

# FAETERJ - Paracambi

## Análise e Desenvolvimento de Sistemas

### Programação Estruturada – PRG - I

---

Prof. Carlos Eduardo Costa Vieira

# Resumo da Apresentação

- Algoritmos
  - Definição e Exemplos;
  - Características e Dificuldades;
  - Qualidades de um Bom Algoritmo;
- Representação de Algoritmos
  - Linguagem Natural;
  - Fluxograma;
  - Pseudo-linguagem;
- Linguagens de Programação;
- Linguagem C e *Integrated Development Environment* (IDE);
- Bibliografia.

# Algoritmos

- Fazem parte do cotidiano das pessoas
  - Ex: Instruções para o uso de medicamentos, indicações de como montar um aparelho, receita de culinária, etc
- Definição: Sequência de ações executáveis para a obtenção de uma solução para um determinado tipo de problema;
- Particularmente importante para problemas a serem solucionados em um computador.

# Algoritmos

- ▣ Idéia intuitiva: processo sistemático de resolver problemas;
- ▣ Um algoritmo computa uma saída (o resultado do problema) a partir de uma entrada (informações inicialmente conhecidas). O algoritmo manipula dados, gerados a partir de sua entrada.



# Exemplos

## ■ Exemplo I: Algoritmo para trocar lâmpadas

1. Se (lâmpada queimada estiver fora do alcance)  
    Pegar a escada;
2. Pegar a lâmpada queimada;
3. Se (lâmpada queimada estiver quente)  
    Pegar pano;
4. Tirar lâmpada queimada;
5. Colocar lâmpada boa.

# Exemplos

## ■ Exemplo2: Algoritmo para fazer uma prova

1. Ler a prova;
2. Pegar a caneta;
3. Enquanto ( (houver questão em branco) e (tempo não terminou) )  
faça  
    Se (souber a questão)  
        Resolvê-la;  
    Senão  
        Pular para outra;
4. Entregar a prova.

# Características

- Descrição de um procedimento rotineiro;
- Finitude: devem terminar após um número finito de passos (tem um início e um fim);
- Definição: cada passo deve ser precisamente definido;
- Entradas: devem ter zero ou mais entradas;
- Saídas: devem ter uma ou mais saídas;
- Efetividade: todas as operações devem ser simples de modo que possam ser executadas em um tempo limitado por um ser humano.

# Dificuldades

- Pode haver mais de uma solução para um problema;
- A criação de algoritmos não é um processo automático e tem muito de arte;
- Grande número de informações que os alunos precisam absorver em pouco tempo;
- Difícil para iniciantes saberem o que o computador pode ou não fazer.



# Qualidades de um Bom Algoritmo

- Definição perfeita: Deve descrever exatamente quais são as instruções que devem ser executadas e em que sequência;
- Ausência de Ambiguidade: Não deve deixar dúvidas sobre o que deve ser feito;
- Eficácia: Conseguir resolver o problema em qualquer situação. Todas as situações de exceção do algoritmo devem ser especificadas e tratadas;
- Eficiência: Sempre se deve buscar, dentre os diversos algoritmos que resolvam um mesmo problema, aquele que utiliza a menor quantidade de recursos (espaço de memória e/ou tempo de processamento).

# Representação de Algoritmos

- Linguagem Natural: Algoritmos expressos diretamente em linguagem natural, como nas receitas;
- Fluxogramas: Representação gráfica;
- Pseudo-linguagem: emprega linguagem intermediária entre linguagem natural e linguagem de programação.

# Linguagem Natural

## ▣ Exemplo:

1. Repetir 10 vezes os quatro  
exercícios abaixo:

Levantar e abaixar o braço  
direito;

Levantar e abaixar o braço  
esquerdo;

Levantar e abaixar a perna  
direita;







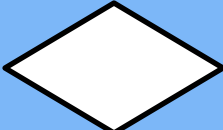
Levantar e abaixar a perna  
esquerda;

# Fluxogramas

- Representação de algoritmos por meio de símbolos geométricos;
- Cada tipo de operação é representado por um símbolo diferente;
- Vantagem: Permitir o acompanhamento visual do fluxo do algoritmo.

