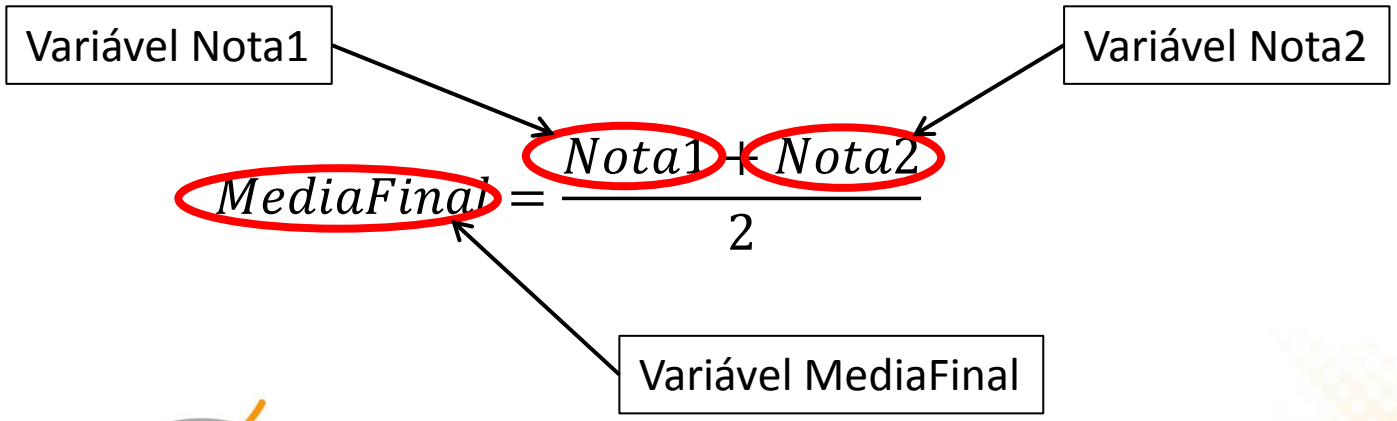
$$MediaFinal = \frac{Nota1 + Nota2}{2}$$

valor constante

Variáveis

- **Variável** é um espaço reservado na memória do computador para armazenar um determinado tipo de dado.

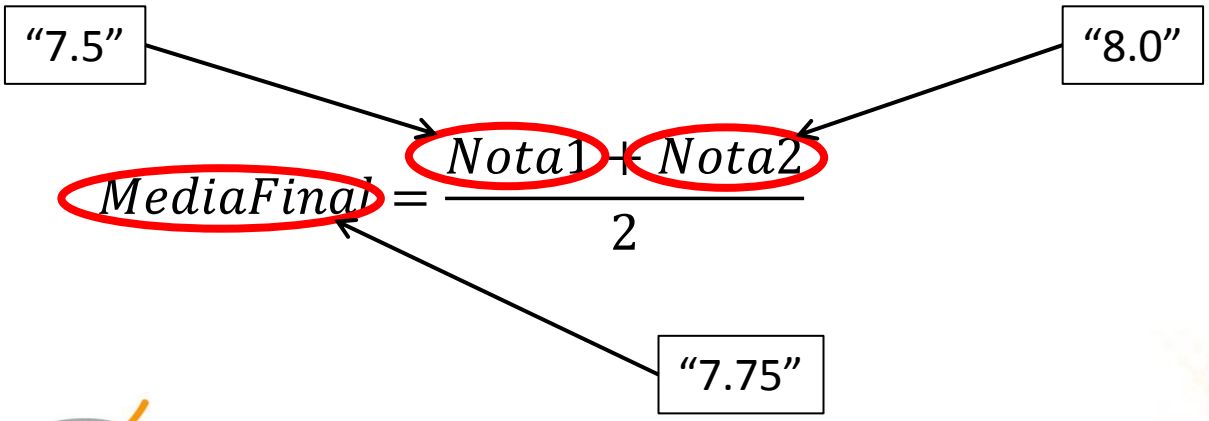
- Variáveis recebem **nomes** para serem referenciadas e modificadas quando necessário.



Variáveis

- O **conteúdo de uma variável** pode se modificado ao longo da a execução do programa.

- Embora uma variável possa assumir diferentes valores, ela só pode armazenar **um valor a cada instante**.



Tipos de Variáveis

- As variáveis e constantes podem ser de três **tipos**:
  - **Numéricas**: Armazenam números. Podendo ser ainda divididas em dois tipos:
    - **Inteiras:** Armazenam números inteiros.
      - Exemplos: 2, 8, 50, 4812, 100...
    - **Reais:** Armazenam números com casas decimais.
      - Exemplos: 0.54, 0.2, 15.84, 0.000032...
  - **Conjuntos de Caracteres**: Armazenam conjuntos de caracteres.
    - Exemplos: “João”, “teste 123 teste”, “isso é uma variável”...
  - **Lógicas:** Armazenam dados lógicos (verdadeiro ou falso)

Entrada e Saída de Dados

- **Comando de Leitura**

Sintaxe: `leia(<var1>,<var2>,…,<varN>);`
Exemplo: `leia(idade);`

- **Comando de Escrita**

Sintaxe: `escreva(<exp1>,<exp2>,…,<expN>);`
Exemplo: `escreva("a idade é: ", idade);`

Operadores Aritméticos

- É possível a utilização de **operadores aritméticos** para se realizar operações aritméticas com as variáveis e constantes.

