Marisangila Alves, MSc
marisangila.alves@catolicasc.org.br
marisangila.com.br

Católica de Santa Catarina

2025/1

Fundamentos de Programação

Pseudocódigo

Sumário

Programa

Computador:

É um dispositivo capaz de realizar cálculos e tomar decisões lógicas com uma velocidade milhões ou mesmo bilhões de vezes mais rápida do que os seres humanos (Deitel; Deitel, 2011).

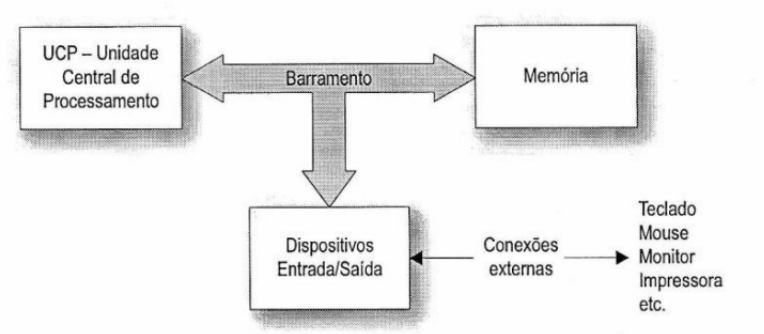


Figura 1: Arquitetura simplificada de um computador (Medina; Fertig, 2005).

Recapitulando...

> *Os computadores processam dados sob o controle de conjuntos de instruções chamados programas de computador. [...] Os vários dispositivos (como teclado, tela, discos, memória e unidades de processamento) que constituem um sistema computacional são chamados de hardware. Os programas executados em um computador são chamados de software.*
>
> (Deitel; Deitel, 2011)

Função básica de um computador (De Oliveira; Manzano, 2004).

- Entrada: Coleta de dados do mundo real, geralmente por meio de entrada manual.
- Processamento: Realização de operações e transformações nos dados.
- Saída: Apresentação visual ou armazenamento dos resultados no mundo real.

> *Todo programa de computador deve, de alguma forma, possibilitar a entrada dos dados do mundo exterior, produz ir a ação de processamento, tanto lógico quanto matemático, sempre que essas ações forem necessárias, e acima de tudo possibilitar a saída de dados que tenham sido processados ou estej am apenas armazenados.*
>
> (De Oliveira; Manzano, 2004)

O que é um Programa?

> *Um conjunto de instruções que será executado pelo processador em uma determinada sequência. Esse programa leva o computador a executar alguma tarefa.*
>
> (Medina; Fertig, 2005)

Resumindo:

Para que um algoritmo escrito em pseudocódigo ou representado como um fluxograma seja executado, é necessário que ele seja traduzido para uma linguagem de programação. Uma sequência de instruções escritas em uma linguagem de programação e armazenada em um arquivo constitui um programa, que, quando executado, se torna um *software*.

Linguagem de Programação

O que é uma Linguagem de Programação?

- A linguagem de programação que um computador é capaz de compreender é composta apenas de números.
- Sendo assim, pode ser extremamente complicado, para nós, seres humanos, criarmos algoritmos na linguagem de máquina.
- Por esse motivo, existem linguagens de programação que podem ser compreendidas por seres humanos.
- No entanto, para que o programa possa ser executado, a linguagem de programação deve ser traduzida para linguagem de máquina.
- As linguagens de programação têm uma sintaxe específica, e são por definição, no idoma Inglês[1].

- **Linguagem alto nível:**
  - São mais próximas da linguagem humana ou escrita convencional.
  - Exemplos: Python, Java, C, C++, C#, PHP, R, Go .
  - São utilizadas na maioria dos *softwares*: aplicativos, websites, sistemas operacionais e dispositivos eletrônicos.
- **Linguagem baixo nível:**
  - Mais próximas da linguagem de máquina, com instruções específicas para o *hardware*.
  - Exemplo: Assembly.
  - Utilizado em programas que interagem diretamente com o *hardware* do computador ou outros dispositivos.
- **Linguagem de máquina:**
  - É a linguagem nativa dos computadores, composta exclusivamente por números binários (0s e 1s).
  - Executada diretamente pela Unidade Central de Processamento (CPU) sem necessidade de tradução.
  - Extremamente difícil de ler e escrever diretamente por humanos.

