

## Aula 1

### Docupedia Export

Author:Goncalves Donathan (CtP/ETS)  
Date:21-Aug-2023 19:16

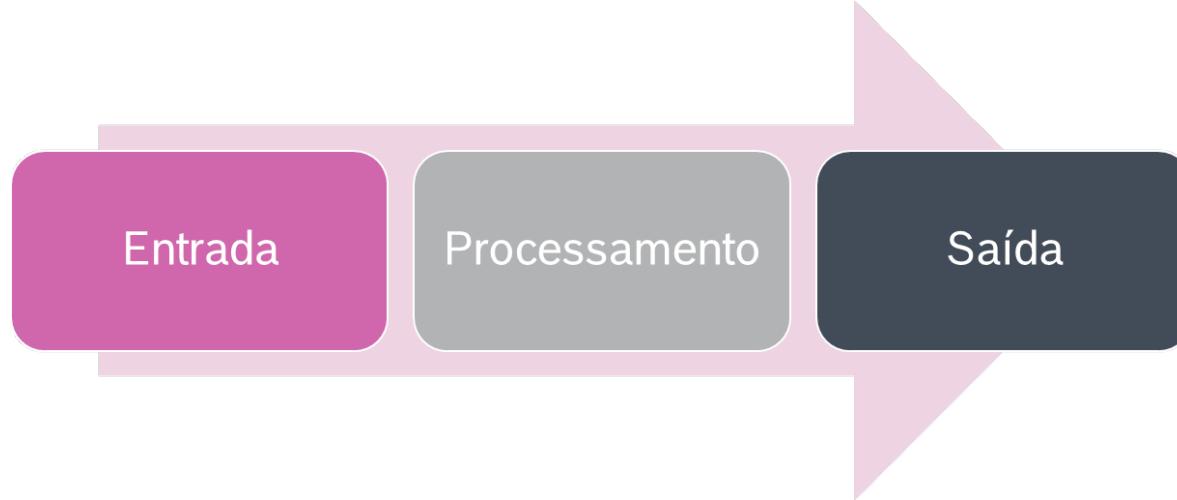
## Table of Contents

<b>1 Funcionamento básico de um computador</b>	<b>4</b>
1.1 Dispositivos de entrada	4
1.2 Processamento	5
1.3 Processamento – Memória	7
1.4 Dispositivos de saída	8
1.5 Visão Geral – Arquitetura Von Neumann	9
<b>2 Programas, Linguagens de Programação e Sistemas Operacionais</b>	<b>10</b>
2.1 Conceitos – Algoritmos e Programas	10
2.2 Conceitos - Programas	10
2.3 Níveis	12
2.4 Gerações	14
2.5 Paradigma	15
<b>3 Sistemas Operacionais</b>	<b>16</b>
<b>4 Programação Básica</b>	<b>17</b>
4.1 Lógica de Programação	17
4.2 Loops	17
<b>5 Conectivos Lógicos</b>	<b>19</b>
5.1 Conectivo E (AND)	19
5.2 Conectivo OU (OR):	20
5.3 Para o conectivo E:	20
5.4 Para o conectivo OU:	20
5.5 Conectivo NÃO (NOT):	21
<b>6 Estrutura</b>	<b>22</b>
6.1 Pseudocódigo	22

6.2 Debug	22
6.3 Exercício 1 – Pseudograma Idade para Votar	23
6.4 Correção	24
6.5 Exercício 2 – Pseudograma Lanchonete	25
6.6 Correção	26
<b>7 Constantes e Variáveis</b>	<b>27</b>
<b>8 Projeto aplicado com Scratch</b>	<b>29</b>
8.1 Exercício 3	34
8.2 Correção	35

# 1 Funcionamento básico de um computador

De maneira simplificada, um computador recebe informações de **entrada**, realiza algum **cálculo** ou manipulação e retorna uma **saída** (que pode ou não ser mostrada para o usuário).



## 1.1 Dispositivos de entrada

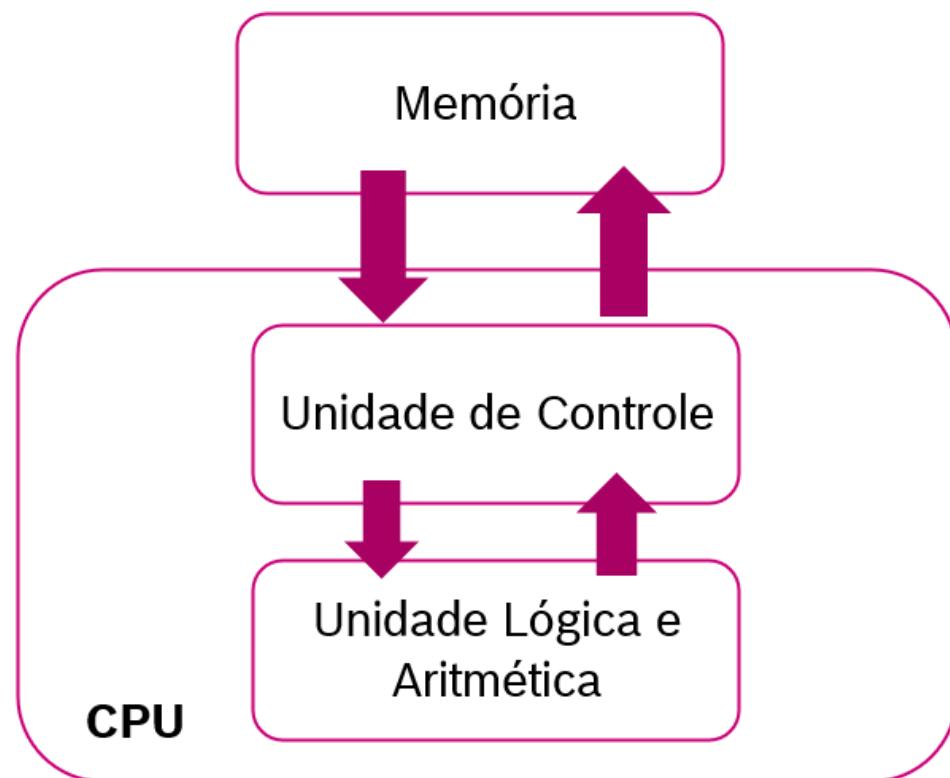
Permitem a interação entre o usuário e o computador. Recebe comandos e informações.

