

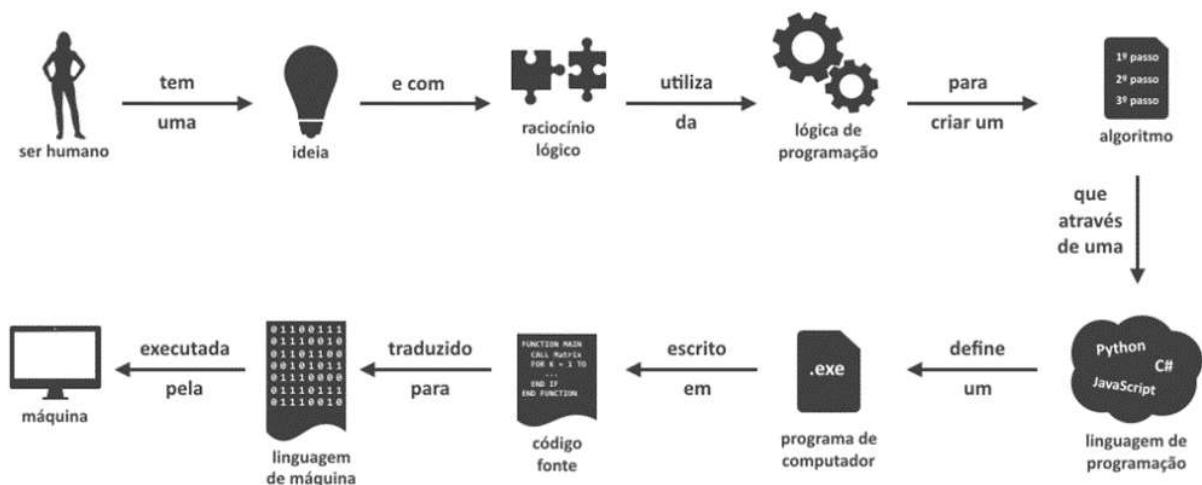


# Linguagens de Programação

## 1 - 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. Daí surge a necessidade de linguagens de programação.

Observe na figura abaixo:



Existem muitos domínios de aplicação para linguagens de programação:

- Inteligência Artificial;
- Educação (Ensino de Programação);
- Ciência e Engenharia;

- Sistemas de Informação;
- Sistemas e Redes;
- World Wide Web.



Dependendo da aplicação, uma ou outra linguagem torna-se mais adequada em função da tecnologia, investimentos, treinamentos, época, etc.

## 2 - Algumas linguagens de programação


### BASIC

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.

```
10 PRINT "Hello, World!"  
20 END
```

### 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.



```
IDENTIFICATION DIVISION.  
PROGRAM-ID.  IDSAMPLE.  
ENVIRONMENT DIVISION.  
PROCEDURE DIVISION.  
    DISPLAY 'HELLO WORLD'.  
    STOP RUN.
```

## PYTHON

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.

```
print("Hello World")
```

## 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.

```
global _main
extern _printf

section .text
_main:
    push    message
    call    _printf
    add     esp, 4
    ret
message:
    db     'Hello, World', 10, 0
```

## 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.

```
#include <stdio.h>
```



```
int main(void)
```

```
{
```

```
    printf("hello, world\n");
```

```
}
```

Exemplo de um programa em C:

```
#include <stdio.h>
```



```
int main()
{
    int x,y;

    printf("Digite um número inteiro:");
    scanf("%d",&x);
    printf("Digite outro número inteiro:");
    scanf("%d",&y);
    if (x > y) {
        printf("O maior valor é %d.\n",x);
    }
    else {
        printf("O maior valor é %d.\n",y);
    }
    return 0;
}
```

## JAVA

Java foi desenvolvida pela SUN Microsystems no início de 1990.

