Gestão da Tecnologia da Informação

# ALGORITMOS

1º semestre

Aula 01

Fatec Bragança Paulista
Jornalista Omair Fagundes de Oliveira

SÃO PAULO
GOVERNO DO ESTADO
SÃO PAULO SÃO TODOS

**Prof. Esp.** Paulo Henrique Leme Ramalho

**2024**

# PAULO HENRIQUE LEME RAMALHO

## GRADUAÇÃO

GESTÃO DA TECNOLOGIA DA INFORMAÇÃO - FATEC BRAGANÇA PAULISTA

## PÓS-GRADUAÇÃO

GESTÃO ESTRATÉGICA EM TI - INSTITUTO FEDERAL - BRAGANÇA PAULISTA

GESTÃO DE PROJETOS EDETI – INSTITUTO DE FORMAÇÃO BRASILEIRA

# TRAJETÓRIA PROFISSIONAL

**ANALISTA DE INFRAESTRUTURA** – TRIBUNAL DE JUSTIÇA DE SÃO PAULO (2015-2020)

**PROFESSOR DE TECNOLOGIA** – INSTITUTO FEDERAL DE SÃO PAULO (FEV. 2020 – DEZ. 2021)

**GERENTE DE PROJETOS DE TI** – EVOLUÇÃO SISTEMAS (JUL.2021 – JUL. 2023)

**PROFESSOR DE TECNOLOGIA** – FASP (FEV.2022 – DEZ. 2022)

**PROFESSOR DE TECNOLOGIA** – INSTITUTO FEDERAL DE SÃO PAULO (FEV. 2024 – JUL. 2024)

**PROFESSOR DE TECNOLOGIA** – FATEC BP (ABR.2022 – **ATUAL**)

**CONSULTOR DE TECNOLOGIA** – INDUSTRIA (2018 – **ATUAL**)

# CONTATO

**LINKEDIN**

https://www.linkedin.com/in/paulooph/

**LATTES**

http://lattes.cnpq.br/3689381830520701

**E-MAIL**

paulo.ramalho@fatec.sp.gov.br

**MICROSOFT TEAMS**

Paulo Henrique Leme Ramalho (*paulo.ramalho@fatec.sp.gov.br*)

# CONTRATO PEDAGÓGICO

**CARGA HORÁRIA SEMANAL:** 4 horas

**CARGA HORÁRIA SEMESTRAL:** 80 horas

**CRITÉRIOS PARA APROVAÇÃO:** Média Final >=  6 e Frequência >= 75%

**CRITÉRIO DE AVALIAÇÃO:** MÉDIA FINAL = (P1 * 0.3) + (P2 * 0.3) + (T1 * 0.4)

- P1 e P2 – Avaliação prática e/ou dissertativa
- T1 – Trabalho da disciplina

**METODOLOGIA DAS AULAS:** Aula expositiva e prática (exercícios de fixação)

# EMENTA

Método para desenvolvimento de algoritmos. Modularidade e abstração. Tipos de dados básicos e representações gráficas dos principais comandos em uma linguagem procedural. Expressões aritméticas, lógicas e literais. Estruturas básicas de programas (sequencia, iteração, seleção simples e múltipla). Algoritmos para manipulação de estruturas básicas.

# OBJETIVO

Analisar problemas e projetar, validar soluções computacionais para os mesmos, através do uso de metodologias, técnicas e ferramentas de programação envolvendo elementos básicos da construção de algoritmos e programas de computador.

# TECNOLOGIAS E FERRAMENTAS

**Pseudocódigo** – Portugol Studio

**Link:** *https://univali-lite.github.io/Portugol-Studio/*

## IntelliJ IDEA

**https://www.jetbrains.com/pt-br/idea/download/?section=windows**

# PLANO DE AULA

- Introdução a entrada, processamento e saída.
- Introdução a Pseudocódigo / Variáveis / Tipos primitivos / Operadores aritméticos / Operadores Relacionais e Operadores Lógicos
- Estrutura de Decisão – Se e Senão ( IF e Else)
- Estrutura de Escolha - Escolha | Caso ( Switch Case)
- Estrutura de Repetição – Faça Enquanto / Enquanto faça / Para ( While , Do While e For)
- Vetor e Matriz
- Funções

# BIBLIOGRAFIA BÁSICA

FORBELLONE, L. V., EBERSPACHER, H. F. Lógica de Programação: a construção de algoritmosASCENCIO, A. F. G, CAMPOS.

