**PRIMEIROS PASSOS PARA COMEÇAR A PROGRAMAR**

**HISTÓRIA DA COMPUTAÇÃO**

**Introdução:**

* Como ciência, a Computação é uma área de avanços cumulativos, baseada em pesquisas anteriores.
* Nessa área surgem novos paradigmas - novas maneiras de se resolver problemas
* Primeiro evoluiu-se o hardware e depois o software
* O objetivo da computação é minimizar esforços e otimizar recursos.
* A história da computação se mistura com a da matemática
* O surgimento dos primeiros computadores ocorreu durante a Guerra tendo avanços exponenciais

**O Berço da Computação:**

* Pai da computação: Alan Turing - Com ele se inicia a revolução computacional.
* Alan Turing definiu o conceito de algoritmo
* Ajudou a decifrar os códigos de guerra
* Abordou termos como inteligência artificial e padrão binário
* Claude Shannon: percebeu a vasta aplicabilidade da álgebra booleana
* Desenvolveu conceitos e a lógica atreladas a base dos computadores digitais -> que são a base dos computadores atuais
* Dissertou sobre compressão de dados e criptografia
* Von Newmann: Teve um papel vasto no desenvolvimento da ciência da computação, contribuindo com diversas áreas - Construção de Hardware, Programação, Arquitetura, Teoria da Computação e Computação Científica.
* Utilizou o mesmo espaço de memória da máquina para armazenar dados e programas e poder executar esse programa - conhecida como Arquitetura de Von Newmann.

**Instrumentos Computacionais:**

* Blaise Pascal - Máquina que realizava as operações de + e -
* Ada Lovelace 1ª programadora -Patrona da Ciência da Computação
* Babbage - Calculador analítico

**Primeiros Computadores:**

* Z1, Z2, Z3 e Z4 - Konrad Zuse
* Mark I - IBM e Harvard
* Eniac -Criado na segunda guerra - 1º computador programável
* Edvac -Von Newmann - Traz o conceito de programa armazenado
* Colossus - Alan Turing

**Era dos computadores pessoais:**

* 1975 - Altair 8800 usando o chip intel 8080. Era um mini computador
* Bill Gates e Paul Allen ciram a linguagem de programação Basic usada para os mini computadores. Mais tarde fundam a Microsoft
* 1976 - Appel cria seu primeiro computador
* 1981 - IBM lança o IBM PC

**INTRODUÇÃO À LÓGICA DA PROGRAMAÇÃO**

**O que é Lógica em Programação?**

* Lógica visa a resolução de um problema
* Problema é uma questão que foge a uma determinada regra. ou melhor é um desvio de percurso, o qual impede de atingir um objetivo com eficiência e eficácia
* Lógica é parte da filosofia que trata das formas do pensamento em geral (dedução, indução, hipótese, inferência etc.) e das operações intelectuais que visam determinação do que é verdadeiro ou não
* Lógica ordenação que segue convenções (as regras associadas a essa ordenação devem ser obedecidas para se resolver determinadas questões ).
* Lógica é uma organização coesa de instruções a fim de se alcançar um objetivo
* Lógica está associada a forma como se desencadeiam acontecimentos
* Lógica é uma forma de raciocínio
* Lógica na programação: organização e planejamento das instruções , assertivas em um algoritmo, a fim de viabilizar a implantação de um programa.
* Seres humanos podem prever comportamentos, computadores não. Por isso é necessário detalhar instruções para o computador. Passo a passo por meio do algoritmo

**Técnicas de lógica de programação:**

* Técnica Linear:
* É um modelo de desenvolvimento e resolução de problemas
* Modelo tradicional
* Não tem vínculo com nenhuma estrutura hierárquica ou com a programação de computadores.
* Execução sequenciada de eventos
* Ordenação os elementos por uma única propriedade (ex. acordar e tomar café)
* Tem recursos limitados
* Técnica Estruturada:
* Estrutura - Organização, disposição e ordem dos elementos essenciais que compõe um corpo (concreto ou abstrato)
* Dá uma ideia de hierarquia, seguindo uma estrutura
* O objetivo desse tipo de técnica é facilitar a escrita, o entendimento, a manutenção e a validação de um ou mais programas
* É uma técnica indicada para processamento de dados - o programa irá processar dados
* Técnica Modular:
* Decompor um problema por meio de módulos- partes independentes
* Essas partes são controladas por um conjunto de regras
* As metas desse tipo de técnica são: simplificação, decompor o problema, verificação do módulo
* Modelo padrão do desenvolvimento de software:

Dados de entrada - processo de transformação - dados de saída

**ALGORITMOS**

**O que é um algoritmo?**

* O computador não opera sozinho. Precisa de instruções detalhadas para funcionar
* O objetivo do computador é o processamento de dados - receber, manipular e armazenar dados - isso é feito por meio de programas.
* **Algoritmos**: é um processo de resolução de problemas passo a passo utilizando instruções
* Instruções - o que precisa ser feito / qual a ordem de execução?
* O Algoritmo precisa ser entendido tanto pelo ser humano quanto pela máquina
* O desenvolvimento de um programa segue fases:
* Análise: Estudo e definição dos dados de entrada e saída
* Algoritmo:Descreve o problema por meio de ferramentas narrativas, fluxograma ou pseudocódigo
* Codificação: O algoritmo é codificado de acordo com a linguagem de programação escolhida
* **Algoritmos**:
* Sequência de passos com objetivo definido
* Execução de tarefas específicas - objetivas e precisas
* Conjunto de operações que resultam em uma sucessão finita de ações
* Exemplos de algoritmos no dia a dia: (execução passo a passo para concluir uma tarefa)
* Preparar um sanduíche -separar dados - execução
* Trocar uma lâmpada
* Fazer uma receita de bolo
* Trajeto até o trabalho

**Construção de algoritmos:**

* Como construir um algoritmo?
* Compreensão do problema - pontos mais importantes
* Definição de dados de entrada - dados fornecidos e cenários
* Definir processamento - cálculos e restrições, como tratar os dados
* Definir dados de saída - dados posteriores ao processamento
* Utilizar um método de construção e Teste e diagnóstico - construção e refinamento do algoritmo
* Ferramentas para construção de um algoritmo:
* Narrativa: Utiliza a linguagem natural. Vantagem: fácil entendimento Desvantagem: pode ter várias interpretações possíveis.
* Fluxograma: Utiliza símbolos pré definidos para sequenciar as instruções. É mais visual. Vantagem: Simples entendimento. Desvantagem: Necessita de um prévio conhecimento da estrutura e dos símbolos
* Pseudocódigo: Portugol. São regras definidas, passos a serem seguidos. -Mais próximo de uma codificação

**FUNDAMENTOS DE ALGORITMOS:**

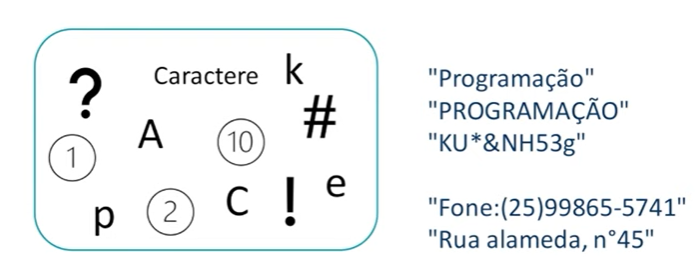
**Tipologia e variáveis**

* Qual a função do computador?
* Utilizado para a resolução de problemas por meio de informações
* As informações podem ser divididas em dados e instruções
* Os **dados** são as informações tratadas e processadas
* Os dados se classificam em 3 tipos:

- **Numérico:**

****

- **Caracteres:**

****

- **Lógicos**



* **Variável:**
* Pode assumir qualquer um dos valores de um determinado conjunto de valores
* Precisa ser previamente identificada e especificar seu tipo
* Regras para determinar o nome de uma variável:

- Atribuição de um ou mais caracteres

- Primeira letra - não número

- Sem espaços em branco

- Vedada a utilização de palavras reservadas Ex.: print

- Evitar o uso de caracteres especiais

- Pode-se utilizar caracteres e números

- Os papéis de uma variável são de ação e controle:

Ação: vai modificar um estado, as variáveis se modificam durante a execução do programa