Exemplos : mouse, teclado, microfone, câmera, pen drive, entre outros.



## 1.2 Processamento

A Unidade Central de Processamento (CPU) pode ser subdividida em: Memória, Unidade de Controle (UC) e Unidade Lógica e Aritmética (ULA).

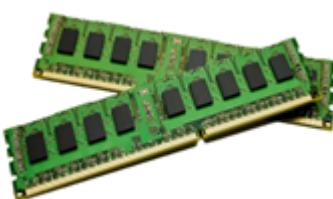




## 1.3 Processamento – Memória

Armazena os dados, os programas que manipulam os dados, e os resultados obtidos. Podem ser divididas em dois tipos:

- **Memória Principal:** É uma memória volátil, isto é, precisa de energia para manter seu conteúdo (só funciona com computador ligado). As informações são armazenadas temporariamente. Pode ser considerada a memória de trabalho do computador. Ex: Memória RAM;



- **Memória Secundária:** É uma memória não volátil. A informação gravada não é perdida quando o computador é desligado, ficando armazenada



permanentemente. Ex: HD (*Hard Drive* ou Disco Rígido).

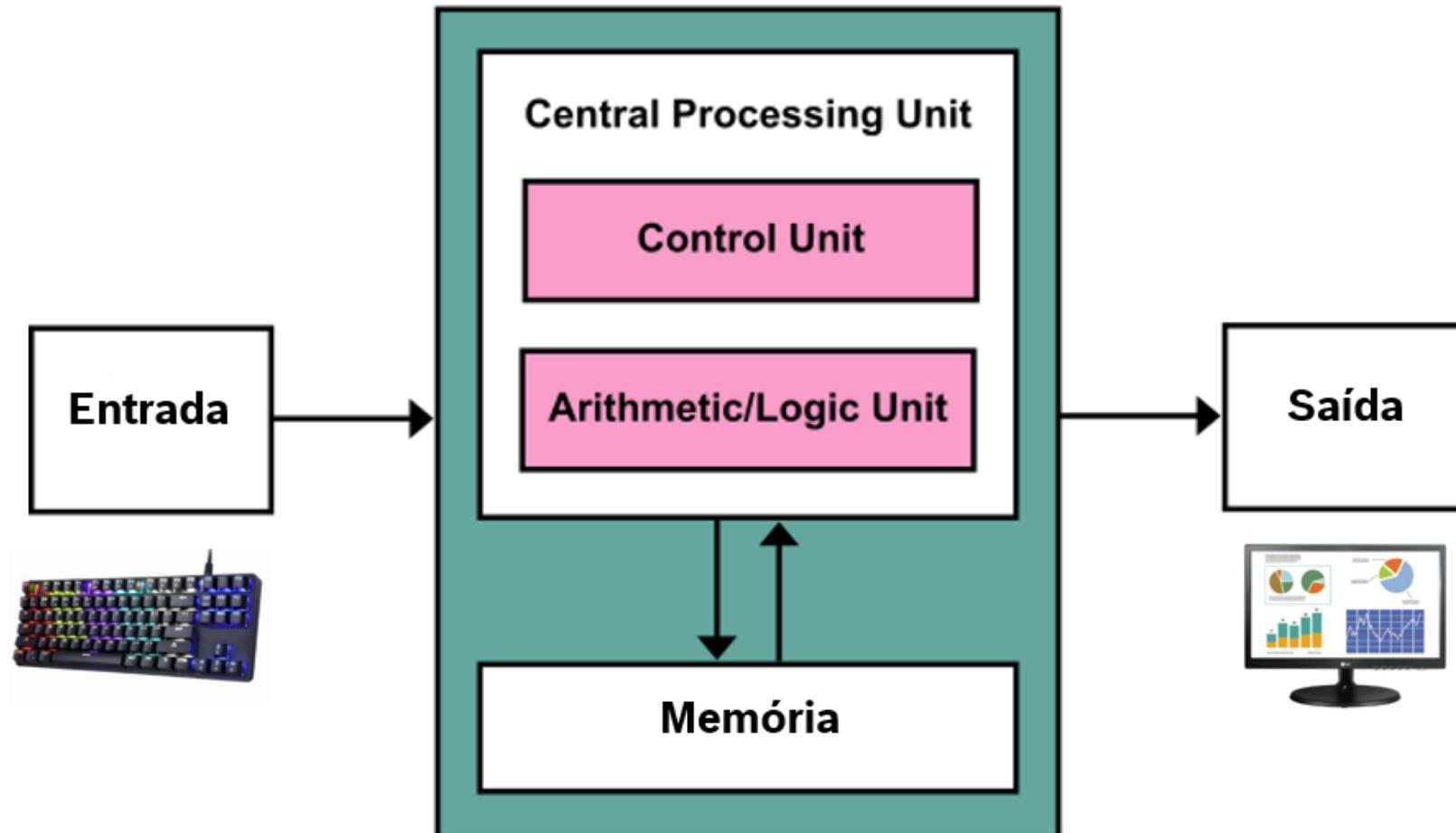
- **Unidade de Controle (UC):** é responsável pelo fluxo de dados. Todo processamento é controlado pela UC. Recebe as informações contidas na memória, decodifica e envia para que a Unidade Lógica e Aritmética possa executar as operações;
- **Unidade Lógica e Aritmética (ULA):** Realizados todos os cálculos e manipulação de dados, ou seja, realiza operações lógicas (E, OU,...) e aritméticas (soma, subtração,...).

## 1.4 Dispositivos de saída

- Retorna o resultado dos cálculos (manipulações), realizados anteriormente. Exemplos são monitor, caixa de som, impressora, entre outros.



## 1.5 Visão Geral – Arquitetura Von Neumann



## 2 Programas, Linguagens de Programação e Sistemas Operacionais

### 2.1 Conceitos – Algoritmos e Programas



Imagine uma cozinha, com diversos ingredientes, utensílios de cozinha, um forno e um padeiro.

O padeiro segue uma receita para obter um bolo, a partir dos ingredientes e com o auxílio do forno e dos utensílios. Cada receita produz um tipo de bolo diferente, ou seja, mudando a receita, muda-se o bolo.

No computador-cozinha:

- os **ingredientes** constituem a **entrada**;
- o **bolo** é a **saída** desejada;
- enquanto a **receita** constitui um **algoritmo**.

Este conjunto forma o **software**.

Já os utensílios, o forno e o próprio padeiro constituem as ferramentas necessárias no processo: o **hardware**.

**Instruções:** Indica uma ação elementar a ser executada, um passo a ser feito;

Ex: “Adicione 1 xícara de farinha.”

**Algoritmos:** Conjunto de instruções bem definidas que são executadas para realizar uma tarefa ou resolver um problema.

Receita de bolo de chocolate;

Instruções para trocar resistência de um chuveiro;

Instruções para trocar uma lâmpada

### 2.2 Conceitos - Programas

Programa de computador ou *software* são **algoritmos** a serem usados por um computador, a fim de obter um determinado resultado. É composto por um código-fonte, desenvolvido em alguma **linguagem de programação**.



Um computador sabe apenas o que dizemos à ele. Para processar qualquer dado, devemos informar quais dados são esses e de que forma devem ser manipulados; A linguagem entendida pelo computador é chamada de **linguagem de máquina**. É de difícil compreensão para o ser humano, pois é constituída apenas por 0 e 1, que são os estados desligado ou ligado de uma chave eletrônica.

```
?PROBLEMS? Enter moves in Std Notation
Examples: P-Q4,N-K5,B-QN5,Q-NQ3,QN-K2,0-0
0-0-0,NXP,QXKP,PXQBP,N/3XP,BXP/5,P-BB=Q
You can also use algebraic, i.e. E2-E4, G1-F3, etc
To Run :RU,CHESS,(white lu),(black lu),(list lu)

**EXAMPLES OF OTHER FEATURES**
/R Reject last move /Rn Restore board back n moves
/Sn or L0n Save or Load game to/From LU n
/LUn Set level of play to n (1-10) Time(E-series) for
each level~= .2S,2S,15S,75S,8M,1HR,6HR,1.5DV,9DV,54DV
/SW Switch colors with opponent. BW Same, but new color's move
/SVCC Let Computer play your color too
/E End CHESS - /B Begin a new game
/Q Turnoff display
/D (264x) Display board -(/D1 invert display)
/D2 (2621) Redisplay after each move (/D3 inverted)
/CW# (/CB$) Overscore white(black) with "#"("$")
#I Clear board
#TC3D4 Remove pieces on algebraic squares C3 & D4
#WRG5 Put a white rook on square G5
#BPF8B2 Put black pawns on F8 & B2
/FMn Find mate in n(1-5) moves
HIT A KEY TO CONTINUE - Questions? See Chuck Whelan
```

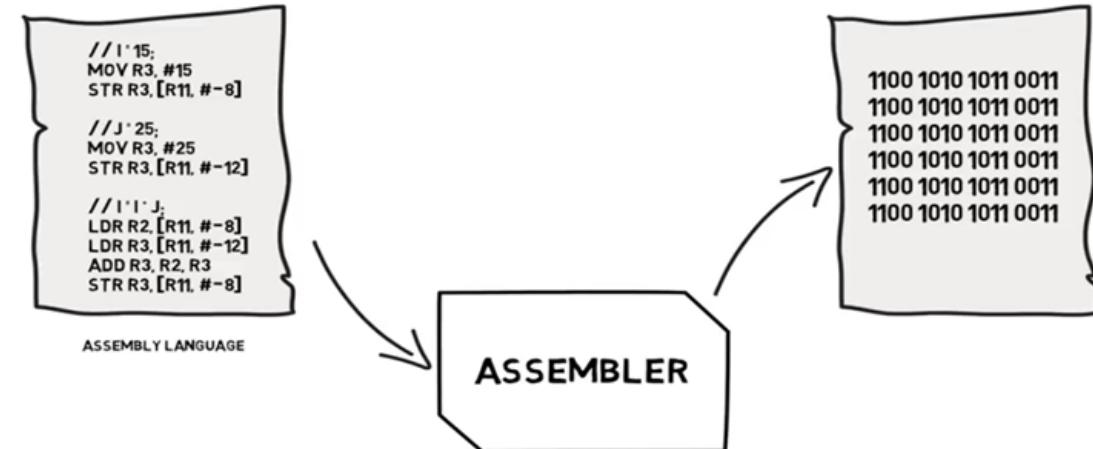
```
00001110 01111101 00111111 00100111 11000000 11110000 00011011 100000011
00011100 100111000 00000001 00000111 11101110 11111000 00111100 111110100
11111101 00110110 01100011 00100011 00011100 00000111 00110000 111100100
00010101 11100111 11011000 01110010 00111110 01100011 11010110 000110010
11100110 11110111 00100111 11000000 11101110 00000011 10010000 111101111
10001110 10011011 10000001 00000110 10110011 11001100 11110001 110000000
11001101 00000000 10011100 00000000 11000001 11000010 10001100 011111110
01001110 11111011 10110011 00001111 10011011 10000111 11110001 000000110
01001100 10000100 00011001 10011001 11111111 00110011 11000000 110100100
00000111 11111111 00101001 00101011 10000011 01011111 11000011 010010100
00000010 10111111 11110000 11001110 11011111 11011000 11111111 110001100
11111110 00000101 11011110 10011111 11001001 01101101 10010001 100111111
00011011 01100100 11000011 00001111 00001000 00001100 00001100 000111100
00111100 11100011 00011001 00010010 10011011 10000011 10111001 110001011
01000011 00000111 11111010 00000111 11100001 11001000 10000011 000001101
10011100 11111100 11000000 10110100 11001000 10011000 11001111 001001100
11101111 11011111 01001110 00110011 01111001 00101101 10011100 011111000
01000111 01101110 00001100 00001111 11000011 11001101 01101111 011111100
01001011 10000111 00011100 00011111 00011101 00011110 01011010 110000011
```

A **linguagem de programação** (LP) é usada para expressar instruções para o computador de forma padronizada, ou seja, **conjunto de regras semânticas e sintáticas** para definir um programa de computador. Desta forma, é possível passar instruções para um computador sem que tudo seja escrito com 0 e 1. LPs podem ser classificadas em relação à seu nível, geração ou paradigma.

## 2.3 Níveis

### Baixo nível

- As instruções são mais próximas à linguagem de máquina (0 e 1);
- Em geral, possuem ótimo desempenho computacional;
- Não visam a facilidade de uso pelo programador (exigem maior esforço);
- São conhecidas como “Assembly”;
- Código gerado para um processador não pode ser usado por outro



### Alto nível



- São as mais utilizadas;
- Maior facilidade de entendimento e uso pelo programador (mais voltadas ao ser humano);
- Necessitam tradução para linguagem de máquina;

- Podem ser executadas em várias plataformas com poucas modificações;

### LINGUAGEM DE ALTO NÍVEL

Exemplos: FORTRAN BASIC PASCAL COBOL  
Exemplos de comandos: PRINT SAVE DO

### LINGUAGEM DE MONTAGEM

Exemplos: ASSEMBLY Z28 ASSEMBLY 8086  
Exemplos de comandos: MPY JMP ADD

Maior necessida de Tradução (Computador)

Maior esforço de Programação (Homem)

### CÓDIGOS OPERACIONAIS

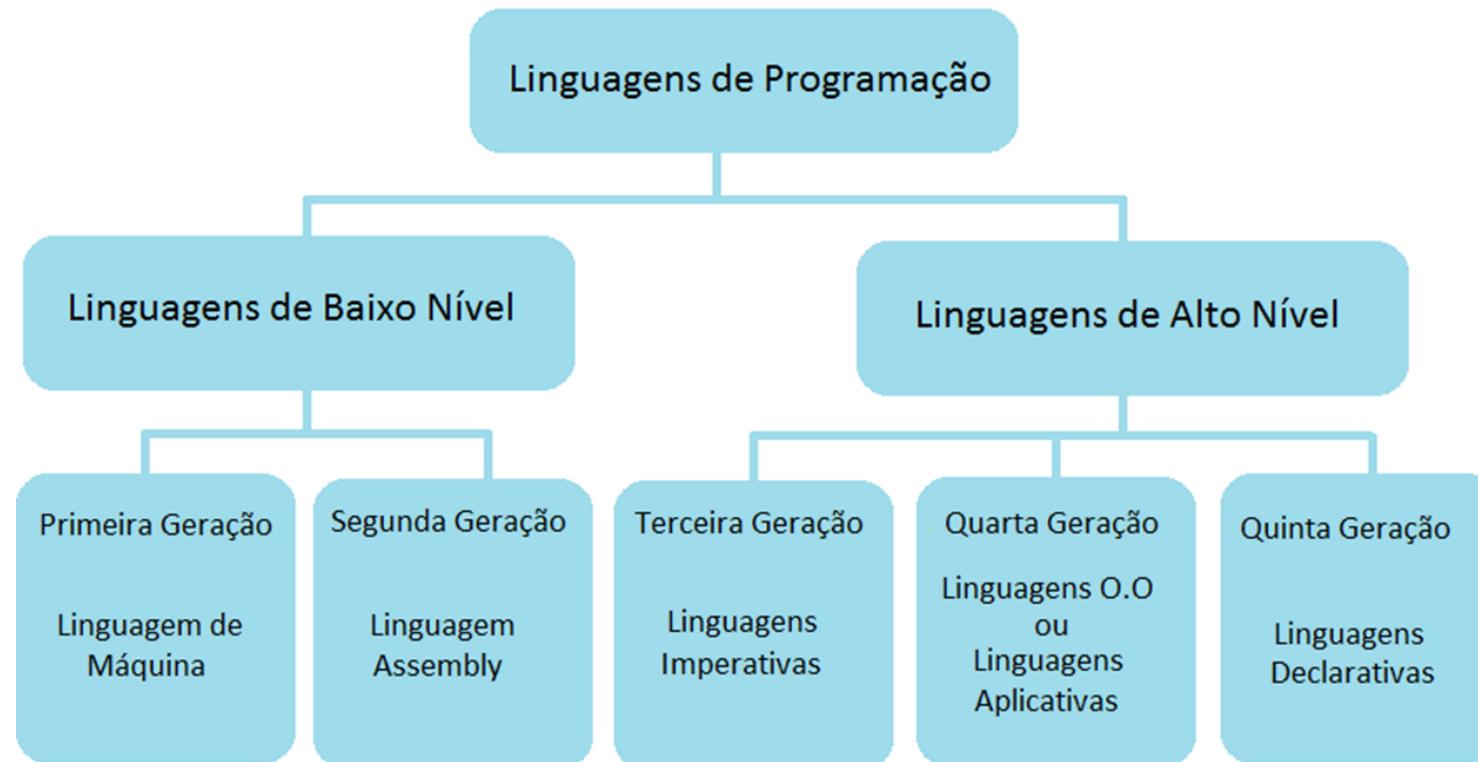
Exemplos de códigos: E60F CD8001 DB3F

## LINGUAGEM BINARIA (LINGUAGEM DE MAQUINA)

Exemplos de códigos: 01000001 00100100 01100001  
A            \$            a

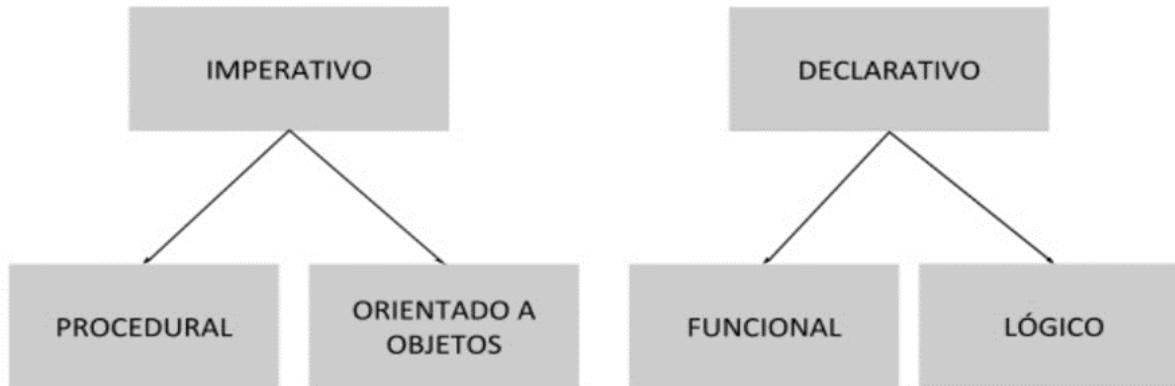
Quanto mais baixo o nível, mais difícil se torna a programação.

## 2.4 Gerações



## 2.5 Paradigma

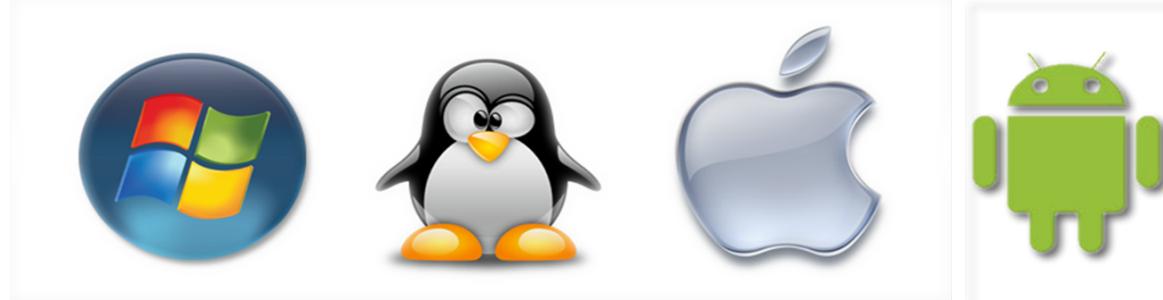
Paradigma é um ponto de vista. Cada paradigma traz novas formas de se pensar programação e soluções de softwares; Com base nos paradigmas, as LPs podem ser classificadas em imperativa (procedural e orientada à objetos) e declarativa (funcional e lógica).



- **Procedural:** Descreve todas as operações a serem realizadas. Cada sentença deve ser executada em ordem e somente juntas fazem sentido. Orientação à ações.
- **Orientado à objetos:** Elementos e conceitos associados ao problema são tratados como objetos. Objetos são entidades abstratas que contêm as características e operações relacionadas com a entidade real.
- **Funcional:** Enfatiza aplicação de funções ao invés de mudanças no estado do programa.
- **Lógico:** Descreve o problema formalmente, similar ao raciocínio do ser humano. Declara fatos, que podem ser relações ou regras que produzem fatos. Mais descriptiva.

## 3 Sistemas Operacionais

Podemos definir um sistema operacional como: “o conjunto de programas que gerenciam recursos, processadores, armazenamento, dispositivos de entrada e saída e dados da máquina e seus periféricos. O sistema que faz comunicação entre o hardware e os demais softwares. O Sistema Operacional cria uma plataforma comum a todos os programas utilizados. Exemplos: DOS, Unix, Linux, Mac OS, Windows, Android...” .



Dentre as funções básicas do sistema operacional, podemos citar:

- Definição da interface com o usuário;
- Compartilhamento de hardware entre usuários;
- Compartilhamento de dados entre usuários;
- Gerenciamento dos dispositivos de entrada e saída;
- Tratamento e recuperação de erros.

## 4 Programação Básica

### 4.1 Lógica de Programação

Lógica de programação nada mais é do que a técnica de encadear pensamentos para atingir um determinado resultado. É como pensamos em instruções para construir um algoritmo (um manual, uma receita...).

Pode ser feito de várias maneiras diferentes;

Quais são os passos para escovar os dentes?

Opção 1:

- Pegar a escova;
- Abrir a torneira;
- Molhar a escova;
- Colocar pasta de dente na escova;
- Escovar os dentes.

Opção 2:

- Pegar a escova;
- Abrir a torneira;
- Colocar pasta de dente na escova;
- Molhar a escova;
- Fechar a torneira;
- Escovar os dentes.

### 4.2 Loops

Muitas vezes é necessário escolher qual ação realizar ou repetir uma determinada instrução diversas vezes até que determinado evento aconteça (**loop**)

Algoritmo: “Descascar batatas”

- Pegar uma faca
- Pegar batata
- Descascar batata

Utilizamos laços de repetição ou **loops**  
Algoritmo “Descascar várias batatas”

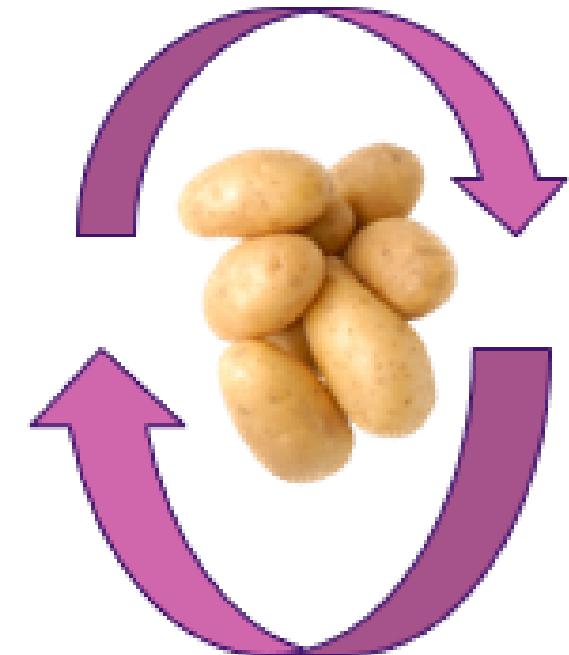
- Pegar uma faca
- **Enquanto** houver batatas:
  - Pegar batatas
  - Descascar batatas

**E se** existirem batatas podres no meio?

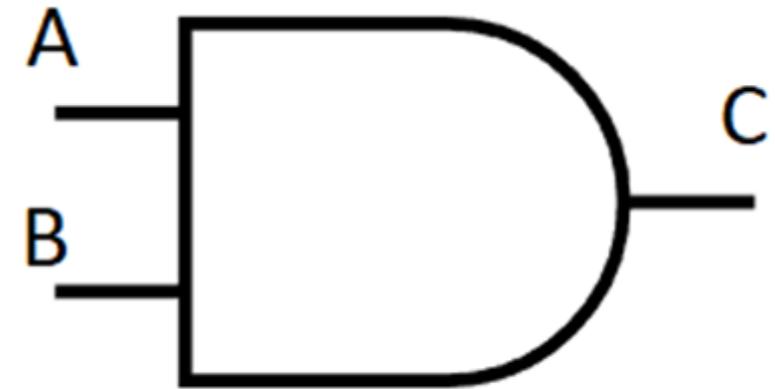
Além da repetição utiliza uma condição.

Algoritmo “Descascar várias batatas”

- Pegar uma faca
- Enquanto houver batatas
  - Pegar batatas
  - **Se** a batata **NÃO** estiver podre:
    - Descascar batatas



## 5 Conectivos Lógicos



### 5.1 Conectivo E (AND)

**Todos** os elementos devem acontecer para que seu valor seja verdadeiro

Para “Descascar batatas”:

- Tem faca? **E**
- Tem batatas? **E**

Caso não haja algum desses elementos, não é possível realizar a tarefa “Descascar batatas”. **TODOS** precisam acontecer!

## 5.2

### Conejto OU (OR):



Pelo menos um elemento devem acontecer para que seu valor seja verdadeiro

Para “fazer compra”:

- Tem dinheiro? **OU**
- Tem cartão de crédito? **OU**
- Tem cheque? **OU**

Se apenas um acontecer, já podemos executar a tarefa “fazer compra”. **Qualquer um** satisfaz.

## 5.3 Para o conectivo E:

- **Todas** as condições devem ser verdadeiras para que o resultado seja **verdadeiro**.

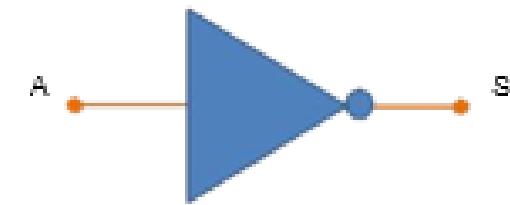
## 5.4 Para o conectivo OU:

- **Qualquer um** deve ser **verdadeiro** para que o resultado seja verdadeiro;
- Só será falso quando **TODAS** as condições forem falsas!

Condição A	Condição B	E (AND)	OU (OR)
Verdadeiro	Verdadeiro	Verdadeiro	Verdadeiro

Verdadeiro	Falso	Falso	Verdadeiro
Falso	Verdadeiro	Falso	Verdadeiro
Falso	Falso	Falso	Falso

## 5.5 Conectivo NÃO (NOT):



- Inverte o estado (de verdadeiro para falso e vice-versa)
- Maria não é casada **NÃO** Maria é casada;
- Pedro mora em Campinas **NÃO** Pedro não mora em Campinas.

## 6 Estrutura

De forma análoga ao funcionamento geral de um computador, um algoritmo pode ser subdividido em 3 fases:

1. **Entrada:** possíveis dados de entrada do algoritmo;
2. **Processamento:** realiza as manipulações necessárias para transformar os dados de entrada em saída;
3. **Saída:** Resultados das etapas anteriores.

### 6.1 Pseudocódigo

Intermediário entre a língua falada e a linguagem de programação na qual o algoritmo será implementado;

Deve ser fácil de interpretar e de codificar;

1. **Entrada:** informar as três notas: p1, p2 e p3;
2. **Processamento:** Calcular a média (somar as 3 notas e dividir por 3);
3. **Saída:** apresentar o resultado do cálculo realizado na fase anterior, ou seja, a média obtida das três notas.

Algoritmo “Média das notas”

```
INÍCIO;
    Leia(P1);
    Leia(P2);
    Leia(P3);
    media = (P1 + P2 + P3) / 3;
    Escreva("media");
FIM;
```

### 6.2 Debug

O debug(teste de mesa) é usado para testar o algoritmo e ter certeza de que está funcionando corretamente;

- Deve-se seguir as instruções de maneira precisa;
- Serve para verificar se o resultado é o esperado.

**Exemplo:**

Notas de um aluno na disciplina de Geografia:

- Prova 1: 7.4
- Prova 2: 7.0
- Prova 3: 8.2

Utilizar o algoritmo “Média das notas”

Qual será a saída do algoritmo?

Leia(p1)	Leia(p2)	Leia(p3)	media	Escreva(media)
7.4	7.0	8.2	$(7.4 + 7.0 + 8.2) / 3$	7.53

## 6.3 Exercício 1 – Pseudograma Idade para Votar

Como verificar se uma pessoa pode votar com base na idade?

- Se a pessoa tiver mais que 18 anos ela pode votar;
- Pessoas com idade entre 16 e 18 anos têm voto facultativo;
- Menores de 16 não podem votar.

Tempo estimado: 10 minutos.

## 6.4 Correção

INÍCIO;

```
If(idade < 16);  
    Escreva("Não pode votar");  
Else if(idade < 18);  
    Escreva("Facultativo");  
Else;  
    Escreva("Obrigatório");
```

FIM.



```
If = Se  
Else if = Senão se  
Else = Senão
```

## 6.5 Exercício 2 – Pseudograma Lanchonete

**Cardápio**

	<b>Hambúrguer</b> R\$ 7,00
	<b>Refrigerante</b> 500ml   R\$ 5,00
	<b>Refrigerante</b> 1,5l   R\$ 7,00
	<b>Batata frita</b> R\$ 7,00
	<b>Combo</b> R\$ 18,00

Você e dois amigos saem para comer fast food e todos querem comer um hambúrguer com batata frita e tomar refrigerante. Vocês tem 3 opções:

- Cada um compra um hambúrguer, uma batata e um refrigerante 500ml;
- Cada um compra um combo;
- Cada um compra um hambúrguer e uma batata e os 3 dividem um refrigerante 1,5l.

• Faça uma lógica para calcular o preço de cada opção e escolher qual será a mais barata para cada um de vocês.

**Tempo estimado:** 10 minutos.

## 6.6 Correção

INÍCIO;

```
Hambúrguer (H1);  
Refrigerante 500 (R1);  
Refrigerante 1,5L (R2);  
Batata (B1);  
Combo (C1);  
Soma = (H1) + (B1) + (R1);  
    Escreva("Op1");  
Valor = (C1);  
    Escreva("Op2");  
Soma = (H1) + (B1) + ((R2/3);  
    Escreva("Op3");
```

```
If(Op1 < Op2);  
    Melhor = Op1;  
Else;  
    Melhor = Op2;  
If(Melhor > Op3);  
    Melhor = Op3;  
Escreva("Melhor");
```

FIM.