Linguagem de máquina	Linguagem de baixo nível	Linguagem de alto nível
0010 0001 1110	LOAD R1, a	`a = a + b`
0010 0010 1111	LOAD R2, b	
0001 0001 0010	ADD R1, R2	
0011 0001 1111	STORE R1, a	

Tabela 1: Comparação entre linguagens de máquina, baixo nível e alto nível.

Linguagem de Programação IV

```python
a = 5
b = 3
a = a + b
```

Python.

```assembly
section .data
    a dq 5
    b dq 3
section .text
    global _start
_start:
    ; LOAD R1, a
    mov rax, [a]
    ; LOAD R2, b
    mov rbx, [b]
    ; ADD R1, R2
    add rax, rbx
    ; STORE R1, a
    mov [a], rax
    mov rax, 60
    xor rdi, rdi
    syscall
```

Assembly.

[1] Linguagens de programação são sempre escritas no idioma inglês, pelo fato de a ideia de constituição de hardware e de software ter sido proposta por cientistas ingleses (Charles Babbage e Ada Augusta)

Compilador

> *O compilador, a partir do código em linguagem de alto ńivel, chamado código-fonte, gera um arquivo com o código em linguagem de máquina, conhecido como código-objeto. Esse código-objeto fica em disco e só carregado em memória no momento da execução.*
>
> (Medina; Fertig, 2005)

Figura 2: Compilação de um programa (Medina; Fertig, 2005).

Interpretador

> *O interpretador faz o mesmo trabalho, porém não gera o arquivo em código-objeto. As instruções são traduzidas para linguagem de máquina em tempo de execução, instrução a instrução.*
>
> (Medina; Fertig, 2005)

Figura 3: Interpretação de um programa (Medina; Fertig, 2005).

Ferramentas de Programação

- **Editor de Texto:**
  - É uma ferramenta simples usada para implementar[2] e editar código-fonte;
  - Genérico, pode ser configurado para qualisquer linguages de programação.
- **IDE (Ambiente de Desenvolvimento Integrado):**
  - É uma ferramenta completa que combina editor de texto, compilador ou interpretador, depurador e outras funcionalidades para facilitar o desenvolvimento de *software*.
  - Normalmente, é específico para apenas uma linguagem de programação.

[2] **Implementar** é o processo de escrever e desenvolver o código necessário para um programa.

(a) Atom (GitHub, 2024)

(b) VIM (Vim, 2024)

(c) Visual Studio Code (Microsoft, 2024)

(d) Sublime (Sublime, 2024)

Figura 4: Editores de texto.

(a) Android Studio (Google, 2025)

(b) Eclipse (Foundation, 2025)

(c) IntelliJ IDEA (JetBrains, 2025a)

(d) NetBeans (Medina; Fertig, 2005)

(e) PyCharm (JetBrains, 2025b)

(f) VS Code (Microsoft, 2025)

(g) Xcode (Apple, 2025)

(h) R Studio (RStudio, 2025)

Figura 5: Principais IDEs para desenvolvimento de *software*.

> *Editor e Interpretador de algoritmos escritos em português estruturado (portugol) com base na linguagem Pascal.*
>
> (Nicolodi, 2024)

Atenção!

- Pseudocódigo é uma linguagem de projeto de programação, geralmente estática, que não permite a criação de programas executáveis.

Atenção!

- Pseudocódigo é comumente utilizado em descrições, documentações e artigos científicos, sendo bastante abstrata e sem rigor sintático, o que permite seu uso no idioma de preferência.
- No entanto, para fins de aprendizagem, o Portugol Studio é um ambiente de desenvolvimento construído para permitir a criação e a execução dos programas escritos em Português Estruturado (ou Portugol).
- Nesse contexto, é possível criar programas que podem ser executados como *softwares*, mas apenas para fins didáticos.
- O Portugol Studio não é utilizado para o desenvolvimento de *softwares* profissionais ou comerciais.
- Acesse: Portugol Studio

Tipos de Dados

Dado e Informação

Um computador e seu programa ou programas de controle usam dois elementos, sendo dados e instruções.

