



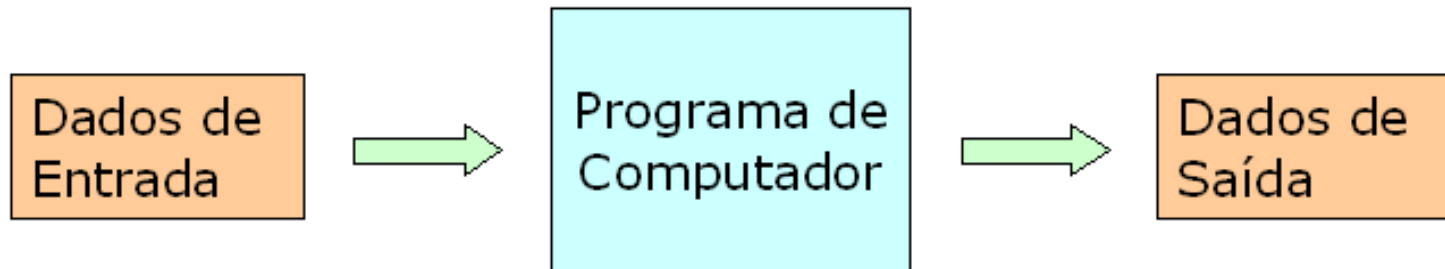
# Introdução à Programação

## Aula 01: Algoritmos

**Prof. Eduardo Corrêa**

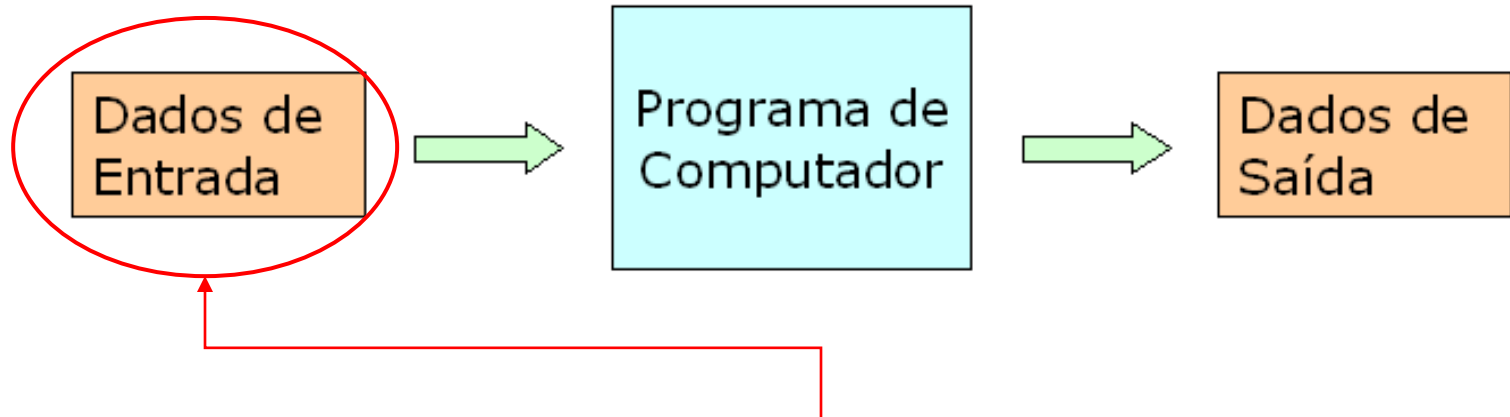
# O que é um Programa de Computador?

- De uma maneira geral, e especialmente na Estatística, um programa de computador funciona dessa maneira:



# O que é um Programa de Computador?

- De uma maneira geral, um programa de computador funciona dessa maneira:

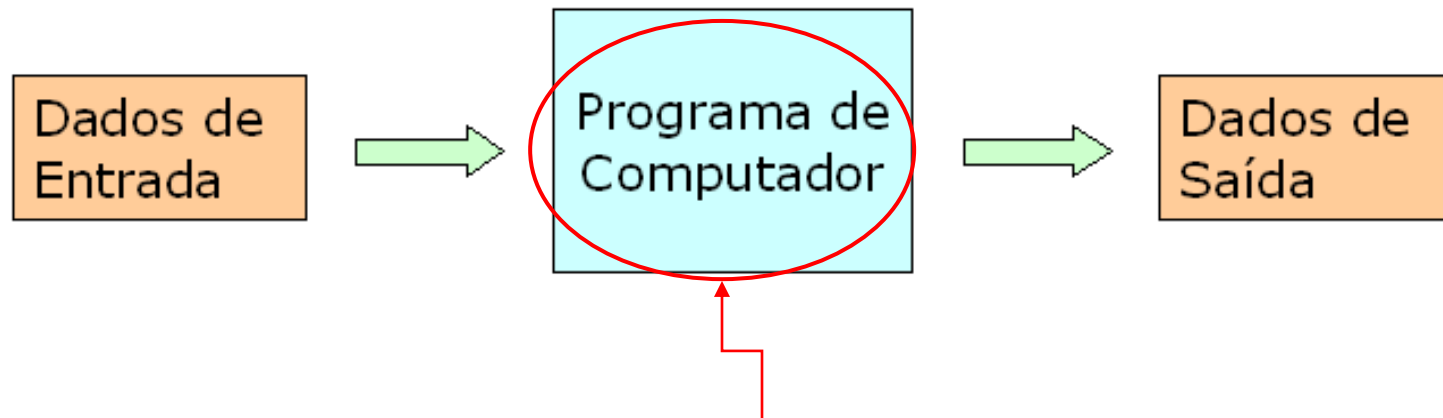


O programa toma algum valor ou conjunto de valores como **entrada**.

Estes valores podem, por exemplo, ser digitados pelo usuário via teclado, ou selecionados com o mouse, ou até mesmo obtidos por um microfone, câmera de vídeo e outros dispositivos.

# O que é um Programa de Computador?

- De uma maneira geral, um programa de computador funciona dessa maneira:

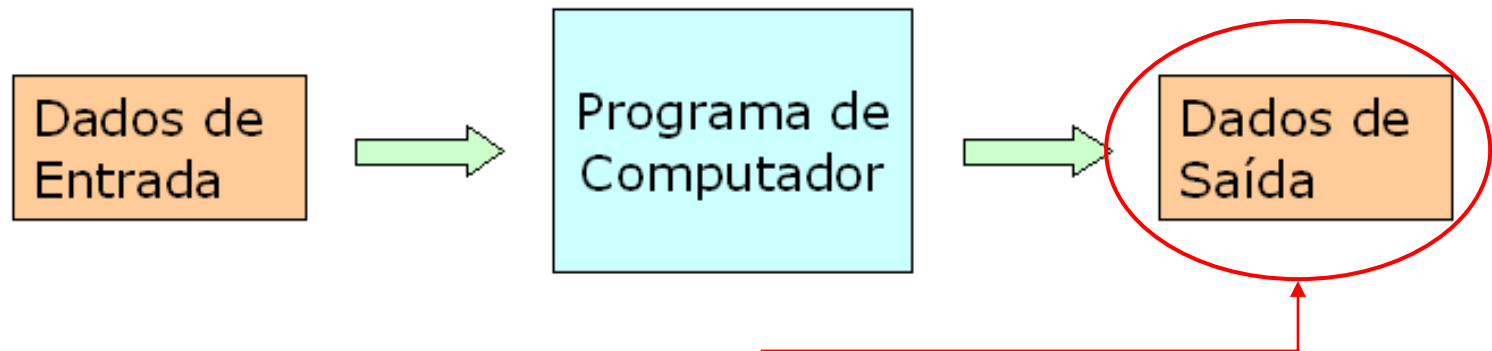


Em seguida, o programa **executa as tarefas** (sequência de **instruções**) especificadas pelo programador, levando em conta os dados passados como entrada.

Como resultado, será produzido algum valor ou conjunto de valores, de interesse do usuário.

# O que é um Programa de Computador?

- De uma maneira geral, um programa de computador funciona dessa maneira:



Os valores produzidos serão apresentados para o usuário. Estes valores são chamados de saída do programa.

Tipicamente a saída é mostrada no vídeo, mas também pode ser direcionada para a impressora ou outro dispositivo.

# O que é um Programa de Computador?

- **Exemplo:**
- Sistema de reconhecimento facial, que analisa a imagem do rosto de uma pessoa e informa se essa pessoa é um adulto ou uma criança.