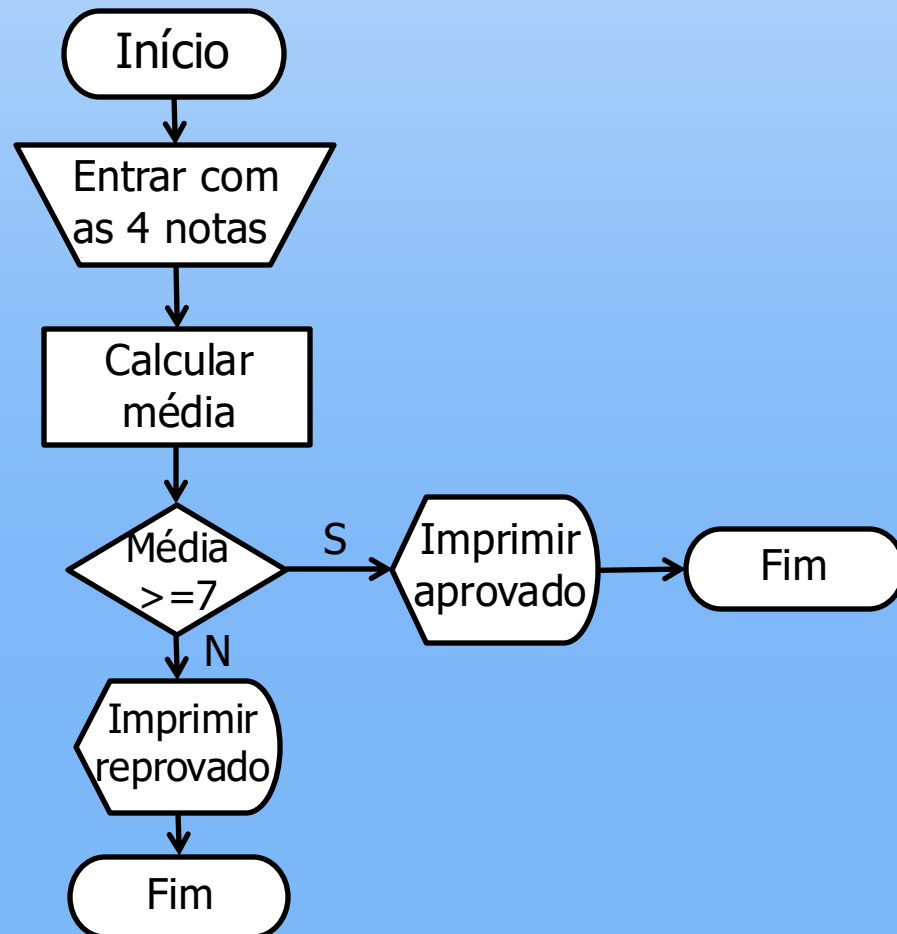
# Fluxogramas

## ■ Alguns símbolos

	Terminal: símbolo utilizado para indicar o início e/ou fim do fluxo do programa.
	Processamento: símbolo utilizado para indicar cálculos e atribuições de valores.
	Seta: permite indicar o sentido do fluxo de dados. Serve para conectar os símbolos existentes.
	Entrada de Dados: utilizado para ler os dados necessários ao programa.
	Saída de dados em vídeo: utiliza-se este símbolo quando se quer mostrar dados na tela do vídeo.
	Saída de dados em impressora: utiliza-se este símbolo quando se quer que os dados sejam impressos.
	Decisão: indica a decisão que deve ser tomada com possibilidade de desvios para outros pontos de fluxo.

# Fluxogramas

- Cálculo da média com quatro notas bimestrais que determinam a aprovação ou reprovação em uma disciplina



# Pseudo-linguagem

- Mais conhecido como Português Estruturado ou Portugol;
- Método que procura misturar as facilidades da linguagem natural com a precisão das linguagens de programação;
- Não existe um padrão para esta forma de descrição;
- Exemplo: Pseudo-linguagem utilizada no software VisuAlg.