> *Os dados são elementos do mundo exterior, que representam dentro de um computador digital as informações manipuladas pelos seres humanos.*
>
> (De Oliveira; Manzano, 2004)

Dado:

O dado é a matéria-prima da informação. Para que uma informação exista , é necessário que um conjunto de dados relevantes a um usuário sejam processados.

Os dados primitivos são classificados em tipos, sendo eles:

- Inteiro;
- Real;
- Cadeia(Caractere);
- Lógico.

Inteiro

- Os dados numéricos positivos e negativos pertencem ao conjunto de números inteiros, excluindo dessa categoria qualquer valor numérico fracionário (que pertence ao conjunto de números reais).

Exemplo:

```
-15,-2,5,150,1590
```

Real

- São reais os dados numéricos positivos e negativos que pertencem ao conjunto de números reais, incluindo nessa categoria todos os valores fracionários e inteiros.

Exemplo:

```
-3.5,5.7,8.0,10.2,75.2
```

Cadeia

- São caracteres delimitados pelos símbolos aspas ‘"‘ ‘"‘. Eles são representados por letras (de A até Z), números (de 0 até 9), e símbolos.
- Normalmente, em cadeia.

Exemplo:

```
"PROGRAMAÇÃO", "Rua Alfa, 52 - Apto.  1", "Fone:  (0xx99) 5544-3322",
"CEP: 11222-333", "(espaço em branco)", "7", -90", "45.989", "@"
```

Lógico

- São lógicos os dados com va lores binários do tipo sim e não, verdadeiro e falso, 1 e O, entre outros, em que apenas um dos valores pode ser escolhido.

Exemplo:

```
0, 1, VERDADEIRO, FALSO
```

Variáveis

Variável é tudo que está sujeito a variações, que é incerto, instável ou inconstante.

Analogia:

Uma variável é como um escaninho: cada escaninho pode armazenar apenas um item de cada vez e precisa ser identificado por um nome único para que possamos acessar ou trocar o item armazenado.

Definição

Do ponto de vista computacional pode-se definir, de forma simplista, que uma variável é a representação de uma região de memória utilizada para armazenar, acessar e mod ificar certo valor por um determinado espaço de tempo. O tempo de armazenamento de um valor em uma variável está relacionado ao tempo de duração da execução de um programa (De Oliveira; Manzano, 2004).

- Uma variável deve ter um tipo definido.
- Uma variável deve ter um nome definido.
- Uma variável armazena apenas um único valor.
- Uma variável pode ter seu valor alterado.

Declarar variáveis:

```
programa{
    funcao inicio(){
        inteiro idade
        real altura
        cadeia nome
        logico faz_sol_hoje
    }
}
```

Código 3: Exemplo de declaração de variáveis.

Variáveis do mesmo tipo podem ser declaradas na mesma linha separadas por vírgula.

```
1 programa{
2     funcao inicio(){
3     inteiro idade, quantidade
4     real altura, peso, preco
5     cadeia nome, sobrenome
6     }
7 }
```

Código 4: Exemplo de declaração de variáveis do mesmo tipo.

Como nomear variáveis?

- Os nomes das variáveis devem começar por uma letra e depois conter letras, números ou underline, até um limite de 30 caracteres.
- As variáveis podem ser simples ou estruturadas (na versão atual, os vetores podem ser de uma ou duas dimensões).
- Não pode haver duas variáveis com o mesmo nome.
- Não devem ser declaradas usando palavras reservadas.

Atribuição de valor:

```
programa{
    funcao inicio(){
        inteiro idade
        real altura
        cadeia nome
        logico faz_sol_hoje

        idade = 16
        altura = 1.65
        nome: 'Fulano'
        faz_sol_hoje = verdadeiro
    }
}
```

Código 5: Exemplo de atribuição de valor.

<- Operador de atribuição.

`x <- 10` : Significa que a variável chamada `x` recebe o valor 10.

Instruções

Entrada:

```
programa{
    funcao inicio(){
        inteiro idade
        real altura
        cadeia nome
        logico faz_sol_hoje
        leia(idade)
        leia(altura)
        leia(nome)
        leia(faz_sol_hoje)
    }
}
```

Código 6: Exemplo de entrada de dados.