Controle: a variável vai ser vigiada, auxilia no controle, pode ser usada em estruturas de repetição

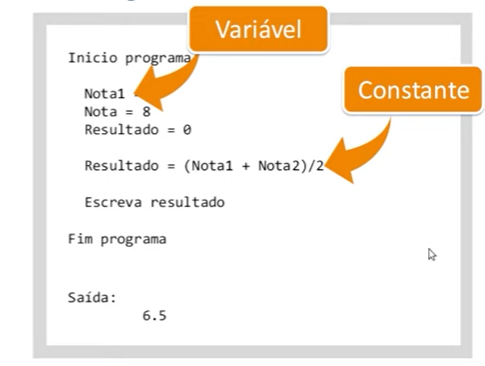
* **Constante:**
* Fixa es estável, valor inalterável. Ex: Pi = 3,14
* **Instruções:**
* São operações que irão determinar o tratamento e como os dados serão processados

**Instruções primitivas**

* Operadores:



* As **instruções** são linguagem de palavras-chave (vocabulário) de uma determinada programação que tem por finalidade comandar o computador que irá tratar os dados



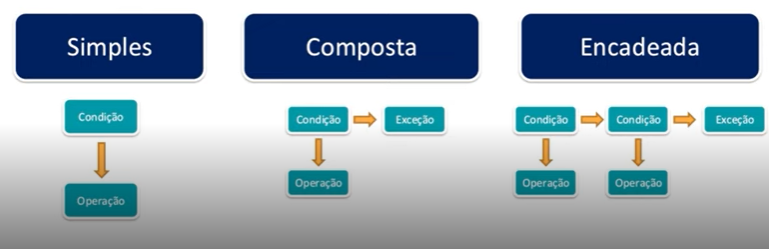
**Estruturas condicionais e operadores:**

* Condição: estado de uma pessoa ou coisa - na programação estado de uma variável
* Condicional: expressão de uma condição ou suposição - contém ou implica uma suposição ou hipótese - verifica a condição

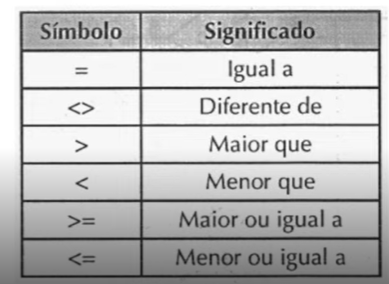
**Estrutura condicional:**



**Tipos de estrutura condicional:**



Dentro da estrutura condicional utilizamos os operadores **relacionais**

****

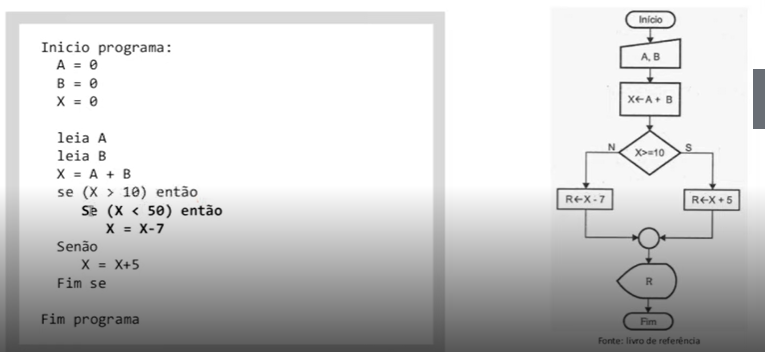
**Exemplo de estrutura condicional simples:**

