Pontifícia Universidade Católica
do Rio de Janeiro

Introdução a Computação

Aula 02 – Introdução a Linguagem C

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

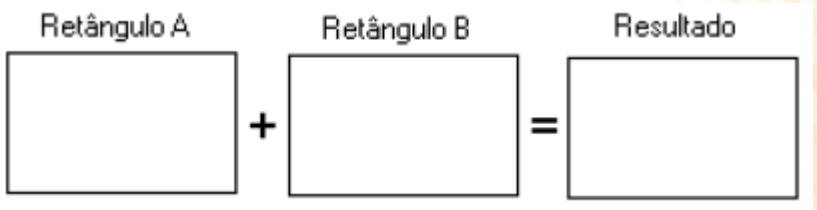
- Até mesmo tarefas simples, podem ser descritas por **sequências lógicas**:

"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);
}
```

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.

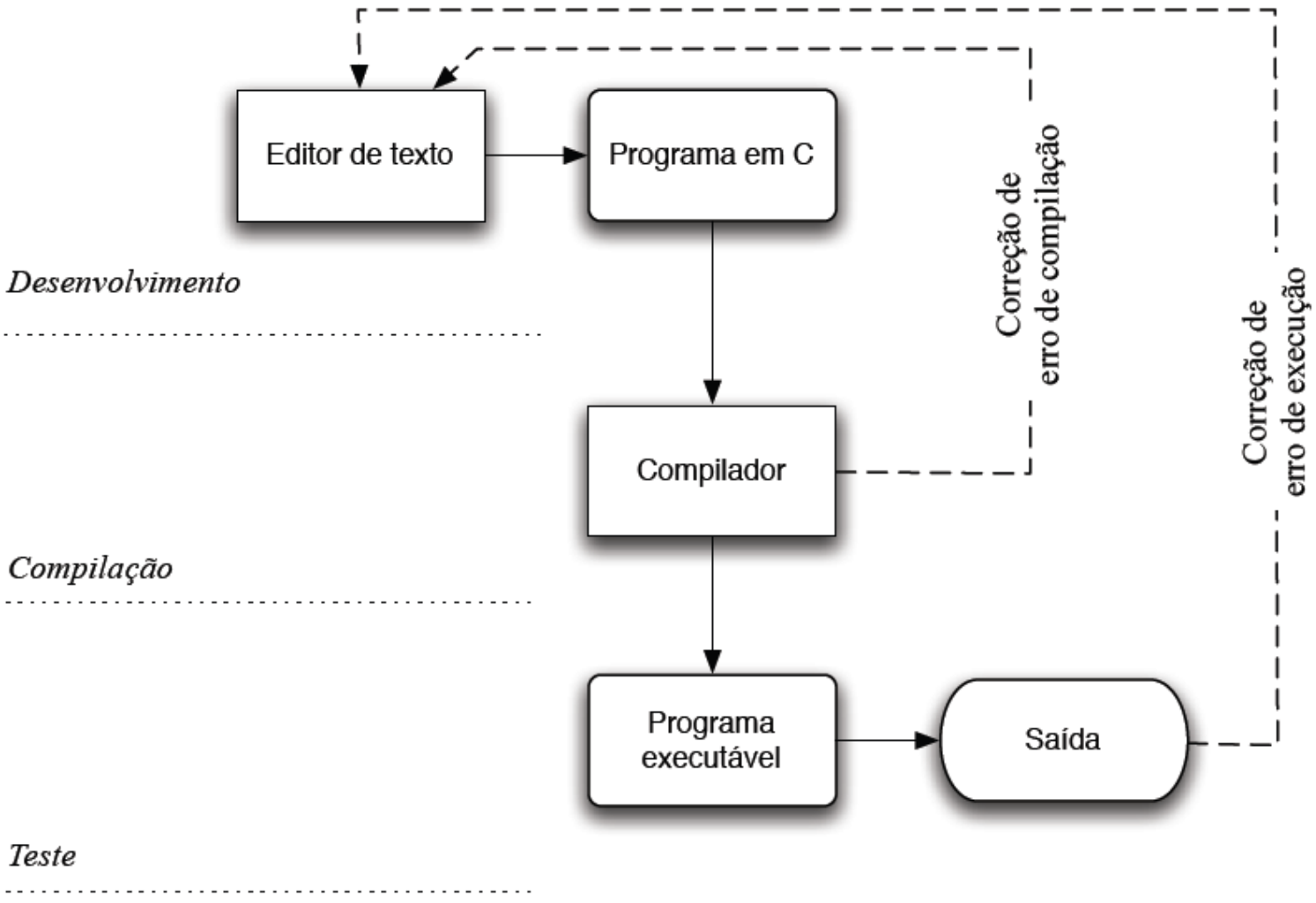
Programação

- Programa é um algoritmo escrito em uma **linguagem de programação**.

- Existem diversas linguagens de programação disponíveis:
  - C, C++, C#, Java, Pascal, Lua, Python...

- No nosso caso, utilizaremos a **linguagem C**.

Ciclo de Desenvolvimento de um Programa



Tutorial Visual Studio

Aula 03 – Introdução ao Visual Studio

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

Estrutura de um Programa C

- **Inclusão de bibliotecas auxiliares:**