**Programa de  
Computador**



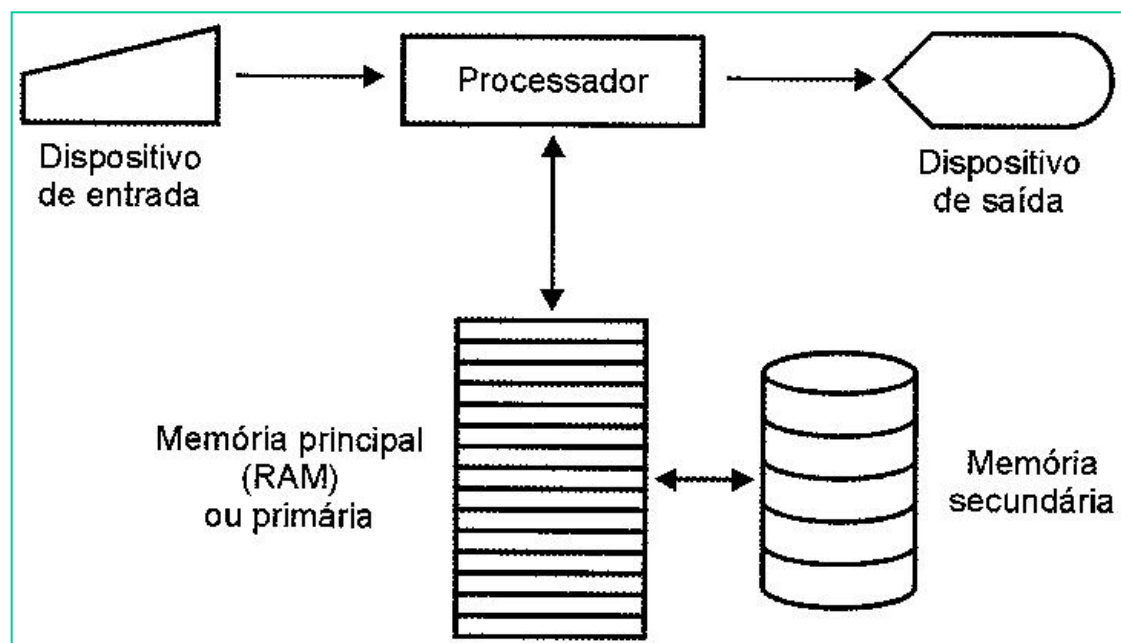
**Criança**

**Entrada: foto do rosto**  
(pode ser obtida  
com uma câmera)

**Saída: criança ou adulto**  
(neste exemplo, o rosto  
é de uma criança)

# Linguagem de Máquina (1/6)

- Como acabou de ser mostrado, o computador é uma máquina capaz de solucionar problemas através da execução de **instruções** que lhe são fornecidas em **programas**.
- O **processador** (ou **CPU**) é responsável por executar as instruções.



## Linguagem de Máquina (2/6)

- Entretanto, há um inconveniente: as instruções devem estar armazenadas na memória principal em **linguagem de máquina** ou **linguagem binária**.
- Essa é a “linguagem natural” do computador, ou seja, a única linguagem que ele consegue entender diretamente.
- Diferente do Português, possui apenas dois símbolos: **0** (zero) e **1** (um).

```
00000000101000100000000100011000
00000000100000100001000000100001
10001101111000100000000000000000
100011100001001000000000000000100
10101110000100100000000000000000
101011011110001000000000000000100
0000001111100000000000000000001000
```



# Linguagem de Máquina (3/6)

- Infelizmente os valores binários têm o comprimento muito extenso, o que torna a sua manipulação difícil para seres humanos.

Número Decimal (Base 10)	Número Binário (Base 2)
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111
100	1100100
1000	1111101000

# Linguagem de Máquina (4/6)

- A CPU de cada computador pode **reconhecer** e **executar diretamente** apenas instruções em linguagem de máquina.
- E, para tornar tudo mais difícil, estas instruções são extremamente simples:
  - Comparar dois dados.
  - Somar ou subtrair dois números.
  - Escrever um byte na memória ou num periférico de saída.

# Linguagem de Máquina (5/6)

- A sequência de 0's e 1's que formaliza uma determinada operação a ser realizada pelo processador denomina-se instrução de máquina.

**0100 00000001 11010001**

- No exemplo acima (hipotético) a instrução **0100** poderia representar a instrução para somar dois números.
  - Neste caso, os números são  $(00000001)_2$  e  $(11010001)_2$
  - Ou seja, essa instrução manda o computador realizar a soma **1 + 209**.
- Um **programa executável** é constituído de um **conjunto de instruções de máquina**.
  - Ex: arquivo **.exe** no Windows.

# Linguagem de Máquina (6/6)

```
00000000101000100000000100011000
00000000100000100001000000100001
10001101111000100000000000000000
100011100001001000000000000000100
101011100001001000000000000000000
101011011110001000000000000000100
00000011111000000000000000000100
```

## Programa em Linguagem de Máquina (binário)

- É possível programar diretamente em linguagem de máquina. No entanto isto não é feito, devido aos seguintes motivos:
  - ☹ Ela não é nada amigável para o programador. É difícil de escrever e entender programas nessa linguagem.
  - ☹ A implementação fica mais sujeita à erros.
  - ☹ Cada computador tem a sua própria linguagem de máquina.  
(**ex:** A instrução de adição pode ter o código "0100" no PC mas no Mac ter outro código.

# Linguagem de Alto Nível

- **Solução:** criar um outro conjunto de instruções que seja mais dirigido às pessoas e menos dirigido à máquina.
- Ou seja, linguagens que se aproximam mais da linguagem natural (**Inglês**, Português...).
- Este novo conjunto de instruções é conhecido como **Linguagem de Alto Nível**.
- **Ex:**
  - C
  - Java
  - Python
  - Julia
  - R
  - PHP
  - C#
  - Swift
  - ...

# Tradutores

- Os humanos escrevem programas numa linguagem de alto nível.
- Depois convertem para linguagem de máquina utilizando um **tradutor**.
- É um **programa especial** que **traduz** os comandos em linguagem **de alto nível** em longas séries de dígitos binários compreendidos pelo computador (ou seja, **para a linguagem de máquina**).

## Programa em Pascal

```
def trocar(x, y):  
    return y, x
```



**Tradutor  
Python**

```
000000001010001000000000100011000  
00000000100000100001000000100001  
10001101111000100000000000000000  
100011100001001000000000000000100  
101011100001001000000000000000000  
101011011110001000000000000000100  
0000001111100000000000000000001000
```

**Programa em Linguagem  
de Máquina (binário)**

# Linguagem de Alto Nível (1/2)

- 😊 Mais amigável para o ser humano.
- 😊 Diminui a chance de erro.
- 😊 Não preciso conhecer a linguagem de máquina de um modelo específico de computador e nem detalhes sobre sua arquitetura.
- 😞 Ainda não representa o maior nível de abstração (pensar em resolver um problema “diretamente” numa linguagem de programação não é tão trivial)

*Isso ainda será explicado!!!*

## Linguagem de Alto Nível (2/2)

- C, C++, Java, Scala, R, Python, Java, Julia, C#, PHP, JavaScript, Go, Swift, ...

### Por que existem tantas linguagens?

- Cada linguagem é adequada para um propósito diferente!
- **Exemplos:**
  - **R**, e **Python** e **Julia** são utilizadas para computação científica, análise de dados e estatística. Elas já vêm “de fábrica” ou podem ser facilmente estendidas com muitas funções estatísticas.
  - **Java** e **C** são linguagens de propósito geral. **Python** também é uma linguagem de propósito geral, mas possui ótimas bibliotecas para estatística.
  - **PHP** e **JavaScript** são utilizadas em aplicações Web.



# Algoritmos (1/3)

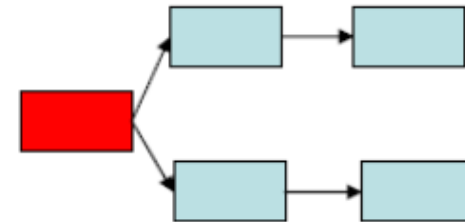
- Antes de começarmos a programar em Python é necessário conhecer o conceito de **algoritmo**.
- “uma **sequência finita** e **não ambígua** de instruções para executar uma tarefa específica”.
- É **parecido** com uma receita (ex.: receita culinária ou instruções para instalar um ventilador de teto), porém é **mais complexo**:
  - Pode incluir repetição de passos até que uma condição seja satisfeita.
  - Pode incluir testes de condições lógicas para executar a tarefa.
- Representa o **nível mais alto** de abstração para a solução de um problema. Um programa de computador é um algoritmo escrito numa dada linguagem.

## Algoritmos (2/3)

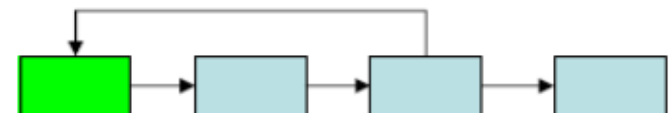
- Um algoritmo é uma **sequência** de passos. Cada passo é de um dos três tipos:

1. Uma operação elementar.

2. Uma operação de controle especificando uma **seleção** entre uma sequência de passos.



3. Uma operação de controle especificando uma **repetição** entre uma sequência de passos.



## Algoritmos (3/3)

- O algoritmo é “um cara legal” pois ele permite com que a gente possa pensar na solução de um problema sem que seja preciso se preocupar com os comandos da linguagem em que iremos programar.
- As principais técnicas para a elaboração de algoritmos são:
  - **Descrição Narrativa** (nem sempre é conveniente na prática, mas é excelente para começar a pensar resolver um exercício na prova).
  - **Fluxograma** (bem chato de fazer, quase ninguém utiliza hoje em dia).
  - **Pseudocódigo** (técnica mais usada na prática: é comum encontrar artigos expressos em pseudocódigo nos livros de estatística e computação).

# Descrição Narrativa

- Consiste em analisar o enunciado do problema e escrever, utilizando **linguagem natural** (Língua Portuguesa), os passos a serem seguidos para a resolução do problema.

😊 **vantagem:** não é necessário aprender nenhum conceito novo, pois todo mundo conhece uma linguagem natural.

😞 **desvantagem:** a língua natural é ambígua e imprecisa.

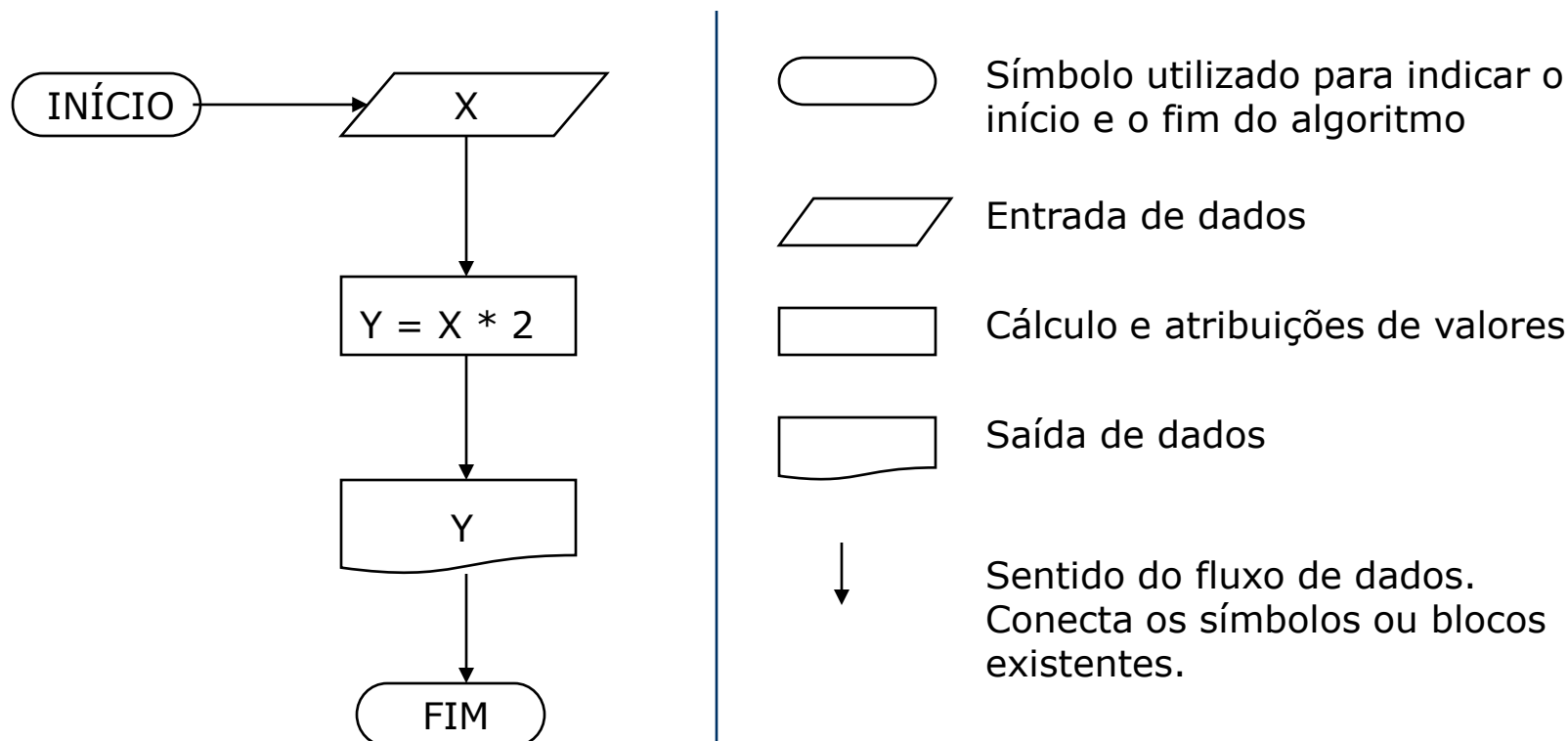
O que é “aguardar algum tempo” ?

## Exemplo: preparar um ovo frito

1. Colocar margarina na frigideira;
2. Acender o fogão;
3. Quebrar a casca do ovo;
4. Despejar o ovo na frigideira;
5. Aguardar algum tempo;
6. Retirar o ovo da frigideira;
7. Apagar o fogão.

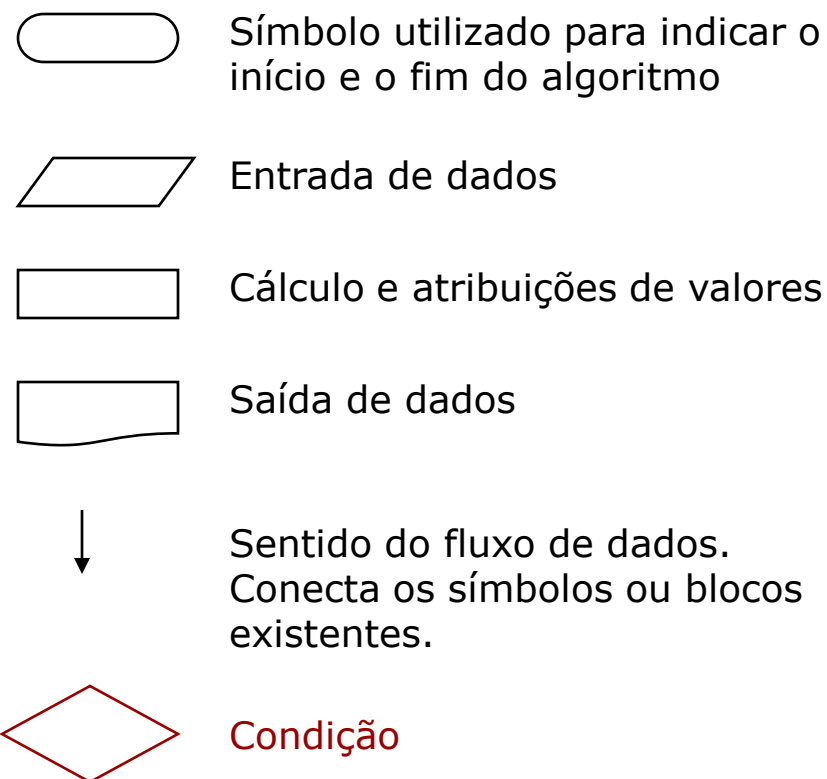
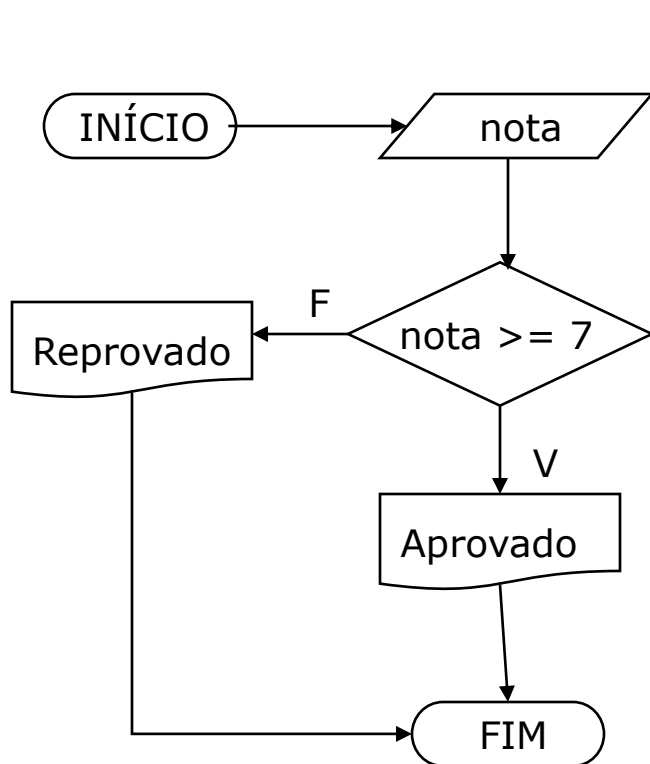
# Fluxograma (1/3)