É “orientada a objetos”, portátil 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.

```
class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!"); // Prints the string to the file.
    }
}
```

```
import java.util.Scanner;
```

```
public class HelloWorld{

    public static void main(String []args){

        int x,y;
        Scanner scanner = new Scanner(System.in);

        System.out.println("Digite um numero inteiro:");
        x = scanner.nextInt();
        System.out.println("Digite outro numero inteiro:");
        y = scanner.nextInt();
        if (x > y) {
            System.out.print("O maior valor eh " + x + "\n");
        }
        else {
            System.out.print("O maior valor eh " + y + "\n");
        }
    }
}
```

## 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.

```
puts "Hello World"
```

## 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 meio de trabalho.

```
console.log("Hello World!");
```

## PHP

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

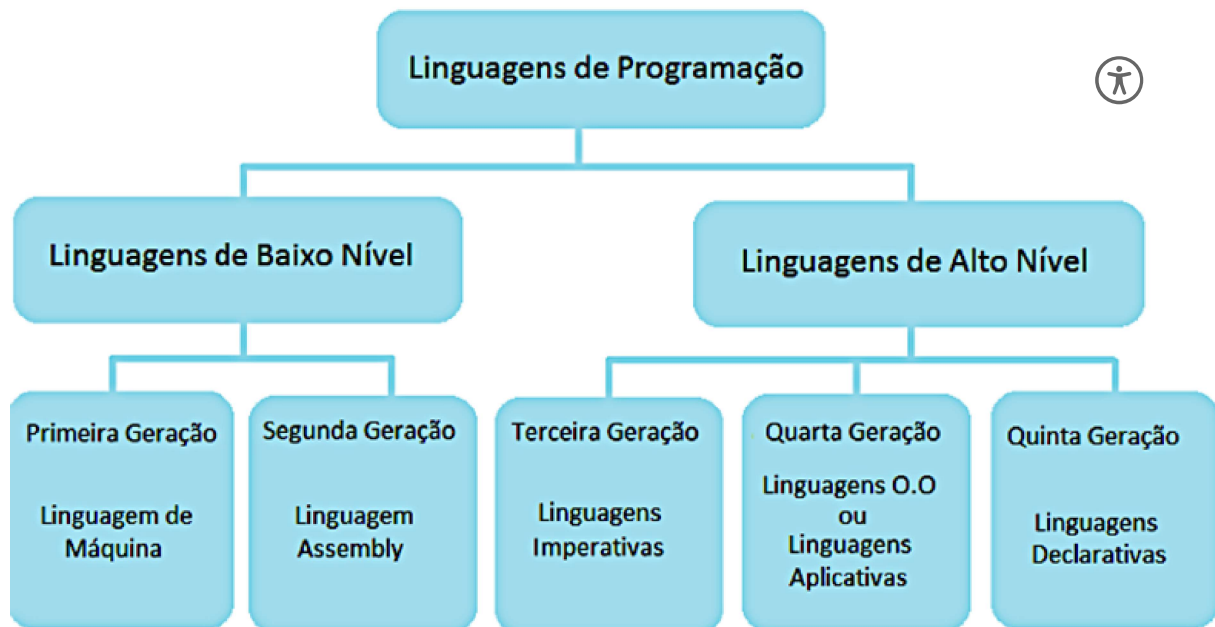
```
<?php echo "Hello, World";
```

## SWIFT

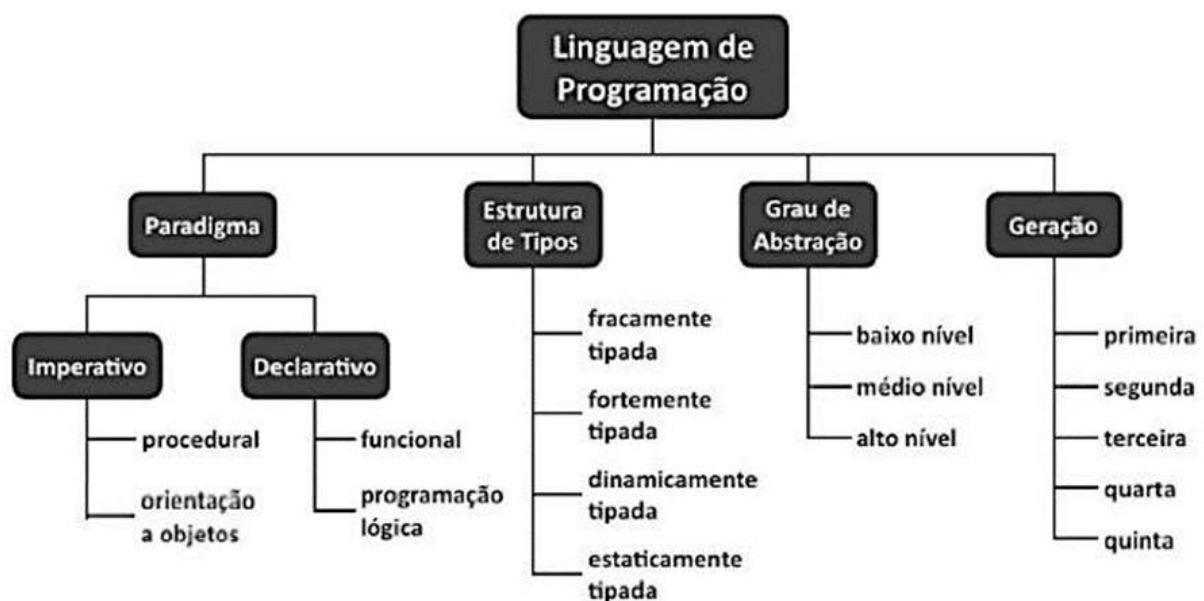
## II – Classificação das linguagens

### Alto nível e baixo nível





### Classificação das linguagens de programação:



### 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



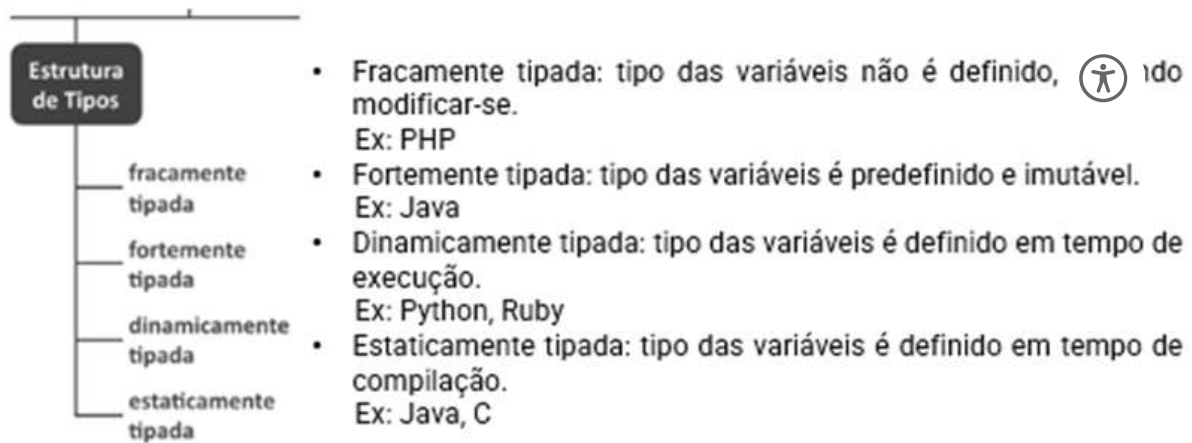
deseja fazer e não como fazer (programação

imperativa e OO).