# Pseudo-linguagem

**Algoritmo** "Cálculo de Média Aritmética"

**Var**

NUM1, NUM2, Media: REAL

**Inicio**

//Entrada - Leitura dos valores

**Escreval**("Calcula a média aritmética de 2 valores.")

**Escreva**("Digite um valor : ")

**Leia**(Num1)

**Escreva**("Digite outro valor : ")

**Leia**(Num2)

//Processamento - Cálculo da Média

Media <- (Num1+Num2)/2

//Saída - Impressão do resultado

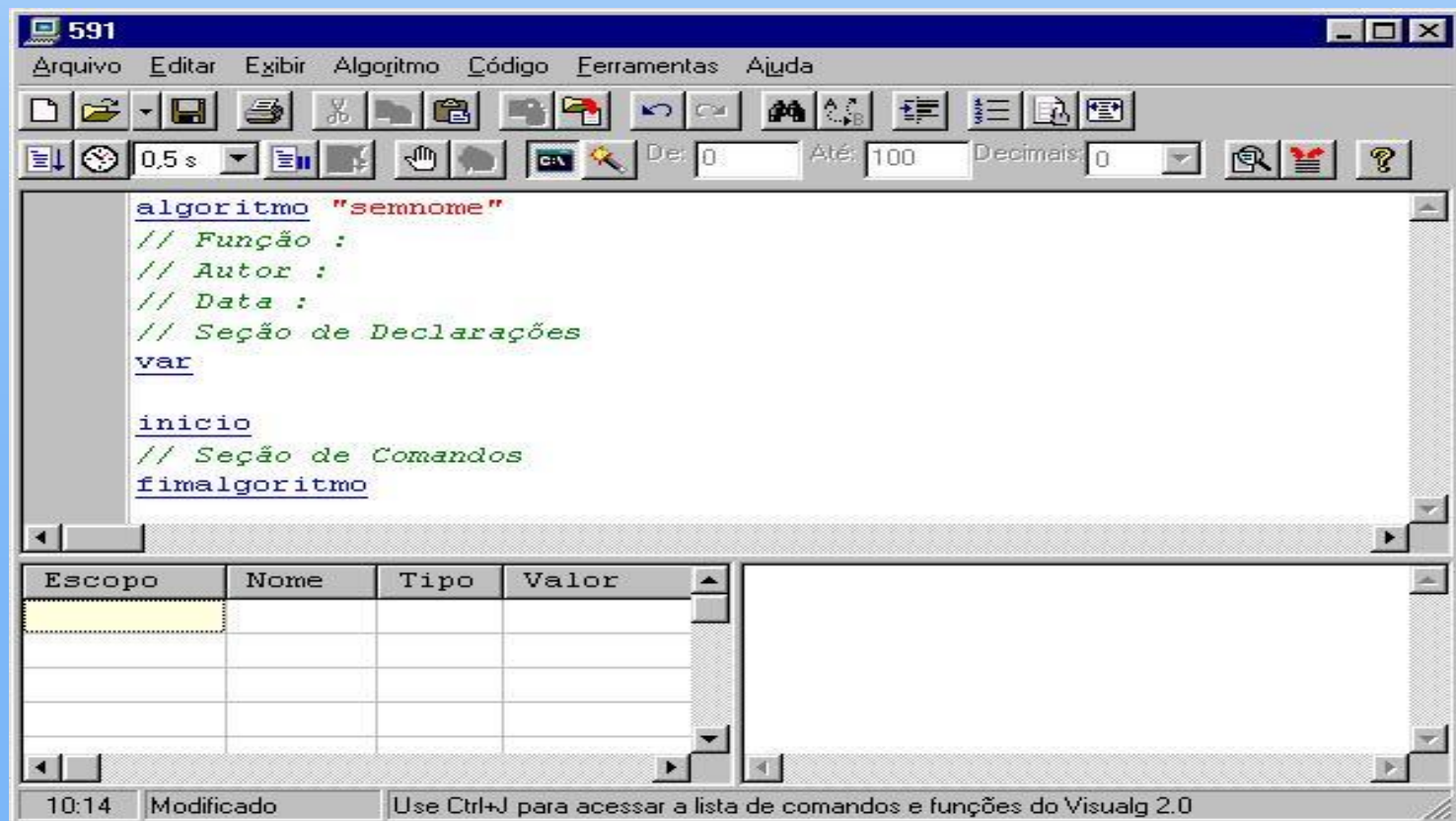
**Escreva**("A média dos dois valores é : ", Media)