- Consiste em analisar o enunciado do problema e escrever, utilizando **símbolos gráficos predefinidos**, os passos a serem seguidos para a resolução do problema.
- **Exemplo:** algoritmo para calcular o dobro de um número.



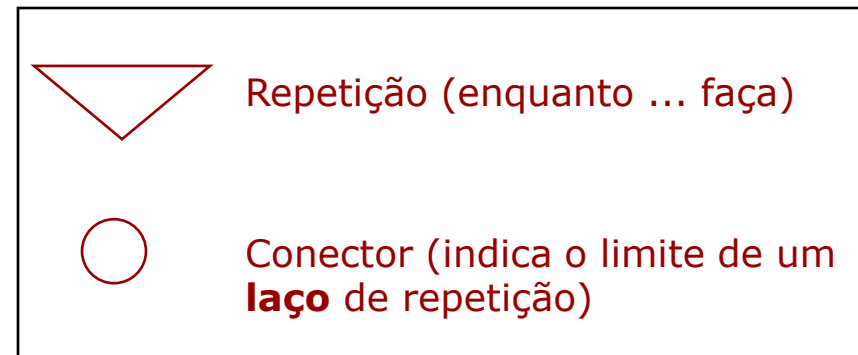
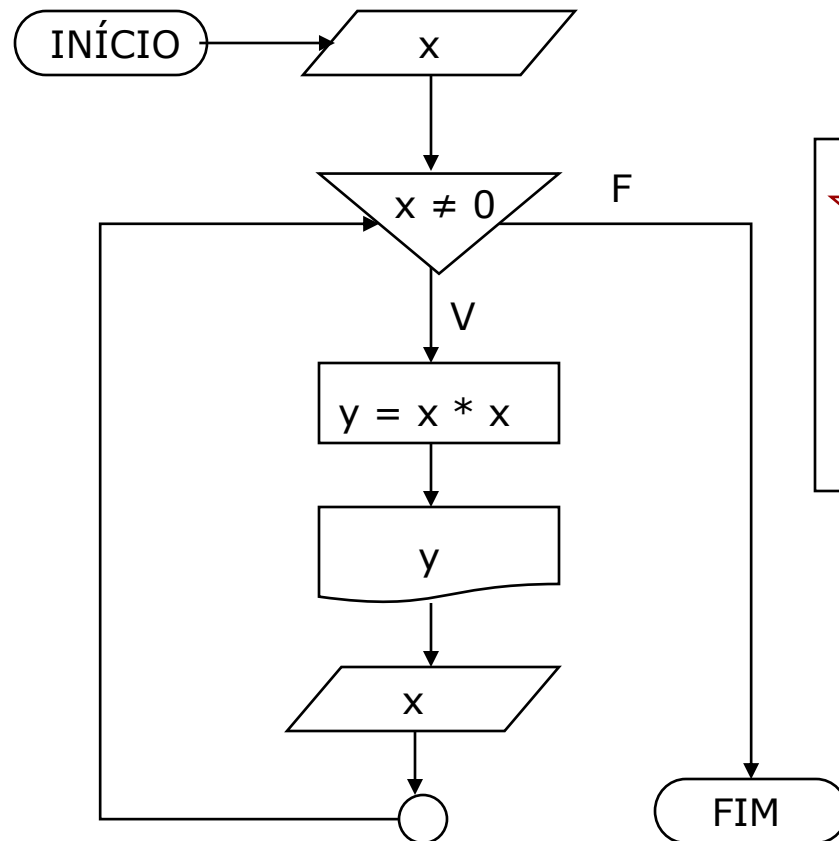
## Fluxograma (2/3)

- **Desvio condicional:** permite escolha do grupo de instruções a ser executado.
- **Ex.:** algoritmo que recebe nota como entrada e imprime situação do aluno.
- Agora o algoritmo tem **2 caminhos diferentes!!!**



## Fluxograma (3/3)

- **Repetição:** permite que um conjunto de passos seja executado repetidas vezes até que uma condição seja atingida.
- **Ex:** algoritmo que lê números como entrada e calcula o quadrado do número **até que** o usuário digite o valor '0' no teclado.