****

**Exemplo de estrutura condicional composta:**

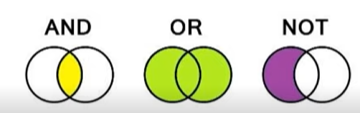
****

**Exemplo de estrutura condicional encadeado:**

****

**Operadores lógicos**

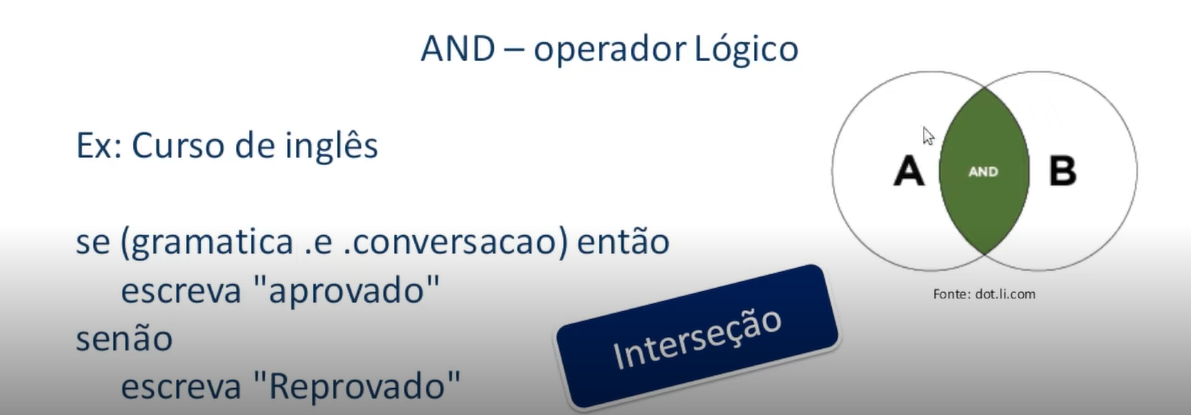
* Operadores relacionados a lógica booleana e eliminam ações



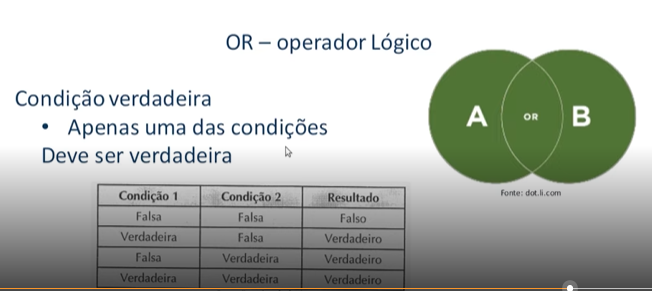
* São utilizados na verificação de V ou F
* E na substituição e encadeamento de condições

Operador AND





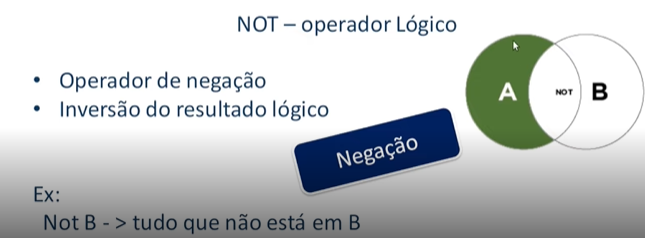
Operador OR





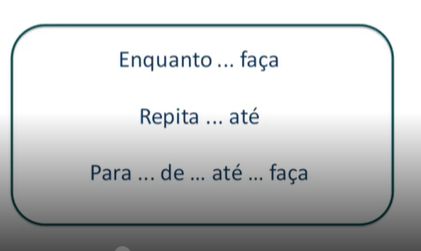
Operador NOT





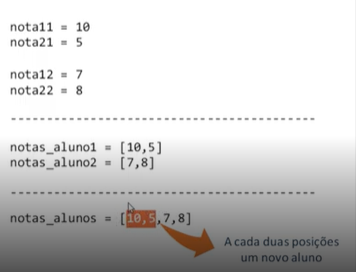
**Estruturas de repetição**

* Também nomeada como controle de fluxos, laços, malha de repetição, loop, repetição
* Executa uma mesmo trecho de código/ programa um determinado número de vezes predeterminadas
* Para que o trecho não seja executado infinitamente são definidas condições de paradas
* Condições de parada podem ser:
* Número de repetições pré definidas
* Uma condição a ser satisfeita
* Pontos positivos da estrutura de repetição:
* Redução de linhas
* Compreensão facilitada
* Redução de erro
* Tipos de estruturas de repetição:



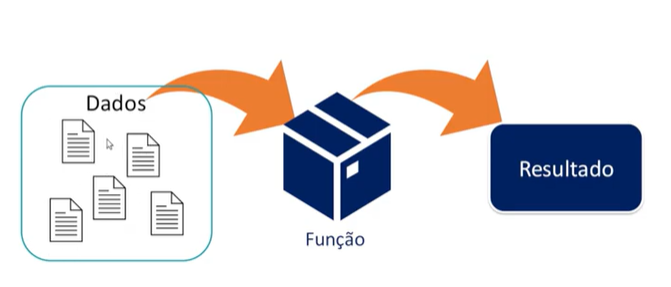
**Vetores e Matrizes**

* Um vetor é caracterizado por uma variável dimensionada com valor pré-fixado
* O vetor recebe uma sequência de valores que possuem posições relacionadas a eles que podem ser acessadas
* Matriz é uma uma tabela organizada em linhas e colunas no formato m x n, onde o m representa o número de linhas(horizontal) e o n representa o número de colunas(vertical)
* Matriz é uma coleção de variáveis
* A matriz e os vetores são utilizados quando há uma grande quantidade de dados que precisam ser representados



**O que são funções**

* Funções também pode nominada como: função, bloco, subprograma, subrotina, subalgoritmo, método
* A função em programação é similar a função matemática
* A função recebe valor/dados e associa essas valores ao seu retorno
* As funções são blocos de instruções que realizam tarefas específicas
* As funções trazem uma ideia de decomposição do algoritmo
* A função permite que sejam reutilizadas instruções - chamar a função
* Código fica mais limpo e conciso
* Permite a modularização do programa
* As funções são blocos de instruções(códigos) identificados por **nomes e parâmetros**



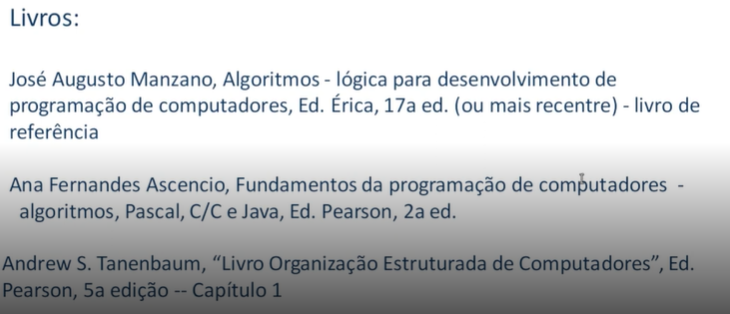
* A função tem a capacidade de alterar o estado do programa

**Instruções de entrada e saída**

* Os dados precisam ser processados e tratados para gerar um resultado
* Entrada: Inserção e recebimento de dados do mundo real por meio de ação de alguma interface , seja teclado, mouse, arquivos, etc.
* Saída: Consiste na impressão dos dados do mundo abstrato, digital por meio de ação de alguma interface
* Saída programada: Condicional(aguarda o dispositivo estar preparado para receber a informação) ou incondicional(envia os dados independente do estado do dispositivo)
* Saída por interrupção



Referências



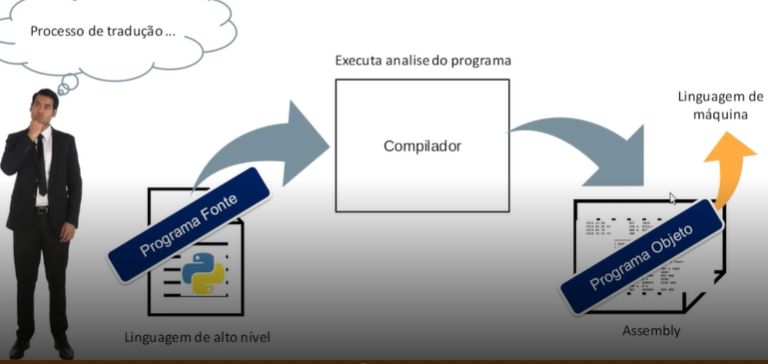
**Linguagens de Programação**

**Introdução à linguagens de programação**

* 1949 1ª linguagem - Assembly - linguagem de baixo nível, próxima ao hardware
* Década de 50 - Cobol, Fortran, Lisp
* Década 60 e 70 - C, Prolog, ML
* Década de 90, surgem as linguagens de alto nível - Java, Python, C#, Ruby, Javascript
* Anos 200, surgem linguagem de multiparadigmas - linguagens especializadas - Go, Rust, Dart, TypeScript
* Problema Computacional: Objeto de discussão que possui instruções passo a passo que são mais facilmente resolvíveis em ambiente computacional
* Tipos de problema: Problema de busca(buscar algo, busca/relação binária), problema de otimização(maximizar ou minimizar uma função) e problema de decisão(verdadeiro ou falso)
* O que é uma linguagem de programação? - Método padronizado composto por um conjunto de regras sintáticas e semânticas de implementação de um **código-fonte**
* Código-fonte - conjunto de palavras com regras
* Código-fonte pode ser: Traduzido ou interpretado

