**Linguagens de programação**

* Introdução as linguagens de programação

Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

O hardware limita o potencial do software. Conceitos como a blockchain, por exemplo, são consideravelmente mais antigos, porém só poderiam ser implementados com poder computacional suficiente.

Tabela

Descrição gerada automaticamente

Instruções que invocavam conceitos utilizados na linguagem de máquina, porém com mais legibilidade para humanos.

Linha do tempo

Descrição gerada automaticamente

Diversas linguagens e conceitos surgiram ao longo do tempo. Sempre buscando oferecer soluções para problemas distintos. Em geral, não existe uma considerada faz-tudo.

Linha do tempo

Descrição gerada automaticamente

Problemas variam de acordo com a época computacional.

Tela de celular com texto preto sobre fundo branco

Descrição gerada automaticamente

Responder se sim ou se não. Dado um número (o que está escondido).

Uma imagem contendo Gráfico

Descrição gerada automaticamente

Uma imagem contendo Interface gráfica do usuário

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

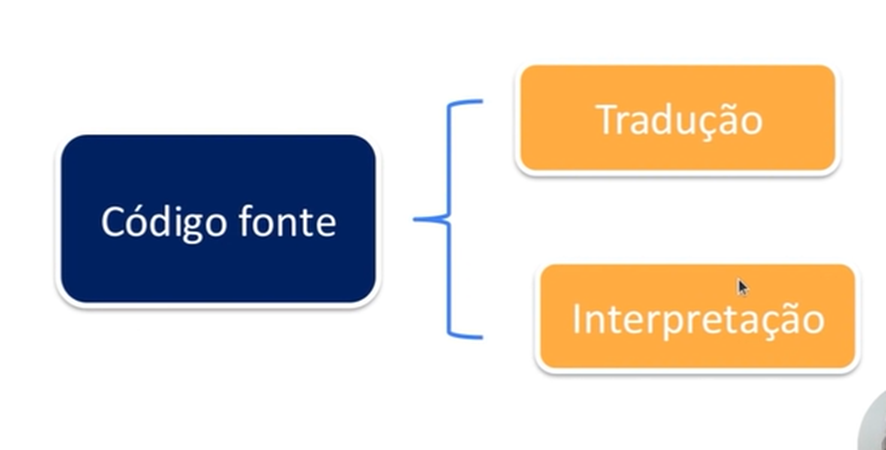
Descrição gerada automaticamente

É uma forma de codificar um algoritmo.

Diagrama

Descrição gerada automaticamente

* Como um computador entende um programa?



Uma linguagem de alto nível gera esse código fonte.

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente



Diagrama

Descrição gerada automaticamente

Diagrama

Descrição gerada automaticamente

Visão simplificada do funcionamento de um compilador, passando por diversas análises até gerar um código (programa objeto, linguagem de baixo nível).

Diagrama

Descrição gerada automaticamente

Interpretar faz o computador compreender o código fonte sem passar por um compilador, logo é mais lento do que simplesmente traduzir e fazer a máquina interpretar o código traduzido.

Diagrama

Descrição gerada automaticamente

É possível observar as vantagens. Linguagens que utilizam de interpretação são geralmente, mais flexíveis e fáceis de compreender, mas tomam mais tempo para processar.

Existe também a **Transpilação**: Linguagem de alto nível -> linguagem de alto nível mas um pouco mais próxima de uma de baixo nível (ex: TypeScript -> JavaScript).

Diagrama

Descrição gerada automaticamente com confiança média

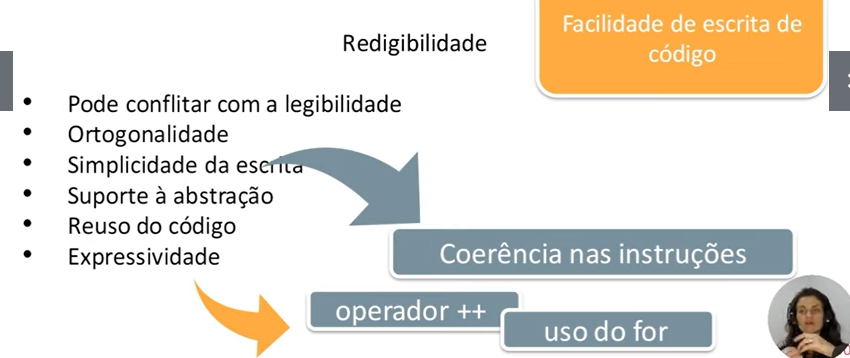
* Características de um programa

Diagrama

Descrição gerada automaticamente

Diagrama

Descrição gerada automaticamente



**Ortogonalidade** em uma linguagem de programação significa que um conjunto relativamente pequeno de construções primitivas pode ser combinado a um número relativamente pequeno de formas para construir as estruturas de controle e de dados da linguagem. Uma falta de ortogonalidade leva a exceções às regras de linguagem.