- Conhecida como baseada em regras.
- Ex. Prolog

```
fac(0,1) :- !.  
fac(I,O) :- I1 is I - 1,  
            fac(I1,O1), O is I * O1.  
?- fac(1,X).  
X = 1  
?- fac(3,X).  
X = 6  
?- fac(5,X).  
X = 120
```

**Quanto à estrutura de tipos**



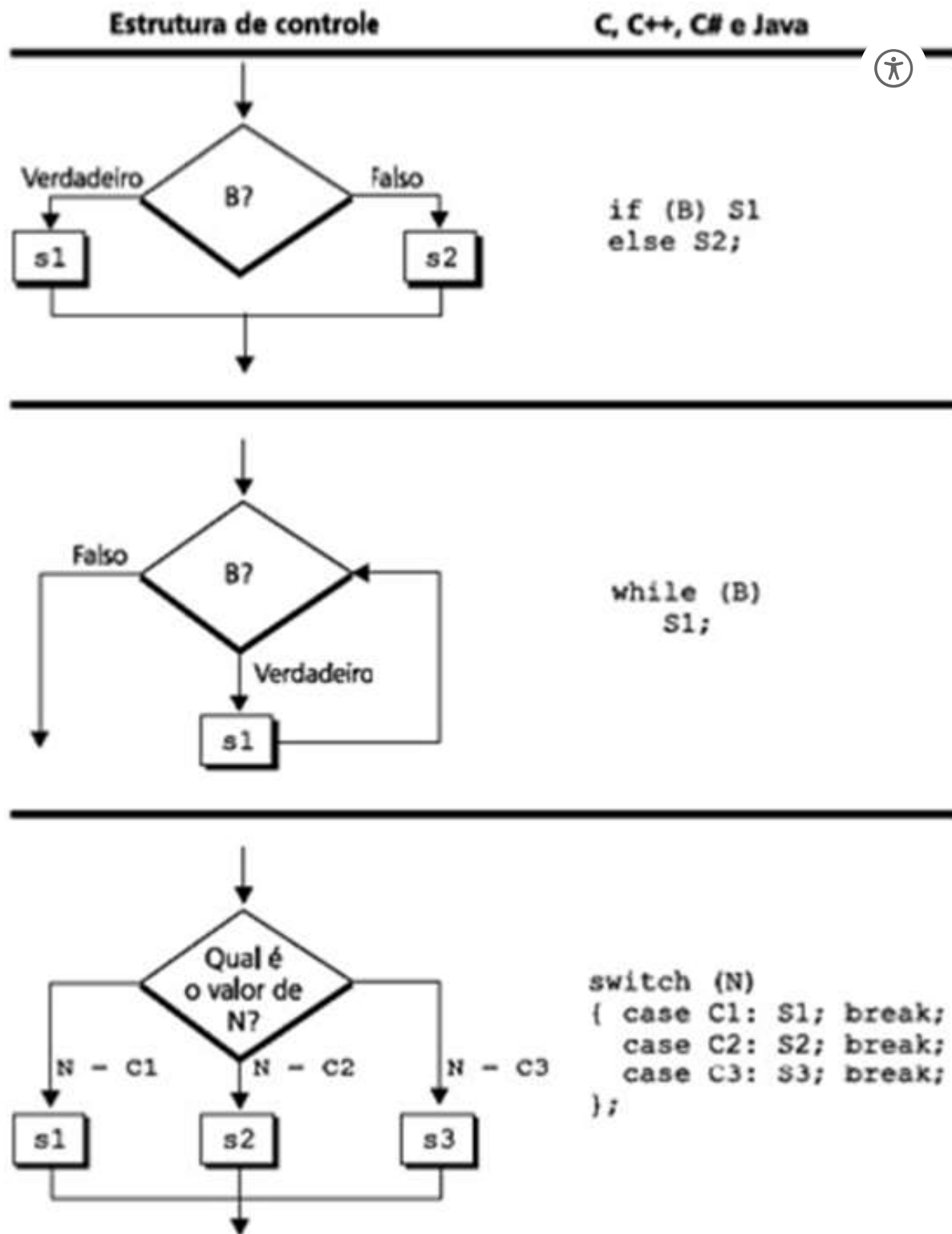
## Quanto ao grau de abstração

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.