**Como um computador entende um programa:**

* Processo de tradução:



Tradução: 1. Geração do programa objeto / 2. Execução do programa objeto

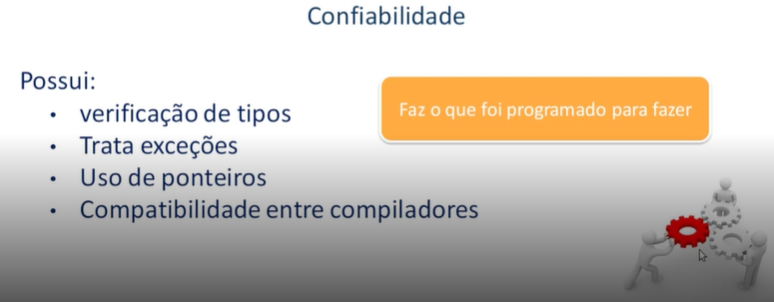
Interpretação: Programa fonte executado diretamente

**Características de um programa:**

* 



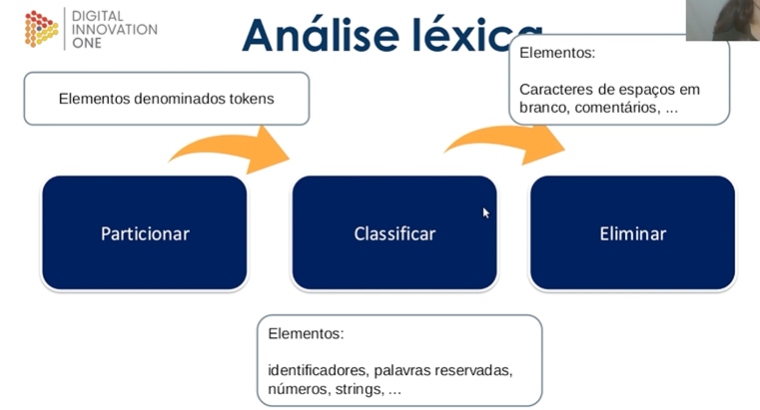


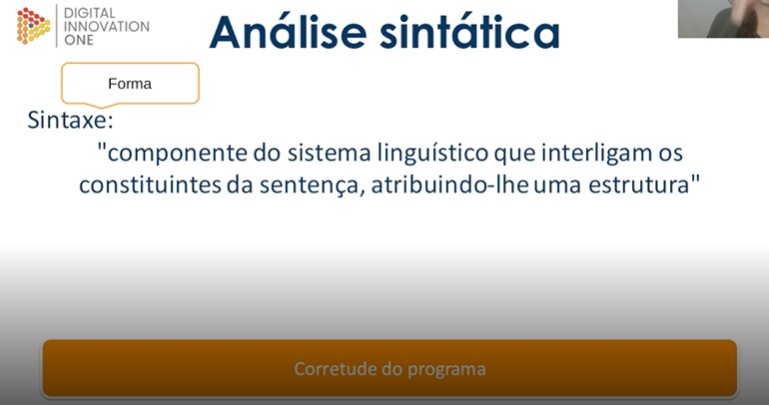




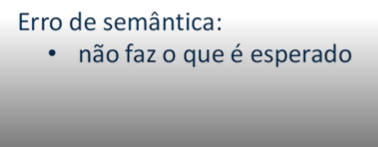
**Análises de código:**

* Análises que o compilador realiza no código









**Paradigmas de programação:**

* O que é um paradigma? - Forma de resolução de problemas com diretrizes e limitações específicas de cada paradigma utilizando linguagem de programação

Tipos de paradigma:

