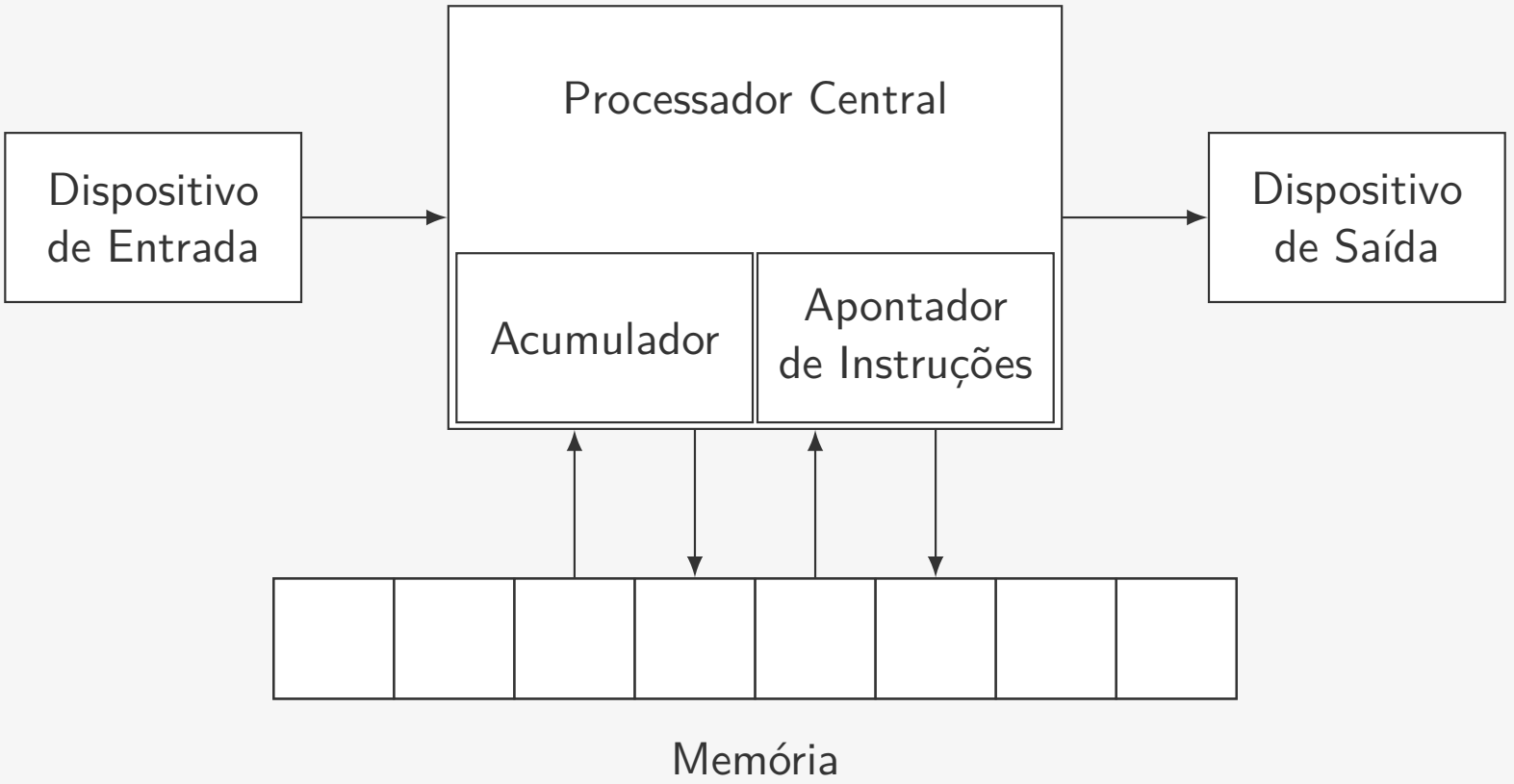
MC102 — Introdução

Rafael C. S. Schouery
rafael@ic.unicamp.br

Universidade Estadual de Campinas

Atualizado em: 2025-03-10 14:26

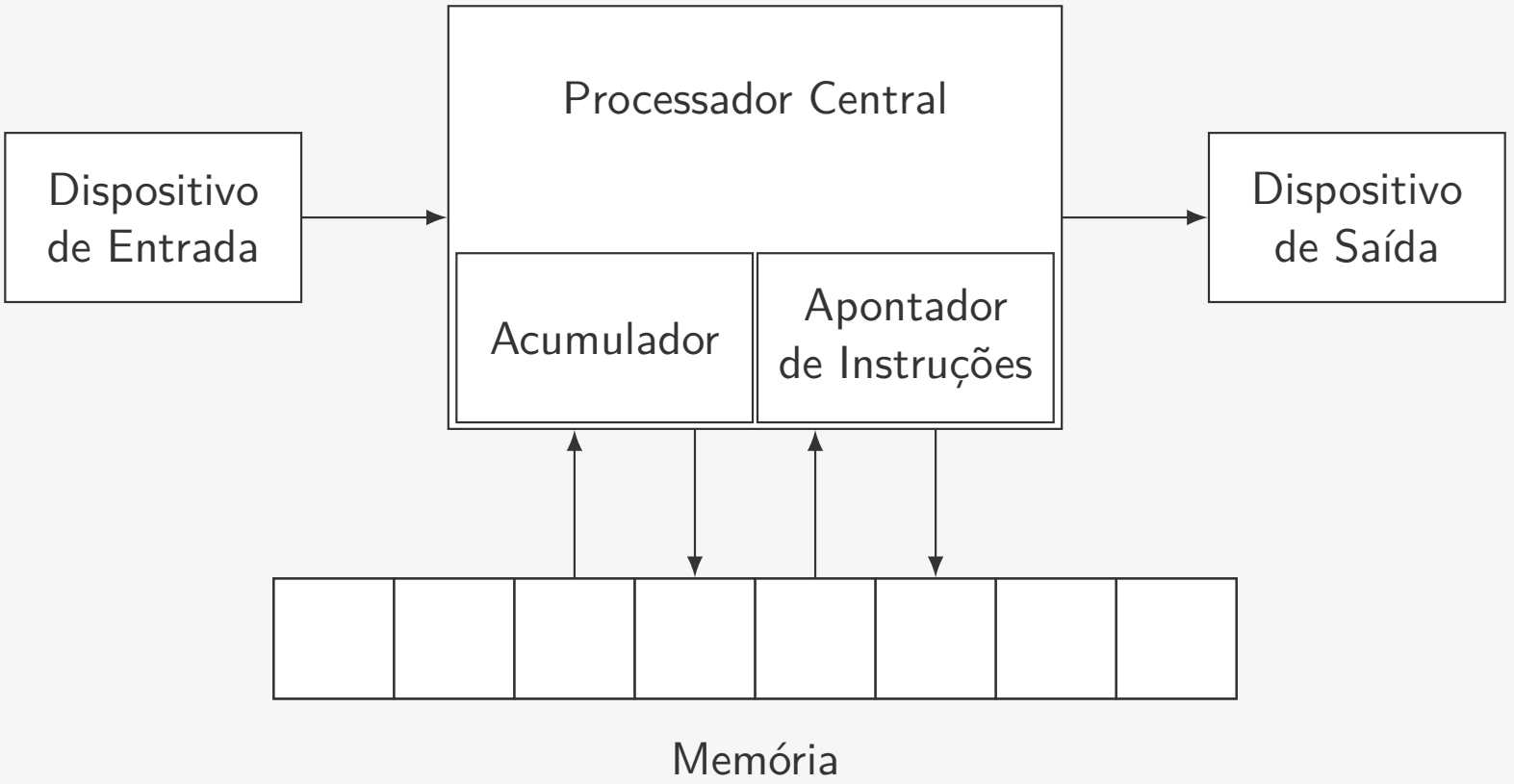
Arquitetura de Von Neumann



Dispositivos de Entrada

- Teclado
- Mouse
- HD (Disco rígido) / SSD (Disco de Estado Sólido)
- Tela Touch
- Rede
- Microfone
- Câmera
- Sensores (Temperatura, Batimento Cardíaco, etc)
- etc...

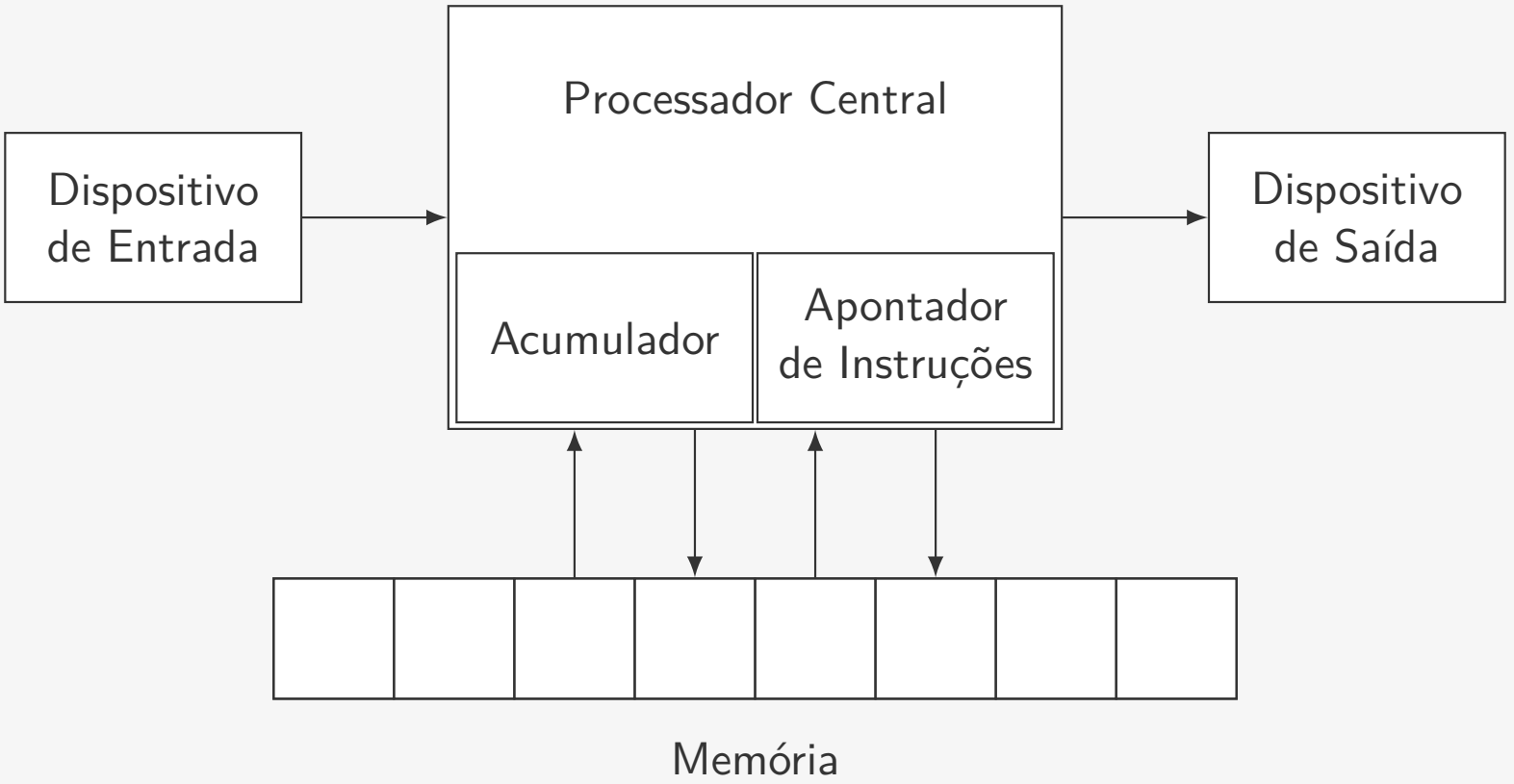
Arquitetura de Von Neumann



Dispositivos de Saída

- Tela
- Impressora
- HD (Disco rígido) / SSD (Disco de Estado Sólido)
- Rede
- Placa de Som
- Vibração
- Feedback Tátil
- etc...