# Pseudocódigo (1/5)

- É uma linguagem próxima à linguagem natural para descrever algoritmos.
- Porém, assim como ocorre com os fluxogramas, ela possui **comandos predefinidos**, que a tornam mais precisa do que a linguagem natural.
- **Eficaz**: Permite com que o programador possa projetar uma solução para seu problema que seja facilmente implementada no computador (diferente do que o ocorre com a Descrição Narrativa).
- **Popular**: Os livros de computação e estatística, assim como os artigos científicos, costumam utilizar pseudocódigo para descrever algoritmos.



## Pseudocódigo (2/5)

- Há vários tipos de notação para pseudocódigo.
- Adotaremos a seguinte abordagem: trocar os símbolos do fluxograma por palavras em Português com nomes similares aos dos comandos reais da linguagem Pascal

**início** : início do algoritmo.

**fim** : fim do algoritmo.

**ler**: para entrada de dados.

**imprimir**: para saída de dados.

**se, senão**: para desvio condicional.

**enquanto**: para laço de repetição.

## Pseudocódigo (3/5)

- **Exemplo 1:** algoritmo em pseudocódigo para calcular o dobro de um número.

```
início  
  ler (n)  
  dobro = n * 2  
  imprimir (dobro)  
fim.
```

## Pseudocódigo (4/5)

- **Exemplo 2:** algoritmo em pseudocódigo que recebe nota final de aluno como entrada e imprime situação do aluno.
  - Se a nota for maior ou igual a 7,0 imprime "Aprovado"
  - Caso contrário, imprime "Reprovado"

```
início
  ler(nota)
  se nota >= 7.0 então:
    imprimir('Aprovado')
  senão:
    imprimir('Reprovado')
fim.
```

## Pseudocódigo (5/5)

- **Exemplo 3:** algoritmo em pseudocódigo para resolver o seguinte problema:
  - Ler um número como entrada e calcular e imprimir o valor do fatorial deste número.

```
início
    ler(n)
    fatorial = 1
    k = 1
    enquanto k < n faça:
        fatorial = fatorial * k
        k = k + 1

    imprimir(fatorial)
fim.
```

# Conclusão

- As 3 técnicas para a elaboração de algoritmos são:
  - **Descrição Narrativa**
    - Pode dar margem a ambiguidades e imprecisões.
    - Porém, é excelente para começar a pensar resolver um problema ou um exercício da prova.
  - **Fluxograma:**
    - Pouco utilizado nos dias atuais.
    - Frequentemente gera diagramas muito grandes (que não cabem em uma página)
  - **Pseudocódigo**
    - Técnica mais usada.
    - Não é tão diferente da descrição narrativa, mas resolve problemas de ambiguidades e imprecisões.
    - Os livros de estatística e computação descrevem algoritmos em pseudocódigo.

# Exercícios

Exercícios **para casa**:

(1) Elabore os seguintes algoritmos em **pseudocódigo**:

- (a) Receber como entrada o raio da esfera e imprimir o valor do volume.
- (b) Ler a distância a ser percorrida em Km e o consumo de um carro (em Km por litro) e, em seguida, calcular e exibir quantos litros são necessários para completar a distância.
- (c) ler 2 notas, calcular e exibir a média. Se a média for maior ou igual a 7, imprimir "aprovado". Se for menor do que 7 e maior ou igual a 3 imprimir "prova final". Se for menor do que 3 imprimir "Reprovado".
- (d) ler o valor de N como entrada e calcular e imprimir o valor de H, sendo:  $H = 1 + (1 / 2) + (1 / 3) + \dots + (1 / N)$ .

(2) Converta algoritmo em pseudocódigo para cálculo do fatorial (slide 28) para a notação em **fluxograma**.