```
#include <nome.h>
```

- **Função Principal:**

```
int main(void)
{
   ...
}
```

Bibliotecas Auxiliares

- **stdio.h**: funções de entrada de saída de dados:
  - printf, scanf...

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

- **math.h**: funções matemáticas:
  - cos, sen, sqrt, pow...

```
#include <math.h>
```

- **string.h**: funções de manipulação de texto (string):
  - strcmp, strlen...

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

Função Principal

```
int main(void)
{
      /* declarações de variáveis locais,
      chamadas a funções auxiliares,
      cálculos de expressões, leitura e
      escrita de dados, etc. */
}
```

Observação: /* delimita um comentário em C */

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.

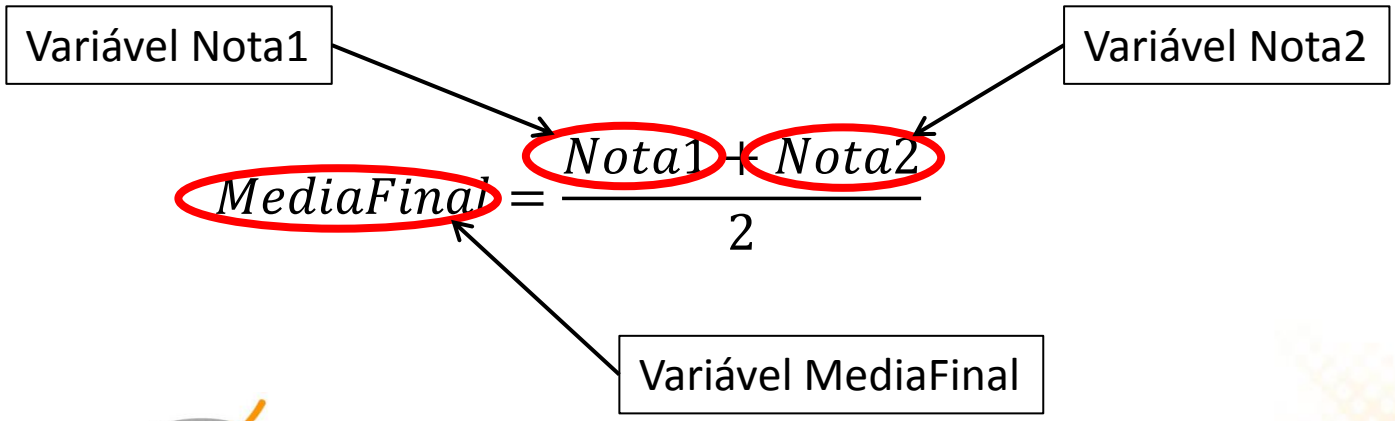
$$MediaFinal = \frac{Nota1 + Nota2}{2}$$

valor constante

Variáveis

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

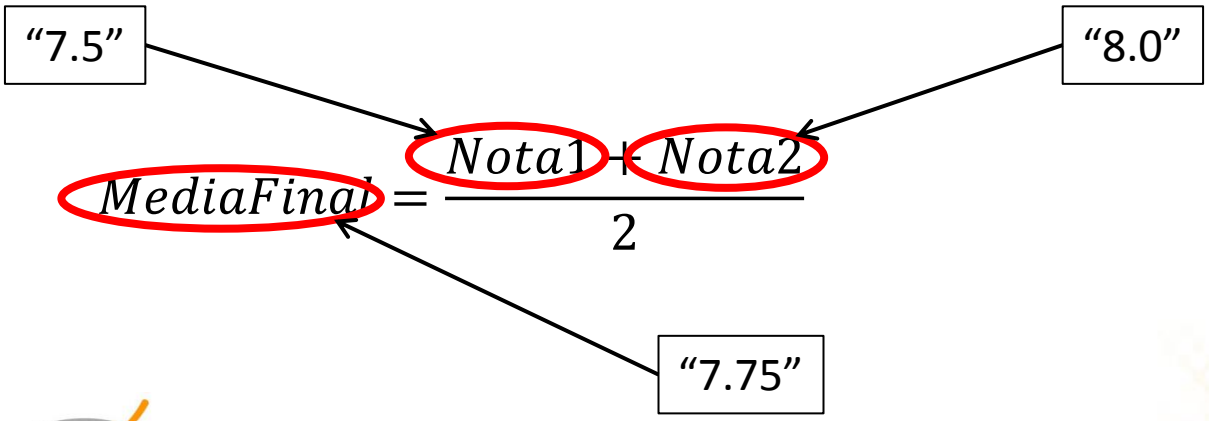
- Variáveis recebem **nomes** para serem referenciadas e modificadas quando necessário.



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)

Variáveis em C

- **Variável** é um espaço reservado na memória do computador para armazenar um tipo de dado.
- Devem receber **nomes** para poderem ser referenciadas e modificadas quando necessário.
- Toda variável tem:
  - um nome
  - um tipo de dado
  - um valor
- **Restrição para nomes:** não é permitido começar o nome com um algarismo (0-9), alguns caracteres não são válidos (*, -, /, +, ...), e palavras reservadas não podem ser utilizadas (main, if, while, ...).

Tipos de Variáveis

Pseudocódigo	C	Exemplo
inteiro	int	2
real	float	2.5
real	double	2.5
caractere	char	a

Observação: Existem outros tipos de dados, mas por enquanto vamos considerar apenas os tipos básicos.

Exemplo:

```
int main(void)
{
   int nota1;
   float valor;


}
```

Declaração de Variáveis

- Variáveis devem ser explicitamente declaradas.
- Variáveis podem ser declaradas em conjunto.

Exemplos:

```
int a;      /* declara uma variável do tipo int */
int b;      /* declara uma variável do tipo int */
float c;    /* declara uma variável do tipo float */
int d, e;   /* declara duas variáveis do tipo int */
int d = 5; /* declaração e inicialização da variável */
```

Operadores Aritméticos

- **Operadores aritméticos** são usados para se realizar operações aritméticas com as variáveis e constantes.

Operação	Símbolo
Adição	+
Subtração	-
Multiplicação	*
Divisão	/
Resto da Divisão	%

Exemplos:

operador de atribuição

```
total = preco * quantidade;
media = (nota2 + nota2)/2;
resultado = 3 * (1 - 2) + 4 * 2;
resto = numero % 3;
```

Funções de Entrada e Saída em C

- **Função "printf":** Permite a saída de dados, ou seja, a escrita de dados na tela.

- **Sintaxe:**

```
printf(formato, lista de constantes/variáveis/expressões...);
```

Exemplo	**Saída**
`printf("%d %f", 33, 5.3);`	33 5.3
`printf("Inteiro = %d Real = %f", 33, 5.3);`	Inteiro = 33 Real = 5.3
`printf("Curso de Programação\n");`	Curso de Programação

Funções de Entrada e Saída em C

- Especificação de formatos:

Formato	Descrição
`%c`	Especifica um char
`%d`	Especifica um int
`%f`	Especifica um float
`%s`	Especifica uma cadeia de caracteres

Observação: Existem outros formatos, mas por enquanto vamos considerar apenas os tipos básicos.

Funções de Entrada e Saída em C

- **Função "scanf":** Permite a entrada de dados, ou seja, a captura de valores fornecidos via teclado.

- **Sintaxe:**

```
scanf(formato, lista de endereços das variáveis...);
```

Exemplo
`scanf("%d", &valor1);`
`scanf("%f", &preco);`
`scanf("%c", &letra);`

Exemplo 01

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

```
#include <stdio.h>

int main(void)
{
  int valor, resposta;
  printf(“Digite um numero: “);
  scanf(“%d”, &valor);
  resposta = valor + 2;
  printf (“Resultado: %d”, resposta);
  return 0;
}
```

Exemplo 02

- Escreva um programa que leia dois números inteiros e retorne a soma deles.

```
#include <stdio.h>

int main(void)
{
  int numero1, numero2, resultado;
  printf("Digite o primeiro numero: ");
  scanf("%d", &numero1);
  printf("Digite o segundo numero: ");
  scanf("%d", &numero2);
  resultado = numero1 + numero2;
  printf ("Resultado da soma é %d", resultado);
  return 0;
}
```

Exercícios

Lista 01

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