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Em programação, um ponteiro ou apontador é um tipo de dado de uma linguagem de programação cujo valor se refere diretamente a um outro valor alocado em outra área da memória, através de seu endereço. Um ponteiro é uma simples implementação do tipo referência da Ciência da computação.

Uma imagem contendo Texto

Descrição gerada automaticamente

Essencialmente, é pensar em como os recursos estão sendo utilizados. Se o código está eficiente para algum cenário etc.

Interface gráfica do usuário, Diagrama, Texto

Descrição gerada automaticamente

Atualizações: algo novo, um framework por exemplo, surgiu. É importante buscar estar sempre atualizado nas novidades da TI.

IA: Procurar implementar conceitos de IA, como o machine learning (python possui bibliotecas que usam desse conceito, por exemplo).

Comunidade ativa: buscar ajuda de outros programadores para melhorar e resolver problemas no código (Stack Overflow, por exemplo).

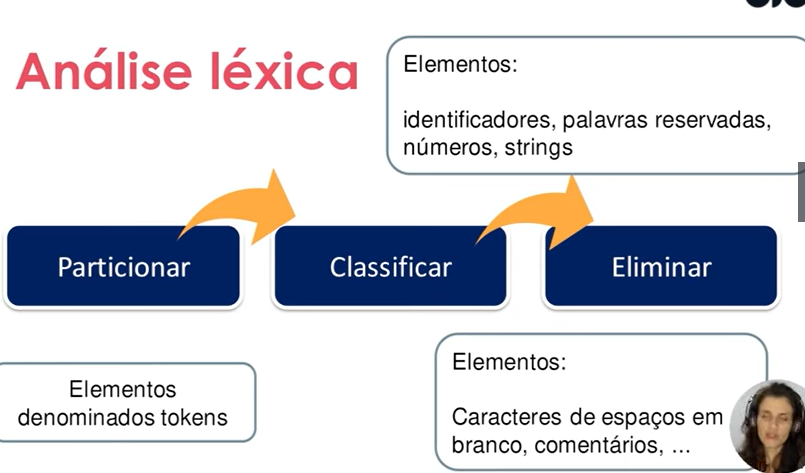
Adoção pelo mercado: Tomar observação das necessidades do mercado. Quais linguagens estão sendo mais utilizadas, qual framework faz sucesso etc.

* Análises de código

Diagrama

Descrição gerada automaticamente

Nessa parte do processo de codificação, temos essas operações que ocorrem dentro do compilador (podemos dizer que é aí que a mágica ocorre).



Ler o programa fonte, caractere por caractere, letra por letra e agrupar os caracteres em lexemas, sequências de símbolos léxicos, conhecidos como tokens.

Particionar, classificar e eliminar: Analisar e agrupar os elementos que serão chamados de tokens (identificadores, palavras reservadas, números, strings, todo conteúdo relevante para a codificação). E eliminar elementos que são importantes para o entendimento de um humano, mas são inúteis para uma máquina.

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto, Email

Descrição gerada automaticamente

A sintaxe de um programa é a forma que ele define, através de palavras reservadas, de indexação, de símbolos especiais, etc, qual é a estrutura relacionada para codificação dentro daquela linguagem específica (corretude do programa).

Uma imagem contendo Texto

Descrição gerada automaticamente

Cada linguagem de programação terá uma sintaxe associada.

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente com confiança baixa

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Estudo do significado propriamente dito. Nesse exemplo “=” foi utilizado como atribuição, “==” foi utilizado para comparação. Usando o operador.

1. = (operador de atribuição) - é usado para atribuir valores a uma variável (em JavaScript, por exemplo).

**x = 10**   
**y = 20  
z = x + y**

1. == (Igual a ou Equal to) - é usado para comparação entre duas variáveis, independentemente do tipo de dados da variável (em Javascript por exemplo).