**FimAlgoritmo**



# Pseudo-linguagem

- O VisuAlg (Visualizador de Algoritmo) é um programa que edita, interpreta e executa algoritmos com uma linguagem próxima do português estruturado (pseudo-linguagem);



# Linguagens de Programação

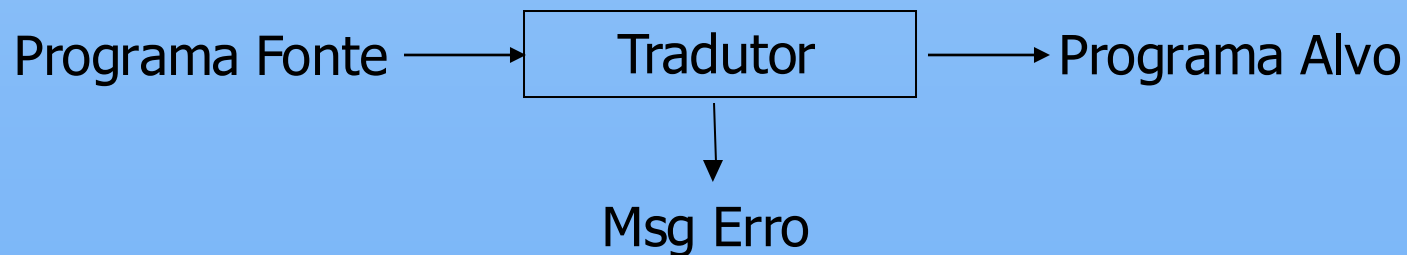
- ▣ Programar é basicamente estruturar dados e construir algoritmos;
- ▣ Programas são formulações concretas de algoritmos abstratos, basedos em representações e estruturas específicas de dados;
- ▣ Uma linguagem de programação é uma técnica de notação para programar, com a intenção de servir de veículo tanto para a expressão do raciocínio algorítmico quanto para a execução automática de um algoritmo por um computador.

# Linguagens de Programação

- Linguagem: meio eficaz de comunicação entre pessoas.
- Linguagem de Programação: comunicação entre o indivíduo e o computador.
  - Linguagem de baixo nível: linguagem de máquina e linguagem simbólica;
  - Linguagem de alto nível: mais próxima às linguagens naturais;
- Programas escritos em linguagem de alto nível devem ser traduzidos para linguagem de máquina.

# Linguagens de Programação

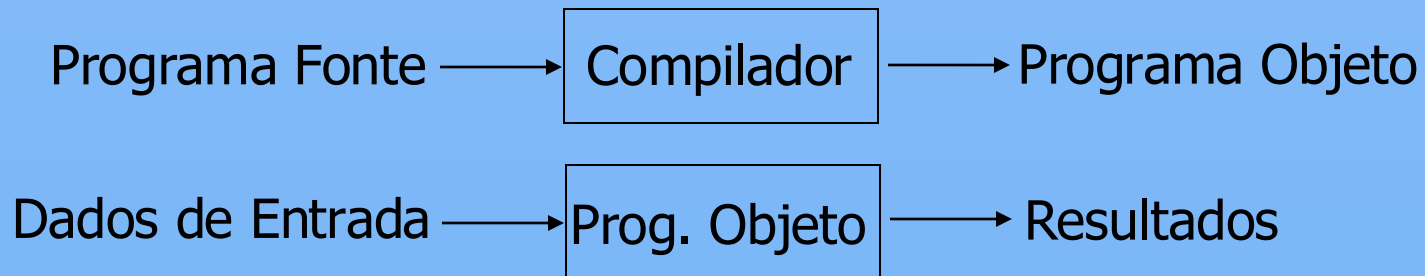
- ▣ Tradutor: Programa que lê um programa escrito numa linguagem (fonte) e o traduz num programa equivalente numa outra linguagem (alvo).



# Linguagens de Programação

## ■ Tipos de Tradutores

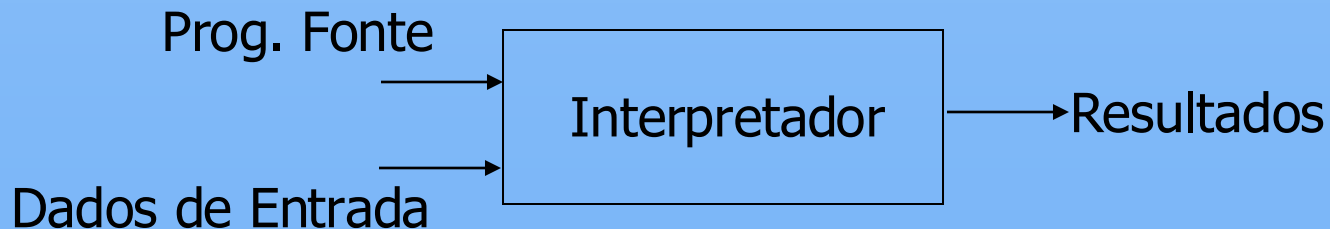
- Montadores (*assemblers*): mapeiam instruções em linguagem simbólica (Assembly) para instruções em linguagem de máquina;
- Compiladores: mapeiam programas escritos em linguagem de alto nível para programas equivalentes em linguagem simbólica ou linguagem de máquina;



# Linguagens de Programação

## ■ Tipos de Tradutores

- Interpretadores: Em vez de produzir um programa objeto, um interpretador executa diretamente as operações especificadas no programa fonte sobre as entradas fornecidas pelo usuário.

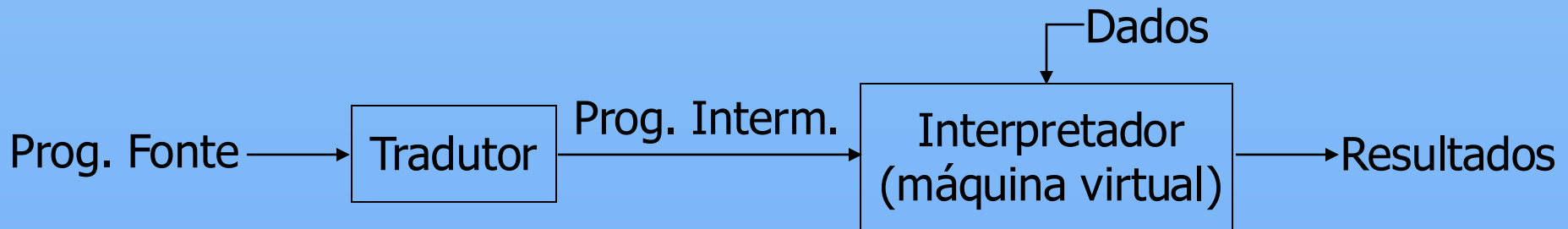


# Linguagens de Programação

## ▣ Tipos de Tradutores

- ▣ **Compilador Híbrido:** Combinam compilação e interpretação. Ex: Linguagem Java

- ▣ Um programa fonte em Java é primeiro compilado para uma forma intermediária (*bytecodes*), que são então interpretados por uma máquina virtual;



# A Linguagem C

- No início era uma linguagem voltada para desenvolvimento de programas que eram utilizados na plataforma UNIX;
- Sucessora da linguagem B criada por Ken Thompson a partir da linguagem BCPL;
- Linguagem expressiva e abrangente;
- Linguagem Compilada;
- Linguagens modernas baseadas em C: C++, C#, Java.



# A Linguagem C

Ken Thompson e Dennis Ritchie são famosos no campo dos sistemas operacionais por terem desenvolvido o sistema operacional UNIX e a linguagem de programação C. Conquistaram reconhecimento e receberam diversos prêmios por suas realizações, entre eles o ACM Turing Award, a National Medal of Technology, o NEC C&C Prize, o IEEE Emmanuel Piore Award, a IEEE Hamming Medal, e foram escolhidos como membros da United States National Academy of Engineering e da Bell Labs National Fellowship.<sup>1</sup>

Ken Thompson frequentou a Universidade da Califórnia em Berkeley, onde graduou-se e fez mestrado em Ciência da Computação, em 1966.<sup>2</sup> Após a universidade, Thompson trabalhou no Bell Labs, onde, por fim, juntou-se a

Dennis Ritchie no projeto Multics.<sup>3</sup> Enquanto trabalhava naquele projeto, Thompson criou a linguagem B, que deu origem à linguagem C de Ritchie.<sup>4</sup> O projeto Multics posteriormente levou à criação do sistema operacional UNIX em 1969. Thompson continuou a desenvolver o UNIX no início da década de 70, reescrevendo-o na linguagem de programação C de Ritchie.<sup>5</sup> Depois de ter concluído o UNIX, Thompson novamente foi notícia em 1980 com o Belle. O Belle era um computador que jogava xadrez, projetado por Thompson e Joe Condon, que ganhou o Campeonato Mundial de Xadrez por Computador. Thompson trabalhou como professor na Universidade da Califórnia, em Berkeley, e na Universidade de Sydney, na Austrália. Continuou a trabalhar no Bell Labs até se aposentar em 2000.<sup>6</sup>

Dennis Ritchie frequentou a Universidade Harvard, onde se graduou em Física e fez doutorado em Matemática. Ritchie foi trabalhar no Bell Labs onde se juntou a Thompson no projeto Multics em 1968. Ritchie é mais reconhecido por sua linguagem C, que concluiu em 1972.<sup>7</sup> Ele agregou algumas capacidades à linguagem B de Thompson e mudou a sintaxe para torná-la mais fácil de utilizar. Ritchie ainda trabalha no Bell Labs e continua trabalhando com sistemas operacionais.<sup>8</sup> Nos últimos dez anos, criou dois novos sistemas operacionais, Plan 9 e Inferno.<sup>9</sup> O sistema Plan 9 é projetado para computação distribuída de alta qualidade.<sup>10</sup> O Inferno é um sistema destinado a trabalho avançado em rede.<sup>11</sup>

# A Linguagem C

## ▣ Índice TIOBE (<https://www.tiobe.com/tiobe-index/>)

Aug 2024	Aug 2023	Change	Programming Language		Ratings	Change
1	1			Python	18.04%	+4.71%
2	3	▲		C++	10.04%	-0.59%
3	2	▼		C	9.17%	-2.24%
4	4			Java	9.16%	-1.16%
5	5			C#	6.39%	-0.65%
6	6			JavaScript	3.91%	+0.62%
7	8	▲		SQL	2.21%	+0.68%

# *Integrated Development Environment (IDE)*

## ■ Finalidade

- Fornecer um editor de texto sofisticado;
- Oferecer um ambiente para a depuração de um programa;
- Gerenciar projetos;
- Escrever o código-fonte (.c);

