**Linguagens de Programação**

**Por que existem linguagens de programação e porque são tantas?**

* Máquinas não entendem a linguagem humana suficientemente para que consigam processar informações conforme desejamos. E mesmo que entendessem, seria difícil para os humanos externalizar em sua própria linguagem, o que esperam que as máquinas façam de forma lógica. **Dependendo da aplicação, uma ou outra linguagem torna-se mais adequada em função da tecnologia, investimentos, treinamentos, época, etc.**

**Algumas linguagens**

**BASIC**: é uma linguagem historicamente importante que ajudou a popularizar a prática de programação. Alguns dos primeiros computadores pessoais vinham com a linguagem BASIC instalada no hardware convidando novos usuários a começar a programar. Várias derivações do Basic estão ou estiveram disponíveis, entre elas Small Basic, Visual Basic, entre outras.

**COBOL**: muitos sistemas na área bancária foram escritos em Cobol e permanecem em uso até hoje. A formação em COBOL não é comum e por isso mesmo os programadores em COBOL costumam ser muito valorizados.

**PYTHON**: é considerada uma das linguagens mais fáceis, quando não a mais fácil de aprender. Com uma sintaxe simples e extremamente legível, torna o desenvolvimento muito direto. Machine Learning e Extração de Dados são implementados com PYTHON.

**ASSEMBLY**: é a linguagem que leva à alta performance da máquina ainda em padrão legível pelas pessoas. É usada em partes de um programa muito sensíveis à performance.  Encontrada em sistemas operacionais e “engine” de jogos, por exemplo.

**C**: talvez a linguagem de programação mais importante do mundo. Sistemas operacionais tais como Windows, MacOS, iOS, e Android são escritos nela, bem como navegadores e “engine” de jogos. Influenciou dezenas de outras linguagens. Próxima da linguagem Assembly Language permite alta performance do software.

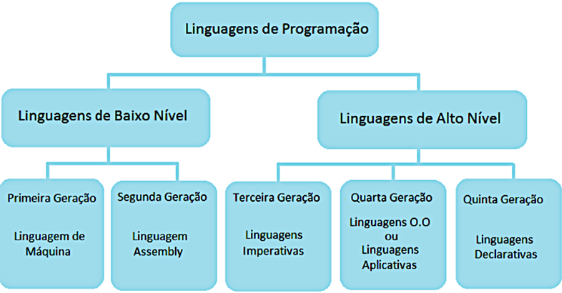
**JAVA**: foi desenvolvida pela SUN Microsystems no início de 1990. É “orientada a objetos”, portável e escalável. Têm muitas semelhanças com C e C++. É a base de diversos sistemas tais como o Android, possuindo uma comunidade forte e ativa na Internet. Muitas empresas têm sistemas desenvolvidos em Java, fazendo com que a procura por profissionais que saibam a linguagem ainda seja expressiva.

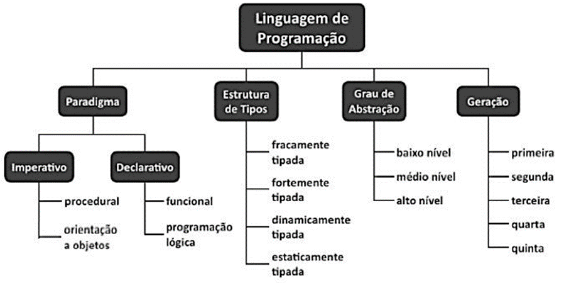
**RUBY**: sintaxe simples e fácil. Inspirada em linguagens como Perl, Smalltalk, Eiffel, Ada e Lis. Muito popular entre as startups, é famosa por ser usada em aplicações mundialmente reconhecidas, como Airbnb, Twitter e GitHub.

**JAVASCRIPT**: pode ser usada tanto no front quanto no back-end, sendo umas das linguagens mais versáteis que existem. É a linguagem majoritária para desenvolvimento web e dificilmente um programador não terá contato com ela alguma vez na vida no mercado de trabalho.

**PHP**: é a linguagem mais famosa para criar backends de websites. Facebook e WordPress foram, em parte, escritas nela.

**Classificações das linguagens**





*--- PARADIGMAS*

**Paradigma Imperativo procedural**

* Primeiro paradigma de programação baseado no modelo clássico de von Neumann;
* Atribuições, sequências de comandos, laços de repetição e comandos condicionais fazem parte deste paradigma;
* Abstração procedural é sua principal característica.

Ex. C, Cobol, Fortran, Pascal, Ada, etc.

**Paradigma Imperativo**

* Orientado a Objetos;
* Um programa é constituído de vários objetos que trocam mensagens entre si;
* Objetos de dados são ativos e não passivos como no paradigma imperativo;
* Definição de classes de objetos, herança e passagem de mensagens caracterizam este paradigma.

Ex. C++, C#, Java, Smalltalk, etc.

**Paradigma declarativo funcional**

* O problema é modelado por um conjunto de funções matemáticas, cada uma com um espaço de entrada e um resultado, usada tradicionalmente em IA.
* As funções interagem entre si, utilizando a composição funcional.

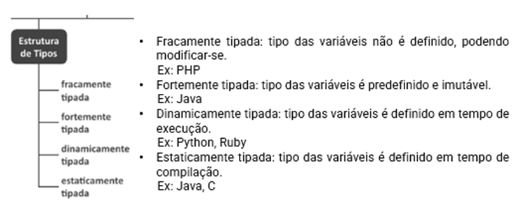
Ex. LISP, Haskell, ML, etc.

**Paradigma declarativo lógico**

* Declarativo – descreve o problema e o que se deseja fazer e não como fazer (programação imperativa e OO).
* Conhecida como baseada em regras.

Ex. Prolog.

*--- ESTRUTURA DE TIPOS*



*--- GRAU DE ABSTRAÇÃO*

* O grau de abstração funciona como uma escala para linguagens: quanto mais abaixo mais próximo da linguagem de máquina, e quanto mais alto, mais próximo da linguagem dos seres humanos.
* **Baixo nível**: possui símbolos que representam o código de máquina propriamente.

Ex: Assembly

* **Médio nível**: possui símbolos que podem ser diretamente traduzíveis para código de máquina, mas também possui símbolos que precisam ser processados por um compilador.

Ex: C#

* **Alto nível**: possui símbolos complexos que necessitam de interpretação de um compilador antes de serem transformados em linguagem de máquina.

Ex: Java, JavaScript, Python, Ruby