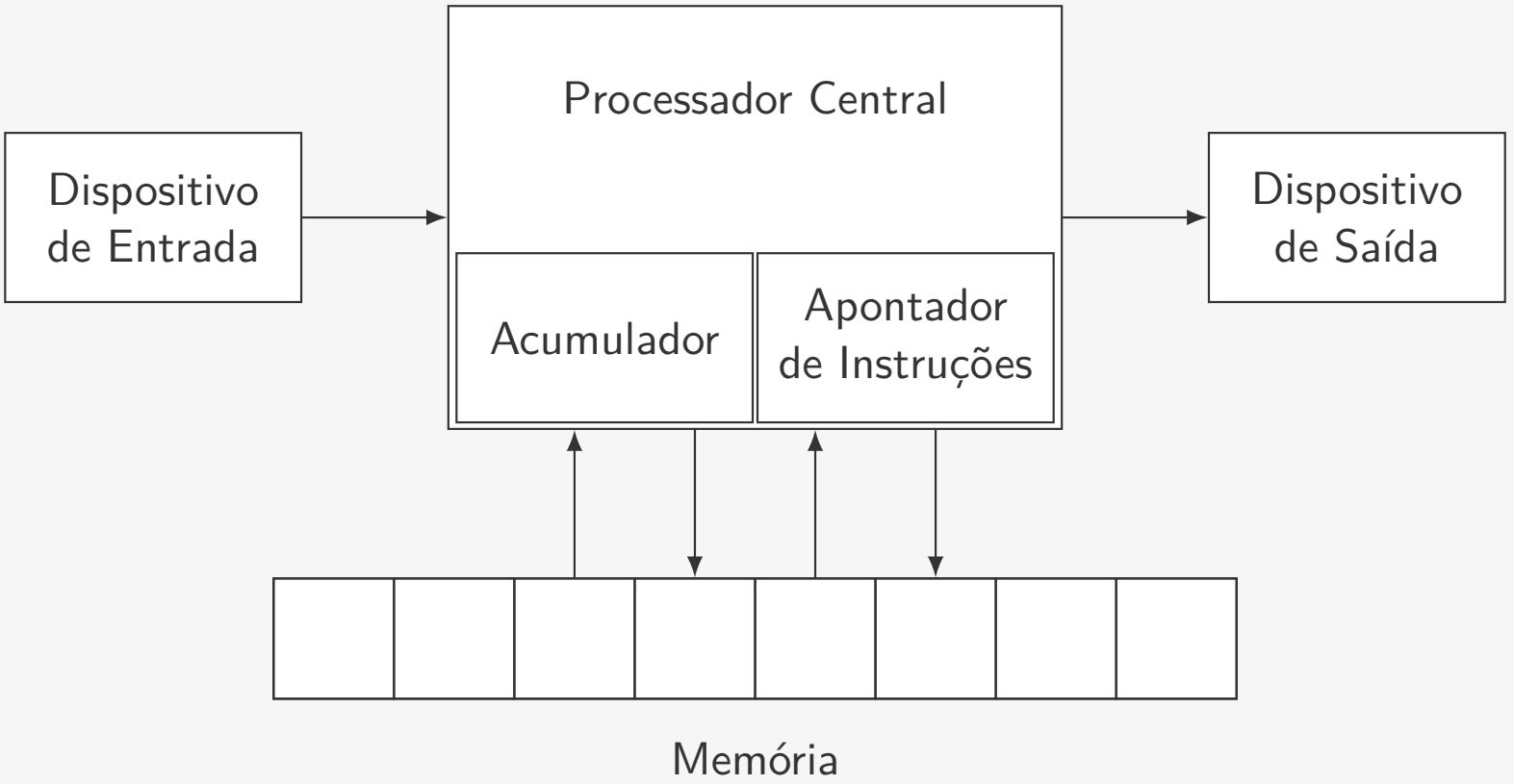
Arquitetura de Von Neumann



Memória

- Cada posição da memória guarda um dado
- Podemos ler ou escrever dados na memória
- Toda posição de memória tem um endereço
  - É assim que ela é acessada
- As instruções a serem executadas pelo processador também ficam na memória