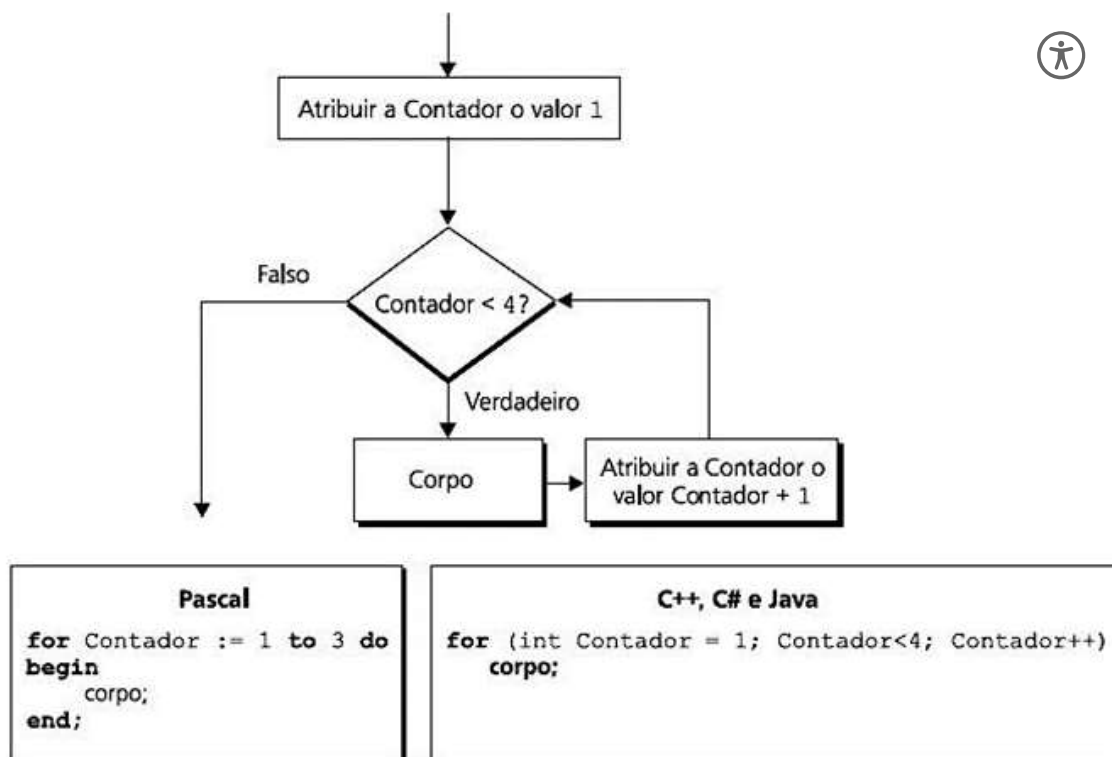
- Baixo nível: possui símbolos que representam o código de máquina propriamente. Ex: Assembly
- Médio nível: possui símbolos que podem ser diretamente traduzíveis para código de máquina, mas também possui símbolos que precisam ser processados por um compilador. Ex: C#
- Alto nível: possui símbolos complexos que necessitam de interpretação de um compilador antes de serem transformados em linguagem de máquina. Ex: Java, JavaScript, Python, Ruby

Algumas comparações

**A mesma estrutura de controle nas linguagens C, C++, C# e Java: idênticas!**



## Outras Comparações



## Declarações de variáveis em diferentes linguagens

---

### a. Declarações de variáveis em Pascal



**var**

```
Comprimento, Largura:    real;
Valor, Imposto, Total: integer;
Simbolo:                  char;
```

---

### b. Declarações de variáveis em linguagem C, C++, C# e Java

```
float Comprimento, Largura;
int Valor, Imposto, Total;
char Simbolo;
```

---

### c. Declarações de variáveis em linguagem FORTRAN

```
REAL Comprimento, Largura
INTEGER Valor, Imposto, Total
CHARACTER Simbolo
```

---

## Dicas

### Qual linguagem de programação aprender?

Escolha qual será sua atuação: *front-end*, *back-end*, *full stack*... e procure conhecer as linguagens de programação mais utilizadas dessa área.