Saída:

```
programa{
    funcao inicio(){
        inteiro idade
        real altura
        cadeia nome
        logico faz_sol_hoje
        escreva("Qual sua idade?\n")
        leia(idade)
        escreva("Qual sua altura?\n")
        leia(altura)
        escreva("Qual seu nome?\n")
        leia(nome)
        escreva("Faz sol hoje?\n")
        leia(faz_sol_hoje)
    }
}
```

Código 7: Exemplo de saída de dados.

Operador	Descrição
+	Adição
-	Subtração
*	Multiplicação
/	Divisão
%	Resto da divisão (Módulo)

Tabela 2: Operadores aritméticos.

Expressões Aritméticas:

```
programa{
funcao inicio(){
        inteiro a
        inteiro b
        inteiro resultado
        escreva("Digite um número:\n")
        leia(a)
        escreva("Digite outro número:\n")
        leia(b)
        resultado = a + b
        escreva(resultado)
    }
}
```

Código 8: Exemplo de uso de operadores.

Tradicional:

$$\left\{\left[\frac{2}{3} - (5-3)\right] + 1\right\} \cdot 5$$

Computacional:

$$((2/3 - (5-3)) + 1) * 5$$

Concatenar saída:

```
programa{
funcao inicio(){
        inteiro a, b, resultado
        escreva("Digite um número:\n")
        leia(a)
        escreva("Digite outro número:\n")
        leia(b)
        resultado = a + b
        escreva("O resultado é: ", resultado)
    }
}
```

Código 9: Exemplo de concatenação.

Comentário:

```
programa{
    funcao(){
    //isso é um comentário!
    escreva("Hello World!\n")
    //comentário são ignorados!
    //comentários poder ser um lembrete ou uma nota para te
     ↪  ajudar!
    }
}
```

Código 10: Exemplo de comentários.

Constante

Do ponto de vista computacional, que é semelhante ao matemático ou científico, constante é uma grandeza numérica fixa utilizada normalmente numa expressão aritmética ou matemática, que define um valor que será inalterado na expressão, independentemente das variáveis envolvidas na operação a ser realizada (De Oliveira; Manzano, 2004).

Atenção

Padrão de nomenclatura: TODAS AS LETRAS DEVEM SER MAIÚSCULAS!

Definição: II

Constante:

```
programa{
    //DECLARAR CONSTANTE AQUI!
    const real PI = 3.14159
    funcao inicio()
    {
        real area, raio
        escreva("Qual o valor do raio?\n")
        leia(raio)
        area = PI * (raio * raio)
        escreva("A área do círculo é:", area)
    }
}
```

Código 11: Exemplo de constante.

Boas Práticas de Programação

Atenção

Cuidado com a identação, código identado incorretamente ou desorganizado pode dificultar a analise do código pra encontrar erros ou até gerar erros.

Atenção!

O nome de uma variável deve ter um significado lógico, de forma que alguém apenas lendo o nome seja possível imaginar o que deve ser armazenado na variável.

```
programa{
    funcao inicio(){
        cadeia n
        real a
        inteiro x
        cadeia dt
    }
}
```

ERRADO!

```
programa{
    funcao inicio(){
        cadeia nome
        real altura
        inteiro idade
        cadeia data_nascimento
    }
}
```

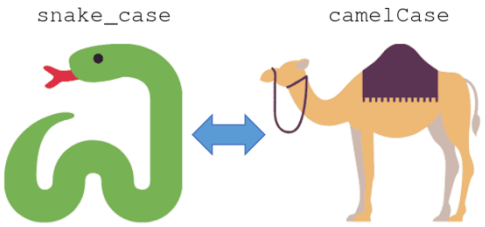
CORRETO!

Boas Práticas de Programação I

Padrões de nomenclatura

```
1  programa{
2      funcao inicio(){
3          cadeia nome_aluno
4          inteiro idade_aluno
5          real media_geral_aluno
6          cadeia
           ↪  data_nascimento_aluno
7          cadeia cpf_aluno
8          cadeia email_aluno
9      }
10 }
```

Código 12: Padrão de nomenclatura snake_case.



Information

Atenção! Sempre nomear variáveis.

```
programa{
    funcao inicio(){
        real preco_produto, preco_final
        escreva("Insira o preço do
        ↪ produto:\n")
        leia(preco_produto)
        preco_final = preco_produto /
        ↪ 1.05
    }
}
```

ERRADO!