Arquitetura de Von Neumann

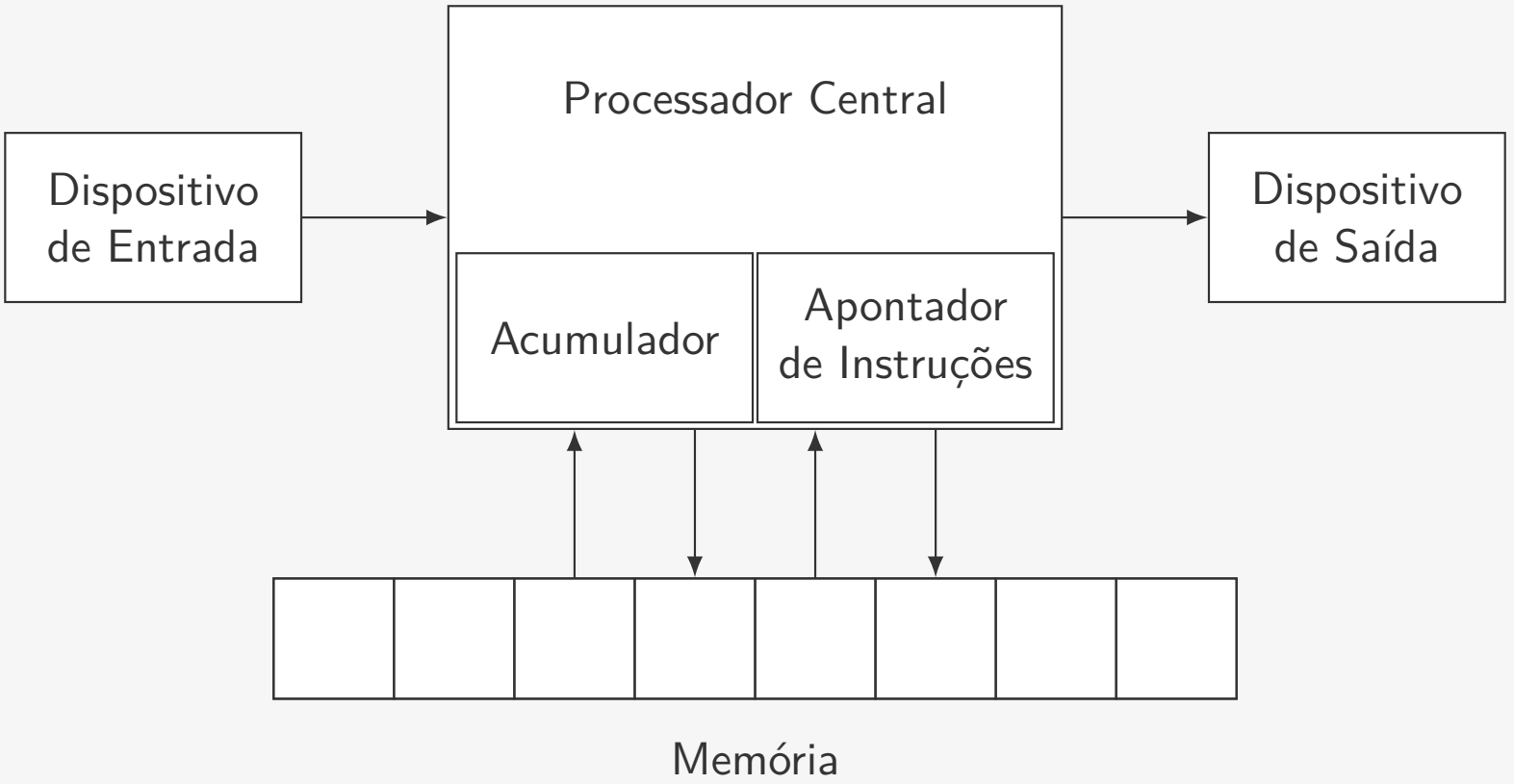


Processador central

Coordena o funcionamento do computador:

- analisa e executa cada instrução
- obtém da memória os dados necessário para executar instruções e coloca resultados na memória
- quando copia informações da memória, não as destrói, elas podem ser utilizadas novamente
- ativa equipamentos de entrada e saída

Arquitetura de Von Neumann



Apontador de Instruções e Acumulador

Dentro do Processador Central temos dois elementos importantes:

- Apontador de Instruções
  - Indica a posição da memória onde está a próxima instrução a ser executada
  - É atualizado pelo Processador Central após a execução da instrução
- Acumulador
  - Funciona como se fosse uma posição de memória
  - Quando uma operação aritmética (como `+`, `-`, `*` e `/`) é executada, um dos operandos deve estar no acumulador e o resultado da operação é colocado no acumulador

Computador a papel

É composto de:

- processador
- apontador de instruções
- acumulador
- memória de 16 posições
- teclado
- tela

Preciso de 22 voluntários!

Abreviaturas:

- AC = Acumulador
- [AC] = Conteúdo do acumulador
- End XX = endereço XX
- [End XX] = conteúdo do end XX

Programa executado

```
1  Carregue zero no AC
2  Armazene o [AC] no end 16
3  Leia um número e armazena no end 15
4  Escreva [end 15]
5  Carregue no AC [end 15]
6  Se [AC] = 0 desvie para end 13
7  Carregue o [end 16] no AC
8  Some [AC] ao [end 15] e armazene resultado no AC
9  Armazene [AC] no end 16
10 Leia um número e armazene no end 15
11 Escreva [end 15]
12 Desvie para o end 05
13 Escreva [end 16]
14 Pare
15 << guarda o número lido >>
16 << guarda a soma parcial >>
```

Pseudocódigo

Um pseudocódigo é uma forma abstrata de escrever o programa

- Não estamos preocupados com a linguagem
- Mas sim com a solução do problema
- E a clareza

