# Intro à Programação e Pensamento Computacional

**Pensamento Computaciona**l

Processo envolvido na expressão de soluções em passos ou algoritmos que podem ser implementados no computador

A solução encontrada através do pensamento computacional deve ser resolvida tanto por uma máquina como um humano.

É baseado em 4 pilares:  
Decomposição > Reconhecimento de padrões > Abstração > Design de algoritmos

Decomposição: Dividir um problema complexo em subproblemas

Estratégias para decompor

- análise: estudar, explorar detalhadamente para decompor

- síntese: processo de reconstrução para fundir elementos de maneira coerente para dar sentido

- sequencial ou paralelo (em “fila” ou concomitantemente)

Passos para decompor: identificar ou coletar os dados > agregar os dados > funcionalidade.

Ex.: cozinhar: identificar os ingredientes > determinar as etapas (sequencial ou paralelo) > executar cada etapa > agregar os ingredientes para finalizar (recompor com coerência)

Ex.²: Criar um app: finalidade > interface > funcionalidades > pré-requisitos

Ex.³: Elaborar um artigo: O que será abordado? > Estrutura > Conteúdo de cada tópico > Textos de conexão

Reconhecimento de padrões: Identificar padrões ou tendências (similaridades e diferenças entre os problemas; padrão comportamental)

Extrair características para extrair os dados, identificando padrões nesses dados

Usos de reconhecimento de padrões: Machine Learning, Redes Neurais, AI.

Abstração: Extrapolar o conceito do problemas para um forma generalista

Processo intelectual de isolamento de um objeto da realidade.

Design de algoritmos: Automatizar - Definir passo a passo a solução do problema

Processo de resolução de problemas “step by step” e deve ser entendido pelo humano e pela máquina

Como construir um algoritmo?  
- Compreensão do problema

- Definição dados de entrada

- Definir processamento

- Definir dados de saida

- Utilizar um método de construção

- Teste e diagnóstico

(Input > Operações > Output)

Tipos de construção de algoritmos:  
Narrativa: Utilização da linguagem natural  
Fluxograma: Estrutura de símbolos pré-definidos

Pseudocódigo: Portugol (Passo a passo a serem seguidos, regras definidas)

**Lógica de Programação**

O que é Lógica?  
Parte da filosofia que trata das formas de pensamento em geral (dedução, indução, hipótese, inferência, etc) e das operações intelectuais que visam a determinação do que é verdadeiro ou não;

É uma forma de pensamento estruturada que te auxilia a resolver problemas;

A organização de planejamento das instruções, assertivas de um algoritmo, a fim de viabilizar a implantação de um programa.  
  
Técnica Linear  
Segue uma sequência linear de etapas

Técnica Estruturada  
Define uma hierarquia de prioridades, não é necessariamente linear