Operação	Símbolo
Adição	+
Subtração	-
Multiplicação	*
Divisão	/

Exemplos:

total = preco * quantidade
media = (nota2 + nota2)/2
resultado = 3 * (1 - 2) + 4 * 2

Exemplo 01

- “Escreva o pseudocódigo de um algoritmo que recebe um valor inteiro, soma 2 a este valor, e exibe o resultado”.

```
variáveis
   valor, resposta : inteiro;

início
    escreva(“Digite um numero:”);
    leia(valor);
    resposta = valor + 2;
    escreva(“Resultado: ”, resposta);
fim
```

Exercícios

Lista 01

- http://www.inf.puc-rio.br/~elima/intro-prog/

Operadores Relacionais

- Os **operadores relacionais** são usados para relacionar duas expressões. O valor desta comparação é sempre um valor lógico (**verdadeiro** ou **falso**).

Descrição	Símbolo
Igual a	==
Diferente de	!=
Maior que	>
Menor que	<
Maior ou igual a	>=
Menor ou igual a	<=

Exemplos:

X = 10 e Y = 5

Expressão	Resultado
(X == Y)	Falso
(X != Y)	Verdadeiro
(X > Y)	Verdadeiro
(X < Y)	Falso
(X >= Y)	Verdadeiro
(X <= Y)	Falso

Operadores Lógicos

- **Operadores lógicos** são utilizados para combinar resultados de expressões.

Operador
AND
OR
NOT

Exemplos:

X = 10 e Y = 5

Expressão	Resultado
(X > 0) AND (X == Y)	Falso
(X > 0) OR (X == Y)	Verdadeiro
NOT (Y < 10)	Falso

Estruturas Condicionais

- **Estruturas condicionais** permitem a criação de algoritmos que não são totalmente sequenciais.

- Com o uso de estruturas condicionais é possível criar **regras** que definem quando uma determinada parte do algoritmo deve ser executada.

```
SE <expressão lógica> ENTÃO
Inicio
    <comando>;
Fim
```

```
SE <expressão lógica> ENTÃO
Inicio
    <comando>;
Fim
SENÃO
Inicio
    <comando>;
Fim
```

Estruturas Condicionais

Exemplo1:

```
variáveis
    nota1, nota2, media : real;

início
    leia(nota1);
    leia(nota2);
    media = (nota1 + nota2)/2;
    SE (media >= 7.0) ENTÃO
        escreva("Aprovado!");
fim
```

Exemplo2:

```
variáveis
    nota1, nota2, media : real;

início
    leia(nota1);
    leia(nota2);
    media = (nota1 + nota2)/2;
    SE (media >= 7.0) ENTÃO
        escreva("Aprovado!");
    SENÃO
        escreva("Reprovado!");
fim
```

Estruturas Condicionais

- "Escreva o pseudocódigo de um algoritmo que recebe três números inteiros e os exiba em ordem crescente".

```
variáveis
    valor1, valor2, valor3 : inteiro;
início
    leia(valor1);
    leia(valor2);
    leia(valor3);
    SE (valor1 < valor2)AND(valor1 < valor3) ENTÃO
    início
      SE (valor2 < valor3) ENTÃO
        escreva(valor1,valor2,valor3);
      SENAO
        escreva(valor1,valor3,valor2);
    fim
    SENAO SE (valor2 < valor1)AND(valor2 < valor3) ENTÃO
    inicio
    .
    .
```

Estruturas Condicionais

```
    .
    .
      SE (valor1 < valor3) ENTÃO
        escreva(valor2,valor1,valor3);
      SENAO
        escreva(valor2,valor3,valor1);
    fim
    SENAO SE (valor3 < valor1)AND(valor3 < valor2) ENTÃO
    inicio
      SE (valor1 < valor2) ENTÃO
        escreva(valor3,valor1,valor2);
      SENAO
        escreva(valor3,valor2,valor1);
    fim
fim
```

Exercícios

Lista 02

- http://www.inf.puc-rio.br/~elima/intro-prog/

Estruturas de Repetição

- **Estruturas de repetição** são utilizadas para indicar que um determinado conjunto de instruções deve ser executado um número definido ou indefinido de vezes, ou enquanto uma condição não for satisfeita.

```
ENQUANTO <expressão lógica> FAÇA
Inicio
    <comando>;
Fim
```

Estruturas de Repetição

- “Escreva o pseudocódigo de um algoritmo que exiba todos os números inteiros entre 0 a 99”.

```
variáveis
    contador : inteiro;
início
    contador = 0;
    ENQUANTO (contador < 100) FAÇA
    início
      escreva(contador);
      contador = contador + 1;
    fim
fim
```

Exercícios

Lista 03

- http://www.inf.puc-rio.br/~elima/intro-prog/