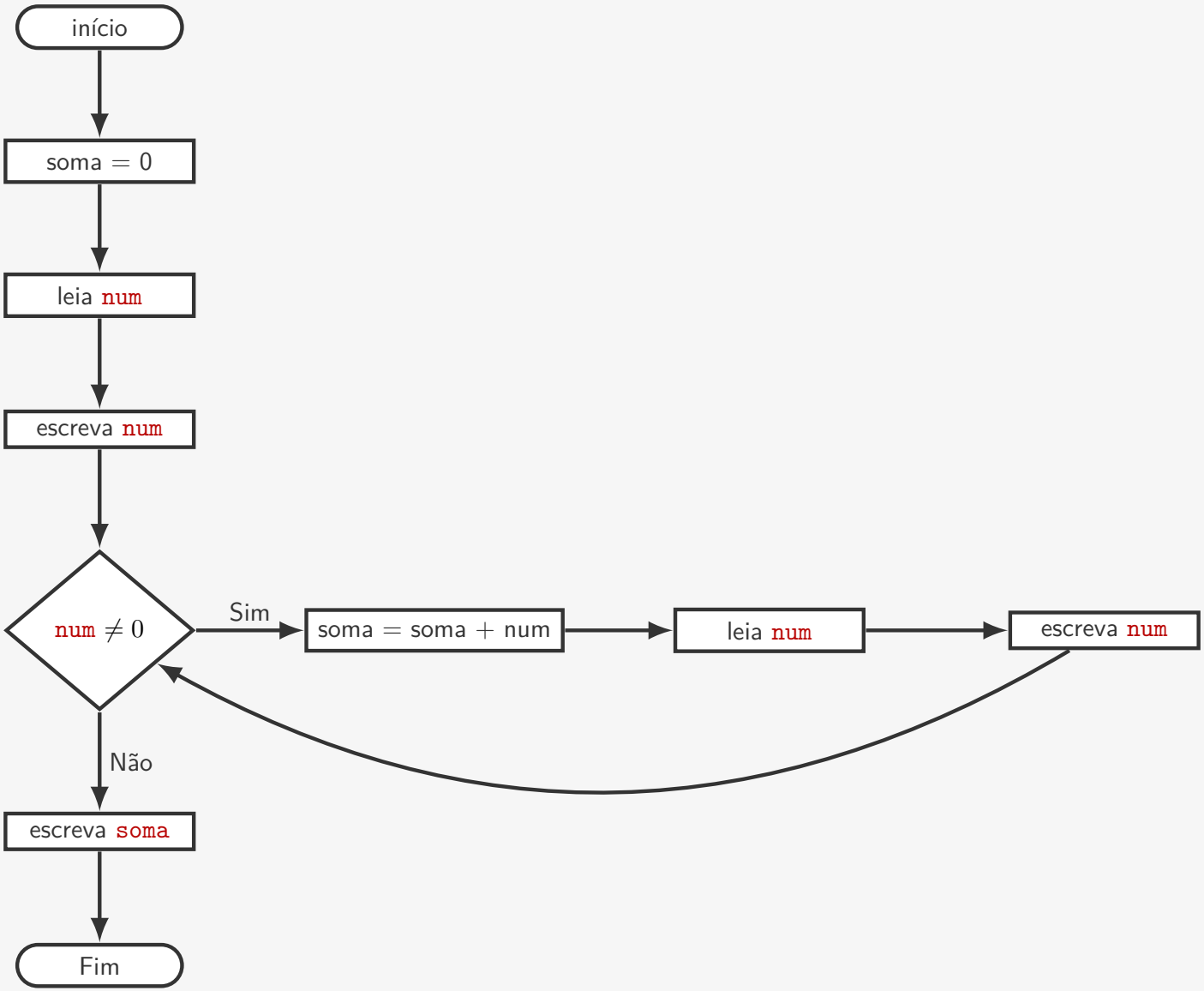
```
1 soma = 0
2 leia num
3 escreva num
4 enquanto num for diferente de zero
5     soma = soma + num
6     leia num
7     escreva num
8 escreva soma
```

Observações:

- O = é usado em um sentido diferente da matemática
- Os passos são simples e claros

Fluxograma

A ideia é representar as instruções graficamente



Em Python

Esse é um código em Python que faz a mesma coisa

```
1  num = int(input("Entre com o número: "))
2  print("numero =", num)
3  soma = 0
4
5  while num != 0:
6      soma = soma + num
7      num = int(input("Entre com o número: "))
8      print("numero =", num)
9
10 print("A soma é", soma)
```

A semelhança com o pseudocódigo é grande, mas tem algumas diferenças...

- Existem regras de como escrever um código em Python

Vamos ver esse código executando!

Algoritmos

Informalmente, um algoritmo é uma sequência de passos que realiza uma tarefa

- Os passos precisam ser simples

Acabamos de ver um algoritmo (representado de três formas) que soma números até que o número zero seja dado

O algoritmo não depende da linguagem de programação!

- Pode ser escrito também em pseudocódigo (mais comum) ou com um fluxograma (muito pouco usado, mas didático)
- Pode ser usado e reutilizado nas mais variadas linguagens!

Estamos interessados não apenas em aprender Python

- Mas aprender a projetar algoritmos!
- Você ainda será um programador se Python ficar obsoleto!

Hardware e Software

Hardware é toda a parte física da computação:

- Processador, memória, periféricos, etc.