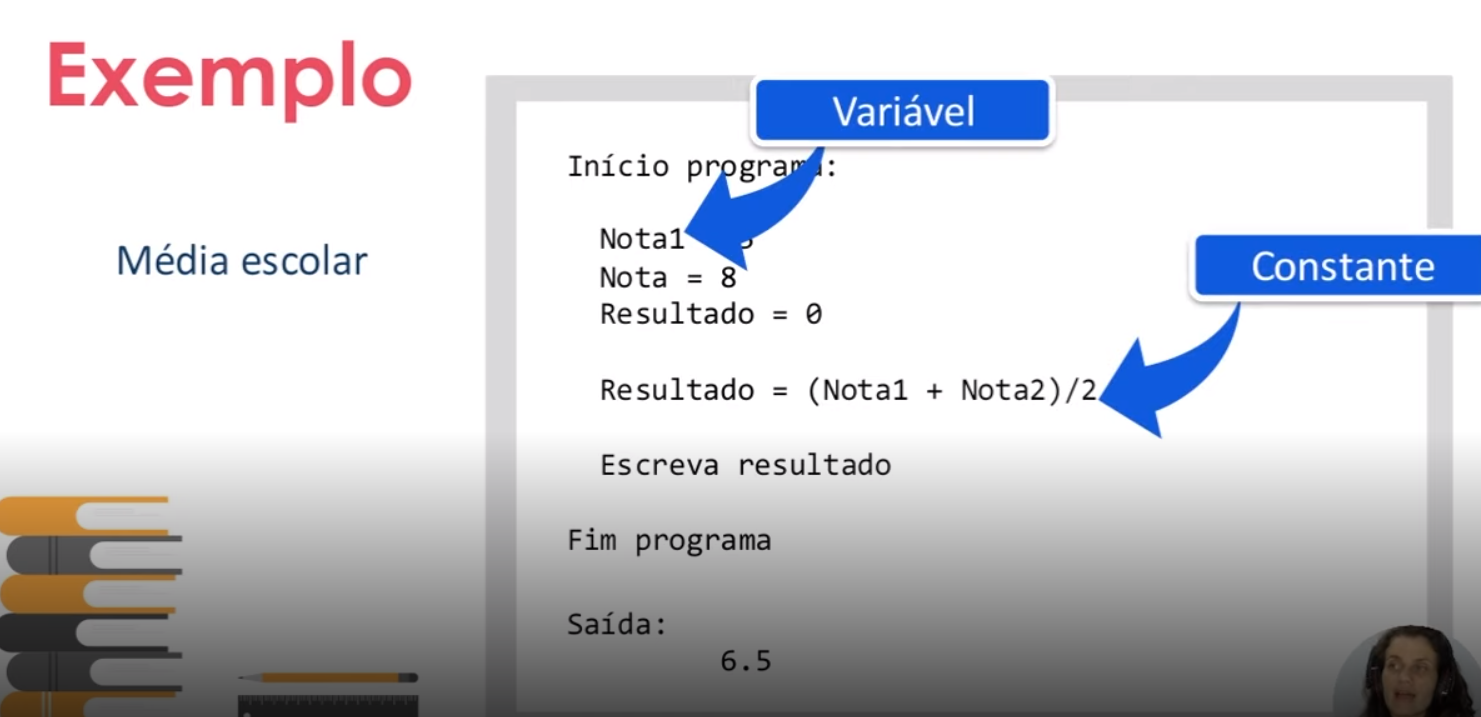
Técnica Modular

Separação das etapas por módulos com regras definidas para cada módulo

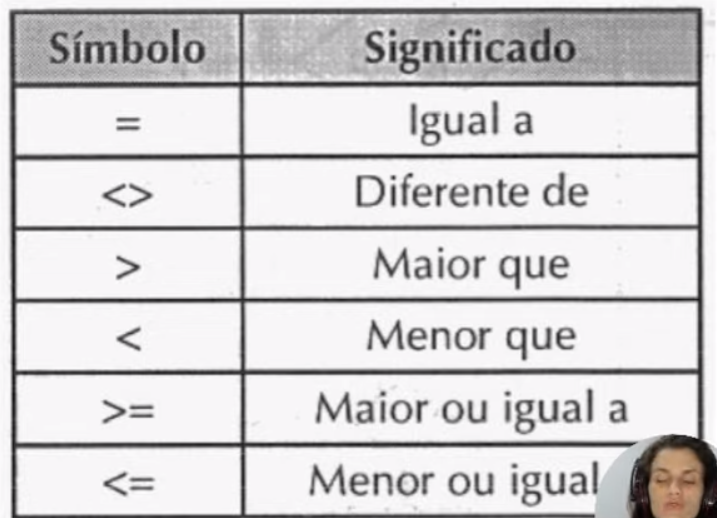
**Fundamentos de Algoritmos**

Tipologia e variáveis

Instruções primitivas - Estrutura sequencial:



Estrutura Condicional:  
Expressa uma condição que se satisfeita, executa uma determinada operação, caso a condição não seja satisfeita acarreta em uma exceção.  
A estrutura pode ser simples, composta (se, senão) e encadeada.

Operadores Relacionais:  


Estruturas de repetição:

Irá executar um certo trecho de um programa a partir de parâmetros estabelecidos dentro da estrutura de repetição.

Condição de parada para parar a repetição.

Vetores e matrizes:

Funções:  
São blocos de instruções que realizam tarefas especificas.

**Linguagens de Programação**

Introdução a linguagens de programação:  
Alan Turing, “pai da computação” a partir de 1940 definiu vários conceitos que são utilizados até hoje na computação.

Em 1975 foi criada a linguagem basic e o primeiro mini computador pela Microsoft e seguida pela Apple em 1976.

A primeira linguagem de programação (linguagem de máquina) surgiu em 1949: Assembly e a partir dos anos 50 começaram a surgir outras linguagens de programação para solucionar problemas computacionais e foram evoluindo até os dias atuais.

Como o computador entende um programa?

Através de um processo de tradução (compilação através de um compilador) onde o Programa Fonte é enviado para o Compilador e após traduzido é um Programa Objeto em que o computador entende.

Características de um programa:

Legibilidade, redigibilidade, confiabilidade e custo

Análises de código:

Léxica: Leitura do programa fonte, letra por letra e agrupar em símbolos léxicos, conhecidos como tokens, classificando e eliminar o que não é necessário

Sintática: A forma que ele define através de palavras reservadas, indexação, símbolos especiais, é a corretude do programa.

Semântica: É a lógica do programa, o que dá significado ao programa, incidindo sobre significantes (palavras, formas, símbolos).

Paradigmas da programação:  
A forma de resolução de problemas com diretrizes e limitações específicas de cada paradigma na linguagem da programação.

Paradigma estruturado ( C ): Ênfase em sequência

Utilizado em programas simples e diretos

Orientação a Objeto: Baseado na utilização de objeto e suas interações

**Primeiro contato com Programação**

Algoritmos em portugol:

Intro ao Git e Github

**Introdução ao Git**

O que é e sua importância:

Criado em 2005 pelo Linus Torvalds para solucionar problemas do Linux de forma colaborativa. É um software de versionamento de código, aprimorando outros versionadores menos populares na época.

Git não é Github!

Benefícios destas tecnologias:

* Controle de Versão
* Armazenamento em nuvem
* Trabalho em equipe
* Melhorar seu código
* Reconhecimento

**Navegação via command line interface e instalação**

Comandos básicos para um bom desempenho no terminal:  
GUI (Graphic User Interface, interface gráfica) x CLI (Command Line Interface, comando por linhas, sem interface gráfica)

Windows x Unix

- cd - cd Muda o diretório, para navegar entre as pastas, após o comando digitar a pasta

- dir - ls Lista todas as pastas e arquivos do diretório onde estamos situados

- mkdir - mkdir Cria pastas (diretórios), digitar o nome na sequencia

- del / rmdir - rm / -rf Del/rm para deletar arquivos e rmdir/rf para repositorios

- cls - clear (Ctrl+L) Limpar a tela do terminal

Atalhos:

Tab - Auto completa os nomes das palavras/diretórios, etc

Como o Git funciona por baixo dos panos:  
SHA1 (Secure Hash Algoritm)

Algoritmo de encriptação que gera um conjunto único de caracteres de 40 dígitos

Objetos fundamentais

* Blobs -
* Trees
* Commits

Sistema distribuído

Segurança

Chaves SSH & Tokens:

**Iniciando Git e primeiros comandos**

Criando um repositório:

git init inicializa um repositório vazio dentro da pasta

Ciclo de vida dos arquivos:  
Git init = Cria um repositório no Git dentro da pasta escolhida

Tracked ou Untracked = 