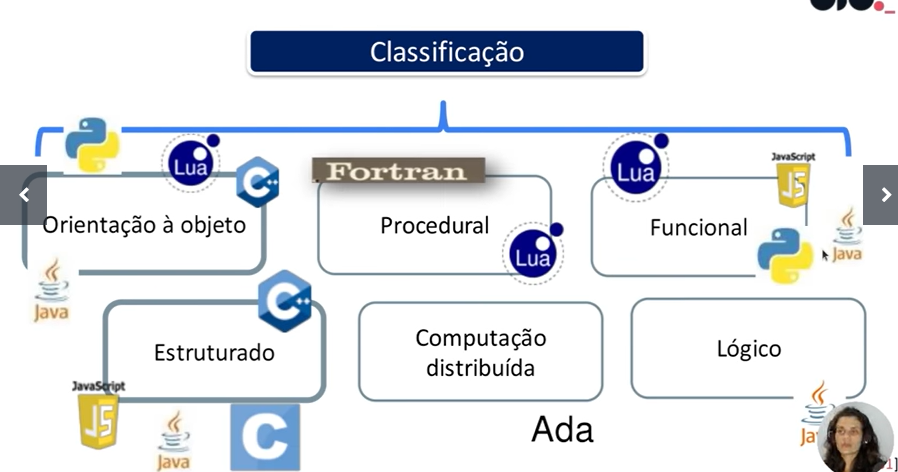
Para **x= 10** temos que :

**x ==     8**  -> retorna false  
**x ==   10**   -> retorna true  
**x ==  "10"** -> retorna true

* Paradigmas de programação

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente



Procedural: chamadas sucessivas e procedimentos separados (ideia de sequência).

Funcional: Instruções baseadas em funções.

Estruturado: estrutura de blocos aninhados.

Computação distribuída: Funções executadas de forma independente (seu programa deve ter módulos independentes.

Lógico: Também é conhecido como “restritivo”. Muito utilizado em aplicações de inteligência artificial. Esse paradigma chega no resultado esperado a partir de avaliações lógico-matemáticas.

Interface gráfica do usuário, Texto, Tabela

Descrição gerada automaticamente com confiança média

Sequência: todas as instruções são executadas em sequência, independentemente de aspectos, como estruturas condicionais.

Decisão: teste lógico.

Iteração: Funções, laços, condições (dominou o mercado até a POO (programação orientada a objetos ).

Texto

Descrição gerada automaticamente

Conceitos básicos, como pilhas, são programados em C.

Texto

Descrição gerada automaticamente

Interface gráfica do usuário

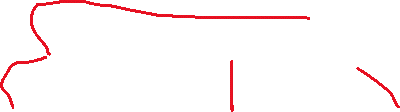
Descrição gerada automaticamente

Uma imagem contendo Diagrama

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente



Caracteristicas: atributos/Comportamentos: métodos

Diagrama

Descrição gerada automaticamente com confiança média

Diagrama

Descrição gerada automaticamente com confiança média

Interface gráfica do usuário, Diagrama

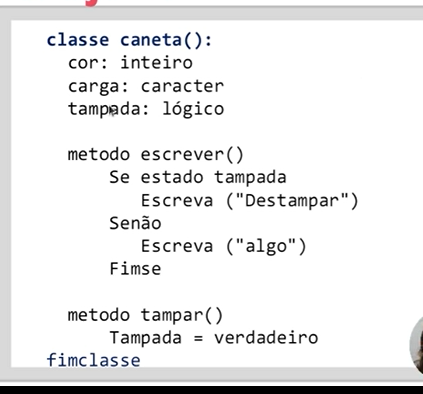
Descrição gerada automaticamente

Diagrama, Texto

Descrição gerada automaticamente

No POO a classe aloca na memória uma instância dessa classe com objeto, e possui operações associadas, que são os métodos. Variáveis, métodos e estado estão relacionados a um objeto e somente a uma classe.

Já na estruturada, é perceptível que as operações estão desassociadas da variável.



Exemplo.

Interface gráfica do usuário, Diagrama

Descrição gerada automaticamente

Diagrama

Descrição gerada automaticamente

Herança. Classes mãe tendem a ser mais gerais, classes filhas tendem a ser mais específicas.

Diagrama

Descrição gerada automaticamente

Códigos associados podem ajudar bastante no reuso do código, pois instanciar um novo objeto permite tal situação devido a código prévio que já definiu aspectos.

Já o Paradigma Estruturado, ataca os problemas específicos e diretos. Por exemplo, análise de dados com python (mais lento) ou C (melhor performance).

No geral, analisar o problema é essencial para definir a linguagem a ser aproveitada.