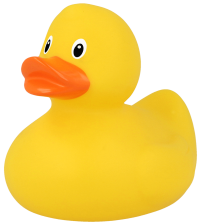
```
programa{
    const real DESCONTO = 1.05
    funcao inicio(){
        real preco_produto, preco_final
        escreva("Insira o preço do
        ↪ produto:\n")
        leia(preco_produto)
        preco_final = preco_produto /
        ↪ DESCONTO
    }
}
```

CORRETO!

Figura 7: Exemplo de nomeação de variáveis.

- Comentários são úteis em ambiente de ensino. No entanto, evite comentários em grandes projetos. Se seu código precisa de um comentário para ser entendido por outra pessoa ou até por você mesmo, é provável que ele seja uma gambiarra. REFAÇA!
- Espaçamentos e linhas em branco são úteis em ambiente de ensino. No entanto, evite excessos de linhas em branco em grandes projetos.

- Diante de problemas complexos, pare, pense e desenhe!
- Quando enfrentar um problema sem solução aparente, tente explicá-lo em voz alta.




"Como você ficou tão bom em programação?"
Programando

APPLE. **Xcode**. Acesso em: 8 jan. 2025. 2025. Disponível em: https://developer.apple.com/xcode.

DE OLIVEIRA, J.F.; MANZANO, J.A.N.G. **Algoritmos: Lógica para Desenvolvimento de Programação de Computadores**. 16. ed. São Paulo: Editora Érica, 2004.

DEITEL, Paul; DEITEL, Harvey. **C: Como Programar**. 6. ed. São Paulo: Pearson Universidades, 2011.

FOUNDATION, Eclipse. **Eclipse IDE**. Acesso em: 8 jan. 2025. 2025. Disponível em: https://www.eclipse.org.

GITHUB. **Atom - A hackable text editor for the 21st Century**. Acesso em: 5 jan. 2025. 2024. Disponível em: https://github.com/atom/atom.

GOOGLE. **Android Studio**. Acesso em: 8 jan. 2025. 2025. Disponível em: https://developer.android.com/studio.

JETBRAINS. **IntelliJ IDEA**. Acesso em: 8 jan. 2025. 2025. Disponível em: https://www.jetbrains.com/idea.

JETBRAINS. **PyCharm**. Acesso em: 8 jan. 2025. 2025. Disponível em: https://www.jetbrains.com/pycharm.

MEDINA, M.; FERTIG, C. **Algoritmos e Programação – Teoria e Prática**. São Paulo: Novatec, 2005.

MICROSOFT. **Visual Studio Code**. Acesso em: 5 jan. 2025. 2024. Disponível em: https://code.visualstudio.com/.

Referências III

MICROSOFT. **Visual Studio Professional**. Acesso em: 8 jan. 2025. 2025. Disponível em: https://visualstudio.microsoft.com/vs/professional/.

NICOLODI, A. **VISUALG 3.0**. Acesso em: 5 jan. 2025. 2024. Disponível em: https://visualg30.yolasite.com/.

RSTUDIO. **RStudio**. Acesso em: 8 jan. 2025. 2025. Disponível em: https://posit.co.

SUBLIME. **Sublime Text**. Acesso em: 5 jan. 2025. 2024. Disponível em: https://www.sublimetext.com/.

UNIVALI. **Portugol Webstudio**. [*S. l.: s. n.*], *acessado em 2025*. Ambiente de desenvolvimento integrado para a linguagem Portugol. Disponível em: https://portugol.dev/.

VIM. **Vim - the ubiquitous text editor**. Acesso em: 5 jan. 2025. 2024. Disponível em: https://www.vim.org/.

WIKI, C2. **Hello World**. [*S. l.: s. n.*], 2025. Acesso em: 09 jan. 2025. Disponível em: https://wiki.c2.com/?HelloWorld.

Estes slides estão protegidos por uma licença Creative Commons

Este modelo foi adaptado de Maxime Chupin.

Marisangila Alves, MSc
marisangila.alves@catolicasc.org.br
marisangila.com.br

Católica de Santa Catarina

2025/1

Fundamentos de Programação

Pseudocódigo