Software é toda a parte lógica da computação:

- Os programas que podem ser executados

Uma máquina de zeros e uns

- O Bit é a menor unidade de medida do computador
  - Apenas dois valores: 0 ou 1
- Um Byte é uma sequência de 8 Bits
- Em geral, uma posição da memória armazena um Byte
- Tudo no computador são zeros e uns!!!
  - Mesmo textos, imagens, sons, vídeos, etc...
  - A diferença é como nós interpretamos os Bytes!

Ordens de magnitude

Quantidade	Sigla	Nome
8 bits	B	byte
1000 B	kB	kilobyte
1000 kB	MB	megabyte
1000 MB	GB	gigabyte
1000 GB	TB	terabyte
1000 TB	PB	petabyte
1000 PB	EB	exabyte
1000 EB	ZB	zettabyte
1000 ZB	YB	yottabyte

Há também kiB, MiB, GiB, etc que são 1024 bytes, 1024 kib, 1024 MiB, etc...

- Dizemos kibibyte, mebibyte, gibibyte, etc
- Às vezes kiB é escrito como KB
- E ainda há muita confusão se é múltiplo de 1000 ou 1024 por razões históricas

Hierarquia de Memória

Em um computador normal, temos várias memórias diferentes:

- Memória Permanente
  - Em geral, Disco Rígido ou de Estado Sólido
  - É a memória de acesso mais lento
  - Mais barata e com maior capacidade
  - Armazena os dados mesmo se não houver energia
- RAM (Random-Access Memory)
  - Bem mais rápida que a Memória Permanente
  - Mas mais cara e com menor capacidade
  - Dados da Memória Permanente são copiados para a RAM
  - Perde os dados se não houver energia
- Memória Cache
  - É a memória mais rápida
  - Faz parte do chip do processador (não pode ser comprada)
  - Capacidade bem pequena
  - Dados da RAM são copiados para a Cache
  - Perde os dados se não houver energia

Programa

Um programa é uma sequência de instruções que especificam como executar uma computação

- Ex: calcular a soma de números até que o zero seja digitado
- É escrito em uma determinada linguagem
  - Chamada de linguagem de programação
  - Ex: Python, C, C++, Java, etc...
- O texto do programa é chamado de código fonte

Exemplo de duas linguagens diferentes

Python

```python
num = int(input("Entre com o número: "))
print("numero =", num)
soma = 0

while num != 0:
    soma = soma + num
    num = int(input("Entre com o número: "))
    print("numero =", num)

print("A soma é", soma)
```

C

```c
#include <stdlib.h>

int main() {
    int num, soma = 0;
    scanf("%d", &num);
    printf("numero = %d", num);
    while (num != 0) {
        soma = soma + num;
        scanf("%d", &num);
        printf("numero = %d", num);
    }
    printf("A soma é ", soma);
    return 0;
}
```

Organização do Ambiente Computacional

Sistema Operacional:

- Conjunto de programas que gerencia o Hardware
- Ex: Linux, macOS, Windows, Android, iOS
- Permite que outros programas usem recursos de maneira segura e organizada

Aplicativos:

- São os programas que os usuários usam
- Ex: Editor de Texto, Navegador, Player de Música
- Faremos novos aplicativos (simples) no curso

Compiladores/Interpretadores:

- São responsáveis por transformar o código fonte em aplicativos
- De fato, também são aplicativos!

Linguagem Interpretada

- Ex: Python, Javascript, Perl, Ruby, PHP
- O interpretador abre o código fonte como um arquivo
- O interpretador executa uma instrução do código fonte por vez
- As instruções no código fonte são traduzida (interpretadas) para comandos do processador
- Muitas vezes chamamos o código fonte de *scripts*
- Em geral, as linguagens são mais expressivas

Linguagem Compilada

- Ex: C, C++, BASIC, COBOL, FORTRAN
- O código fonte é transformado em um aplicação (executável)
- Não depende mais do compilador para ser executado