Não é ideal se prender a uma linguagem ou tecnologia específica, principalmente no começo da carreira.

Uma mudança de empresa ou projeto pode exigir que você lide com outras linguagens e o mercado muda a linguagem “da moda” de tempos em tempos.

## Front-end



Quem trabalha com Front End é responsável por desenvolver por meio de código uma interface gráfica, normalmente com as tecnologias base na Web (HTML, CSS e JavaScript...).

Algumas pessoas podem confundir um pouco esse trabalho com o que um designer faz mas a diferença é que o designer vai utilizar alguma ferramenta visual para desenhar a interface, do Photoshop ao Sketch, e quem faz front-end estará mais próxima do código em si, que irá rodar em um navegador Web como Chrome, Firefox ou Safari.

## Back-end

O Back End trabalha em boa parte dos casos fazendo a ponte entre os dados que vem do navegador rumo ao banco de dados e vice-versa em um ambiente em que o usuário final use a interface mas de certa forma ignore o que está por trás dela. Normalmente as linguagens são Python, Java, C#, PHP e muitas outras.

## Full Stack

Full stack developer é quem trabalha com Front End e Back End.

Com o passar dos anos é meio natural, após começar por um dos lados você vai aprendendo como o outro funciona.

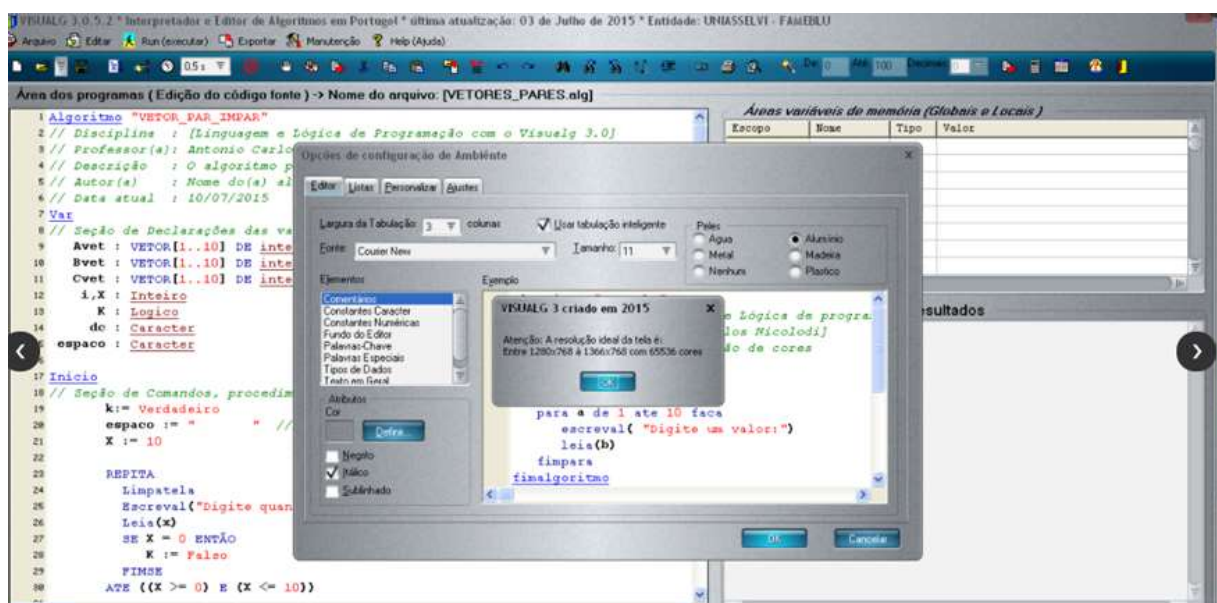


## Linguagem “quase de programação” VisuAlg



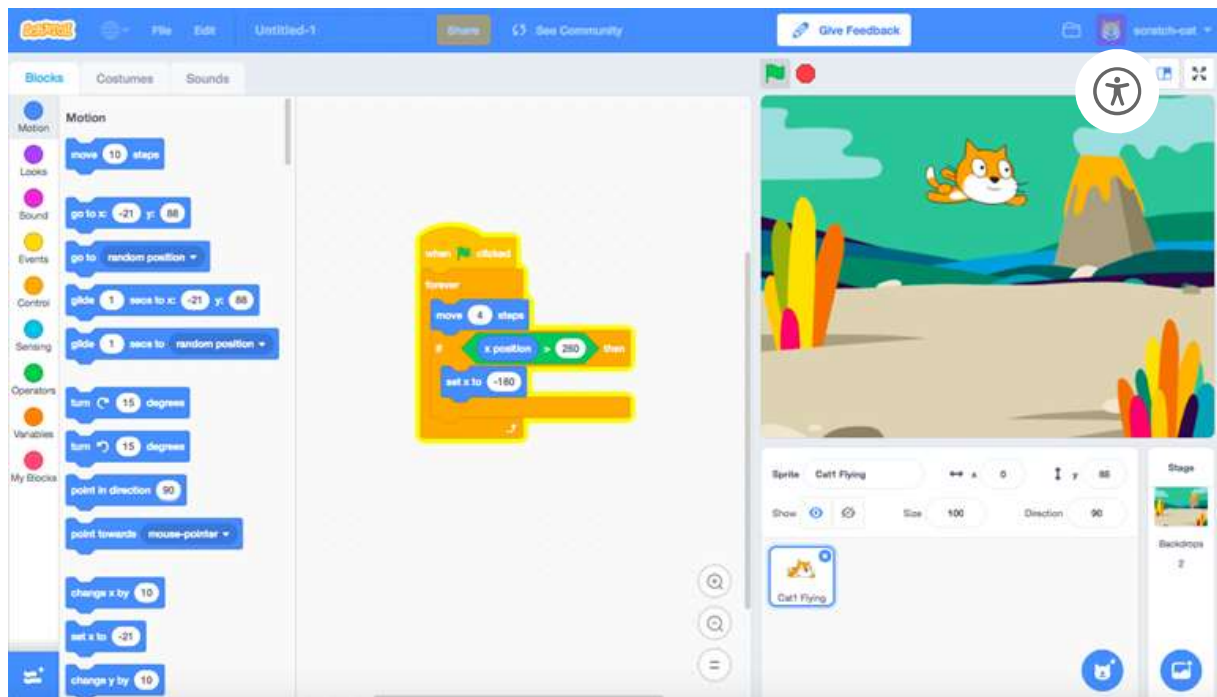
O VisuAlg é um programa que permite criar, editar, interpretar e que também executa os algoritmos em português (estruturado português) como se fosse um “programa” normal de computador.

É um programa de livre uso e distribuição, GRÁTIS e DOMÍNIO PÚBLICO, usado para o ensino de lógica de programação em várias escolas e universidades no Brasil e no exterior.



## Linguagem SCRATCH

Esta linguagem foi desenvolvida pelo MIT com o propósito de iniciar as pessoas em programação. Tem sido usada em algumas instituições como primeira linguagem e facilita muito o desenvolvimento de habilidades de lógica de programação.



## Referência Bibliográfica

BROOKSHEAR, J.G. **Ciência da Computação: uma visão abrangente**. Porto Alegre: Bookman, 2013.

FORBELLONE, A.L.V. & EBERSPACHER, H. F. **Lógica de Programação – A Construção de Algoritmos e Estruturas de Dados**. 3ª. Edição. São Paulo, SP: Prentice Hall, 2005.

SEBESTA, R. W. **Conceitos de linguagens de programação** [recurso eletrônico] /

Robert W. Sebesta; **tradução técnica**: Eduardo Kessler Piveta. –

9. ed. – **Dados eletrônicos**. – Porto Alegre : Bookman, 2011.

**Ir para exercício**