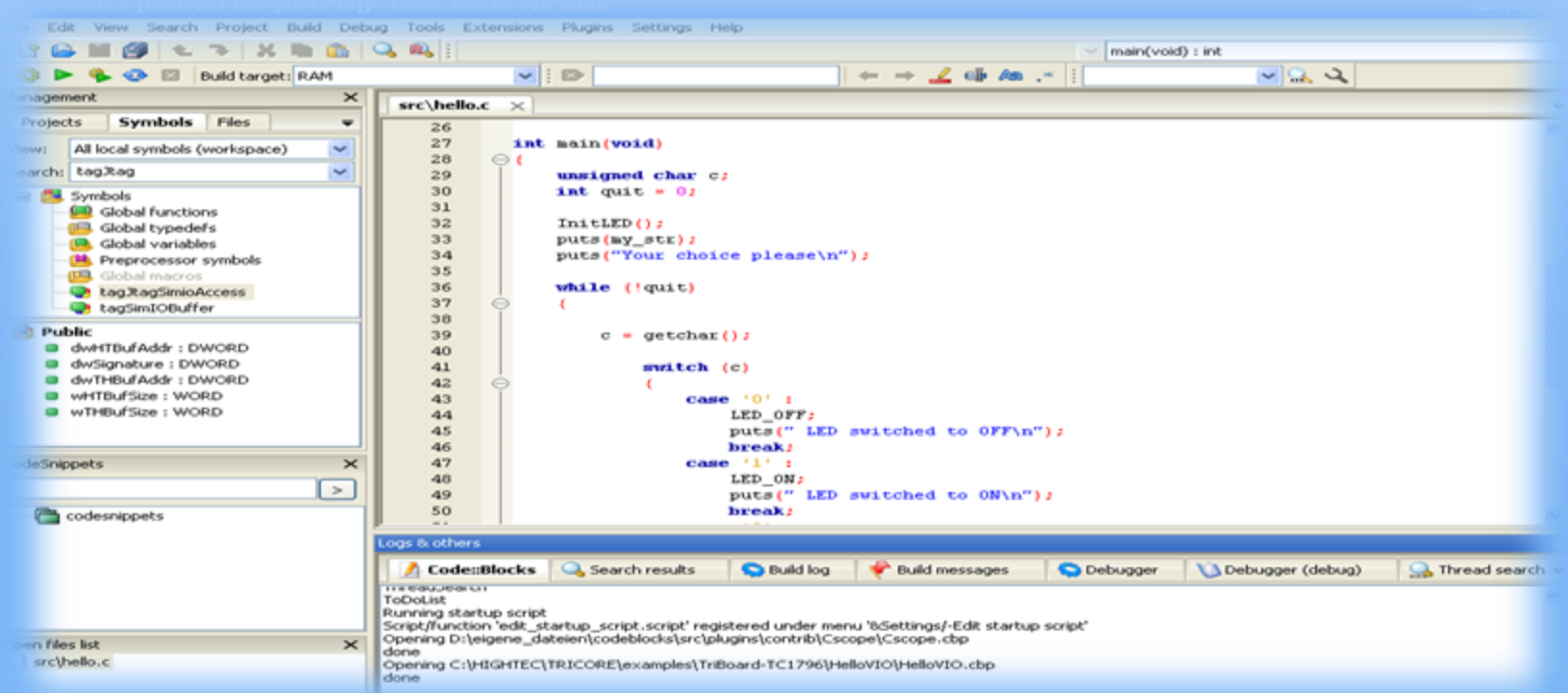
## ■ Exemplos:

- *Code::Blocks, CLion, Visual Studio Code, etc.*



# Integrated Development Environment (IDE)

- Usaremos o *Code::Blocks* para digitar o código e executar o programa (*RAD* – *Rapid Application Development*)



# Bibliografia

- GOODRICH, Michael T; TAMASSIA, Roberto. **Estrutura de Dados e Algoritmos em Java.** 4. ed. Porto Alegre: Bookman, 2007.
- LOPES, Anita; GARCIA, Guto. **Introdução à Programação: 500 Algoritmos Resolvidos.** Rio de Janeiro: Elsevier, 2002.
- PUGA, Sandra; RISSETTI, Gerson. **Lógica de Programação e Estrutura de Dados:** com Aplicações em Java. São Paulo: Prentice-Hall do Brasil , 2003.