E.A.V. Fundamentos da Programação de Computadores: algoritmos, Pascal e C/C++ e Java. SãoPaulo: Longman, 2007.ARAUJO.

C.DE. Algoritmos – Fundamento e Prática. Visual Books, 2007.

# BIBLIOGRAFIA COMPLEMENTAR

FEOFILOFF, P. Algoritmos em Linguagem C. São Paulo: Campus, 2009.DOWNEY, A. Think Python. GNU free documentation License, 2008.

DOWNEY, A., ELKNER, J., MEYERS, C. Como Pensar como um Cientista da Computação, GNU free documentation Licence.

# APRESENTE-SE!

- NOME
- IDADE
- PORQUE ESCOLHEU ESTE CURSO?
- TRABALHA COM QUE?
- JÁ TEVE CONTATO COM ALGORITMOS?

Fatec Bragança Paulista Jornalista Omair Fagundes de Oliveira

SÃO PAULO GOVERNO DO ESTADO SÃO PAULO SÃO TODOS

O QUE É

# ALGORITMOS?

Fatec
Bragança Paulista
Jornalista Omair Fagundes
de Oliveira

SÃO PAULO
GOVERNO DO ESTADO
SÃO PAULO SÃO TODOS

# O QUE É ALGORITMOS?

**Algoritmo é uma sequência de passos ou regras definidos para realizar uma tarefa ou resolver um problema específico**. Em termos simples, é um **conjunto de instruções lógicas** que, quando seguidas corretamente, levam à resolução de um determinado problema ou à execução de uma tarefa específica.

Eles **são usados em programação de computadores para descrever a lógica por trás de um determinado processo ou função**. Um algoritmo pode ser implementado em diferentes linguagens de programação para automatizar a execução de tarefas.

# PSEUDOCÓDIGO

Pseudocódigo **é uma forma de representar um algoritmo utilizando uma mistura de linguagem natural e algumas construções estruturadas** que se **assemelham a uma linguagem de programação.** O termo "pseudo" indica que não é uma linguagem de programação real, mas sim uma descrição de alto nível de um algoritmo que pode ser facilmente compreendida por humanos.

O objetivo do pseudocódigo é fornecer uma representação clara e compreensível do algoritmo, independentemente da linguagem de programação específica que será utilizada para implementá-lo. Isso facilita a comunicação entre programadores, analistas e outras partes envolvidas no desenvolvimento de software.

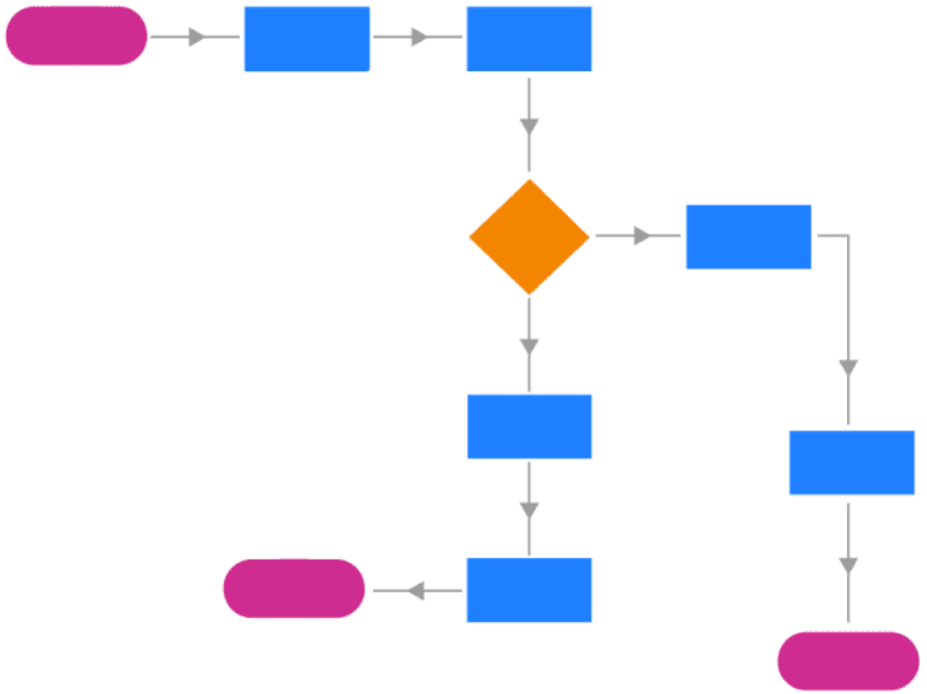
# VEJAMOS UM EXEMPLO DE ALGORITMO EM PSEUDOCÓDIGO

Vamos criar um algoritmo para fazer um sanduíche.

1. Início
2. Pegue duas fatias de pão
3. Espalhe a manteiga em uma das fatias de pão
4. Coloque uma fatia de queijo sobre a manteiga
5. Coloque uma fatia de presunto sobre o queijo
6. Adicione uma folha de alface
7. Coloque a segunda fatia de pão sobre o recheio
8. Pressione suavemente para unir as camadas
9. Fim

# FLUXOGRAMA

Um fluxograma é uma representação visual de um processo ou algoritmo. Ele utiliza símbolos gráficos para descrever as etapas sequenciais de uma operação, ajudando a visualizar e compreender a lógica do processo. Os fluxogramas são amplamente utilizados em diversas áreas, incluindo programação de computadores, engenharia, negócios, e processos industriais.

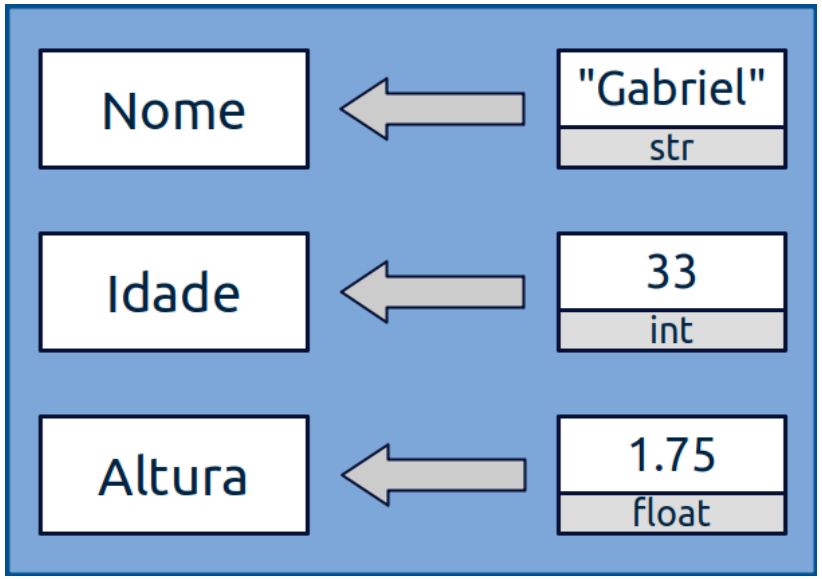


# FLUXO DE UM ALGORITMO COMPUTACIONAL

# VARIÁVEIS

**Uma variável é um espaço de armazenamento simbólico utilizado para representar e armazenar valores em um programa de computador ou em um algoritmo**. Em termos mais simples, uma variável é como uma caixa que contém informações e que pode ser identificada por um nome ou rótulo.

As variáveis são fundamentais na programação de computadores porque permitem que os programas armazenem e manipulem dados dinamicamente durante a execução. Cada variável tem um tipo que define o tipo de dados que ela pode armazenar.



# REGRAS PARA CRIAÇÃO DE VARIÁVEIS

**Nomeação:** Use nomes descritivos que indiquem claramente o propósito da variável. Evite nomes genéricos ou abreviações excessivas.

**Início do Nome:** A maioria das linguagens exigem que o nome comece com uma letra.

**Caracteres Válidos:** Evite espaços e caracteres especiais.

**Diferenciação de Maiúsculas e Minúsculas:** Em muitas linguagens, diferencia-se maiúsculas de minúsculas, então "idade" e "Idade" podem ser consideradas diferentes.

**Palavras Reservadas:** Não use palavras reservadas da linguagem como nomes de variáveis.

**Comprimento do Nome:** Mantenha os nomes de variáveis em um comprimento razoável para facilitar a leitura do código. Evite nomes muito curtos ou muito longos.

**Clareza e Legibilidade:** Escolha nomes que tornem o código claro e compreensível para você e outros desenvolvedores.

# EXEMPLO DE VARIÁVEIS

*inteiro* **idadeDoUsuario**

*caractere* **primeiraLetra**

*real* **salarioMensal**

*booleano* **sexo**

*real* **imc**

*caractere* **dataDeNascimento**

Essas são diretrizes gerais, e é importante verificar a documentação específica da linguagem que você está utilizando para garantir conformidade com suas regras específicas.

# TIPOS DE VARIÁVEIS

Os "**tipos de variáveis" referem-se aos diferentes tipos de dados que uma variável pode armazenar em programação**. Cada linguagem de programação possui uma série de tipos de dados que podem ser utilizados para representar diferentes tipos de informações. Os principais tipos (tipos primitivos) de variáveis incluem:

- Inteiro ( exemplo: 1, 65, 9.568)
- Real (exemplo: 3.14 , 0.75)
- Caracter (exemplo: ( a, A )
- Booleano (exemplo: V ou F, 0 ou 1)

# OPERADORES ARITMÉTICOS

Os operadores aritméticos são símbolos utilizados em linguagens de programação para realizar operações matemáticas entre variáveis ou valores.

**Adição ( + ) :** a+b ou 1+2

**Subtração ( - ) :** a-b ou 3-2

**Multiplicação ( * ) :** a*b ou 2*2

**Divisão ( / ) :** a/b ou 4/2

**Exponenciação ( + ) :** a^b ou 2^2

**Resto / Módulo ( % ) :** a%b ou 5%2

**Incremento ( ++ ) :** a++

**Decremento ( -- ) :** a--

# OPERADORES RELACIONAIS

Os operadores relacionais são usados em programação para comparar valores e expressões, retornando um valor lógico (verdadeiro ou falso) como resultado da comparação. Esses operadores são frequentemente usados em estruturas condicionais para controlar o fluxo de execução do programa com base em condições específicas.

# OPERADORES RELACIONAIS

| Operador | Nome | Exemplo | Resultado |
|---|---|---|---|
| == | Igual | a == b | Verdadeiro se a for igual a b |
| != | Diferente | a != b | Verdadeiro se a não for igual a b |
| <> | Diferente | a <> b | Verdadeiro se a não for igual a b |
| === | Idêntico | a === b | Verdadeiro se a for igual a b e for do mesmo tipo |
| !== | Não idêntico | a !== b | Verdadeiro se a não for igual a b, ou eles não são do mesmo tipo |
| < | Menor que | a < b | Verdadeiro se a for menor que b |
| > | Maior que | a > b | Verdadeiro se a for maior que b |
| <= | Menor ou igual | a <= b | Verdadeiro se a for menor ou igual a b. |
| >= | Maior ou igual | a >= b | Verdadeiro se a for maior ou igual a b. |

# LÓGICA BOOLEANA

A lógica booleana é um ramo da lógica matemática que trata de operações lógicas em valores binários, ou seja, valores que podem ser verdadeiros (1) ou falsos (0). Na programação, os conceitos da lógica booleana são amplamente utilizados, especialmente em expressões condicionais e controle de fluxo.

# LÓGICA BOOLEANA – AND (E)

Retorna verdadeiro apenas se ambos os operandos são verdadeiros.

Símbolo: **&&** ou **and** (dependendo da linguagem).

Exemplo: **( A && B )** é verdadeiro apenas se A e B forem verdadeiros

# LÓGICA BOOLEANA – OR (OU)

Retorna verdadeiro se pelo menos um dos operandos for verdadeiro.

Símbolo: **|| ou or** (dependendo da linguagem).

Exemplo: **( A || B )** é verdadeiro se pelo menos A ou B forem verdadeiros.

# LÓGICA BOOLEANA – NOT (NÃO)

Inverte o valor do operando, transformando verdadeiro em falso e vice-versa.

Símbolo: **! ou not** (dependendo da linguagem).

Exemplo: **!A** é verdadeiro se A for falso.

# LÓGICA BOOLEANA – TABELA

| p | q | p E q |
|---|---|---|
| V | V | V |
| V | F | F |
| F | V | F |
| F | F | V |

| p | q | p OU q |
|---|---|---|
| V | V | V |
| V | F | V |
| F | V | V |
| F | F | F |

| p | NÃO p |
|---|---|
| V | F |
| F | V |

# EXERCÍCIO DE FIXAÇÃO

**EXERCÍCIO 1:**

Crie um algoritmos que solicite o nome, e ano de nascimento, e posterior apresente esses dados na tela.

**EXERCÍCIO 2:**

Faça um algoritmos que leia seu nome, seu ano de nascimento e apresenta sua idade.

**EXERCÍCIO 3:**

Faça um algoritmo que leia 4 notas e imprima a média simples.

**EXERCÍCIO 4:**

Faça um algoritmo que leia um número e imprima a tabuada de 0 a 10.

**EXERCÍCIO 1:**

Crie um algoritmos que solicite o nome, e ano de nascimento, e posterior apresente esses dados na tela.

```
programa
{

    funcao inicio()
    {
        cadeia nome
        inteiro anoNascimento

        escreva("Informe seu nome: ")
        leia(nome)

        escreva("Informe seu ano de nascimento: ")
        leia(anoNascimento)

        escreva(nome," você nasceu em ", anoNascimento)
    }
}
```

## EXERCÍCIO 2:

Faça um algoritmos que leia seu nome, seu ano de nascimento e apresenta sua idade.

```
programa
{

    funcao inicio()
    {
        cadeia nome
        inteiro anoNascimento

        escreva("Informe seu nome: ")
        leia(nome)

        escreva("Informe seu ano de nascimento: ")
        leia(anoNascimento)

        escreva(nome," você nasceu em ", anoNascimento, " e está com ", 2024-anoNascimento, " anos")
    }
}
```

## EXERCÍCIO 3:

Faça um algoritmo que leia 4 notas e imprima a média simples.

```
programa
{

    funcao inicio()
    {
        real n1,n2,n3,n4,media

        escreva("Informe a primeira nota: ")
        leia(n1)

        escreva("Informe a primeira nota: ")
        leia(n2)

        escreva("Informe a primeira nota: ")
        leia(n3)

        escreva("Informe a primeira nota: ")
        leia(n4)

        media = (n1+n2+n3+n4)/4

        escreva("A média final é: ", media)
    }
}
```

## EXERCÍCIO 4:

Faça um algoritmo que leia um número e imprima a tabuada de 0 a 10.

```
programa
{

    funcao inicio()
    {
        inteiro n

        escreva("Informe a primeira nota: ")
        leia(n)

        escreva("\n" , n, "x 0 = ", n*0)
        escreva("\n" , n, "x 1 = ", n*1)
        escreva("\n" , n, "x 2 = ", n*2)
        escreva("\n" , n, "x 3 = ", n*3)
        escreva("\n" , n, "x 4 = ", n*4)
        escreva("\n" , n, "x 5 = ", n*5)
        escreva("\n" , n, "x 6 = ", n*6)
        escreva("\n" , n, "x 7 = ", n*7)
        escreva("\n" , n, "x 8 = ", n*8)
        escreva("\n" , n, "x 9 = ", n*9)
        escreva("\n" , n, "x 10 = ", n*10)
    }
}
```