## 7 Constantes e Variáveis

**Constante:** Possui valor fixo, **NÃO** é alterado durante a execução;

- Constante numérica: pi = 3.14

**Variável:** Seu valor **pode ser alterado** durante a execução. Serve para armazenar dados do programa na memória principal, onde cada variável corresponde a uma posição de memória;

- nome1 = "Maria" idade = 13
- nome2 = "João" idade = 15
- nome3 = "Cecília" idade = 21

As variáveis **nome1**, **nome2** e **nome3** apontam para locais diferentes na memória. A variável **idade** aponta para o **MESMO** local na memória. Neste caso, qual é o valor da variável **idade**?

```
idade = 13  
idade = 15  
idade = 21  
print("A idade é", idade)  
  
A idade é 21
```

Para armazenar todas as idades precisamos de variáveis diferentes!

- idade1 = 13
- idade2 = 15
- idade3 = 21

Por que utilizar variáveis? São reutilizáveis!

```
idade1 = 13  
idade2 = 15  
idade3 = 21  
print("As idades são: ", idade1, idade2, idade3)
```

As idades são: 13 15 21

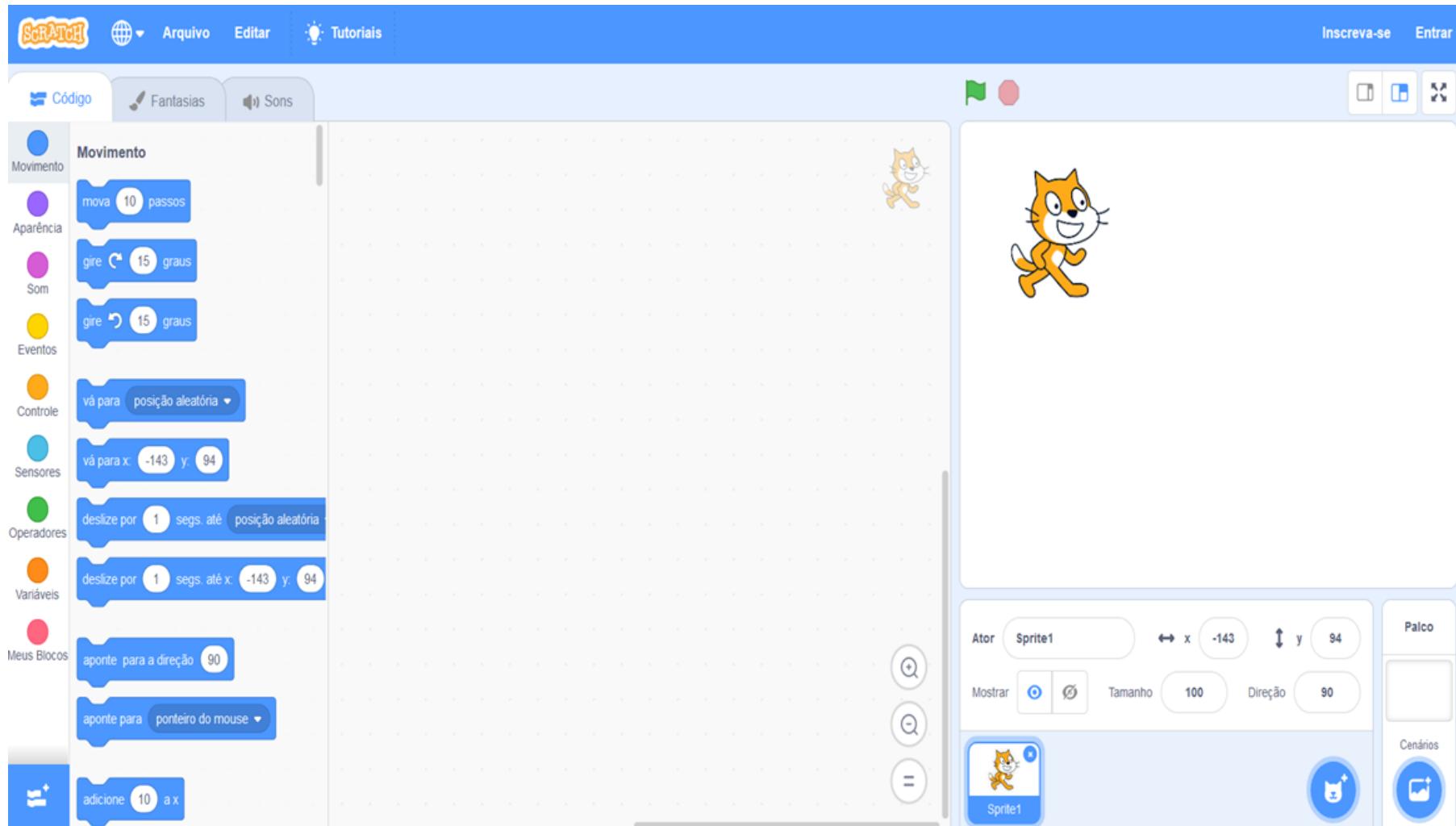
"Informe o valor do lado de um quadrado para que sua área seja calculada."

- Leia(lado)

- $\text{área} = \text{lado} * \text{lado}$
- Escreva( $\text{área}$ )

Várias áreas podem ser calculadas utilizando as mesmas variáveis.

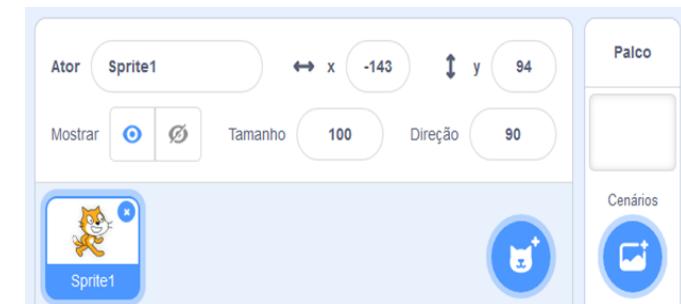
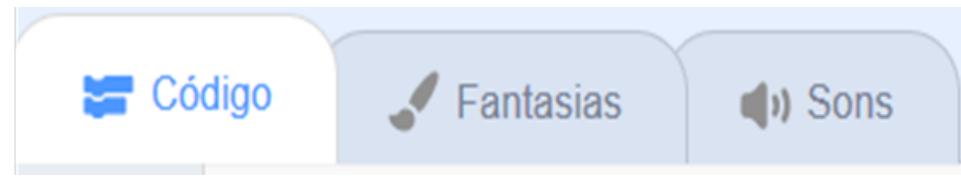
## 8 Projeto aplicado com Scratch

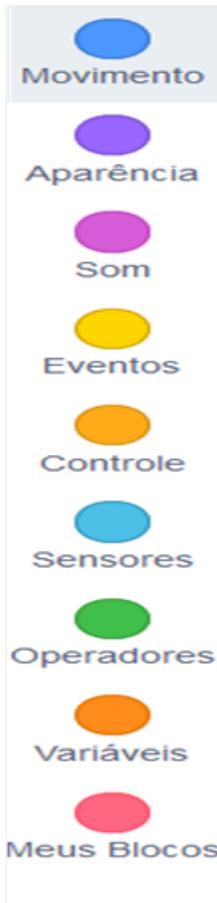


**Botões de ação:** Botões **Iniciar** (verde) e **Parar** (vermelho) script.

**Lista de Sprites:** Aqui são mostrados os **objetos usados na animação**. O objeto em edição fica selecionado.

**Opções para a área de script:** É possível navegar pelas abas para editar o código, as **fantasias dos personagens** e os **sons produzidos**.



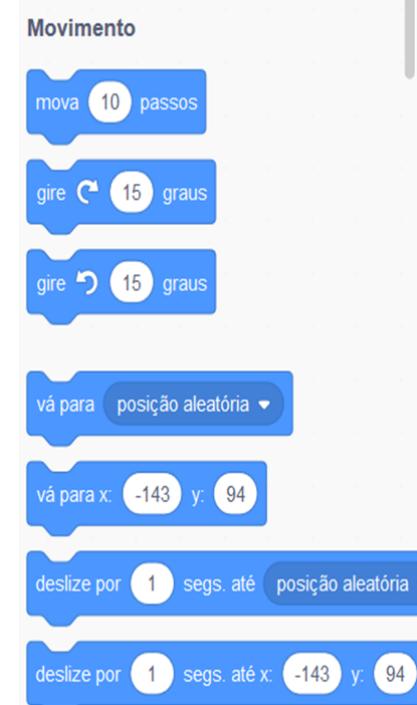
**Categorias de comando:**

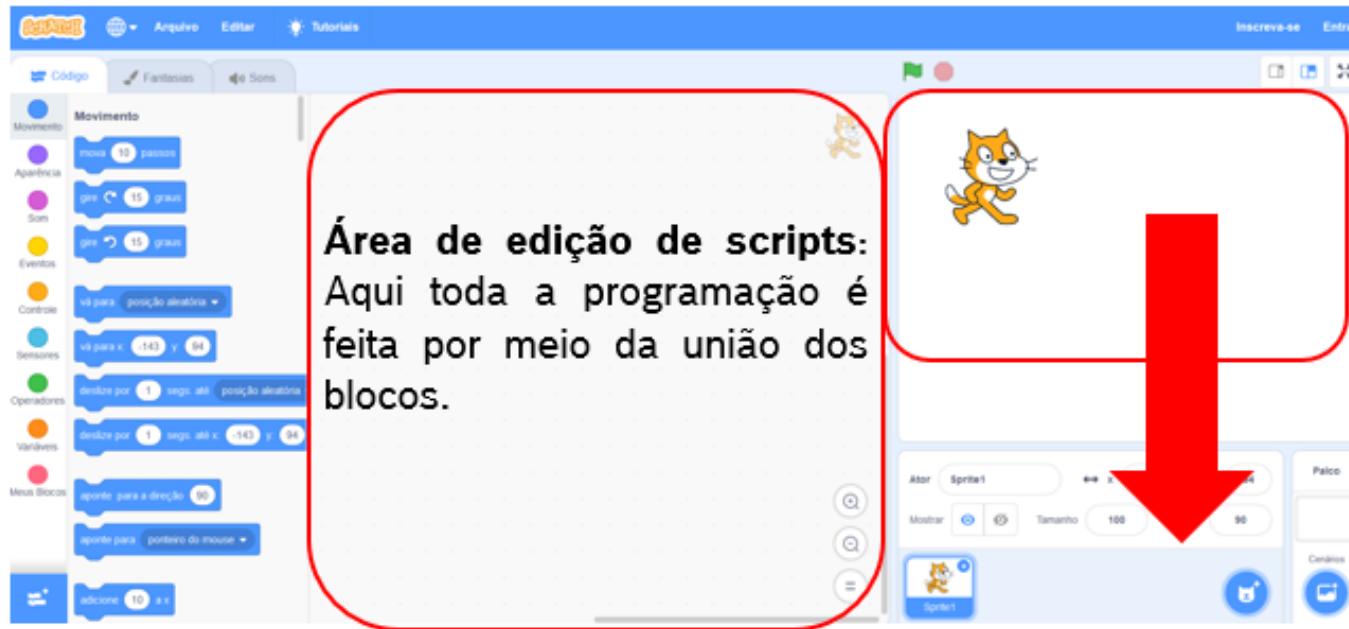
Os comandos são divididos em 9 categorias: **Movimento, Aparência, Som, Eventos, Controle, Sensores, Operadores, Variáveis e Meus blocos**

**Blocos de comando:** Aqui estão agrupados os **comandos que de fato farão parte do código**.

Sempre são mostrados os blocos de comando da categoria selecionada.

Ao lado, alguns exemplos de blocos de comando da categoria “**Movimento**”





**Área de edição de scripts:**  
Aqui toda a programação é feita por meio da união dos blocos.

**Palco:** Nesse espaço é possível ver o resultado da programação criada. O objeto inicial é o gato.



## 8.1 Exercício 3

Faça o gato andar uma certa distância e virar 90°;  
Faça isso 4 vezes para dar uma volta completa.

**Tempo estimado:** 5 minutos.

## 8.2 Correção

