Funcionamento de um computador

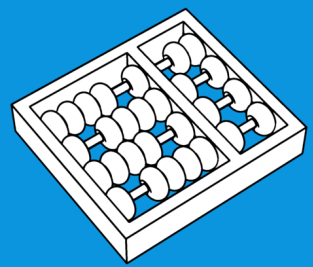
MC102 - Algoritmos e Programação de Computadores

03/24

1

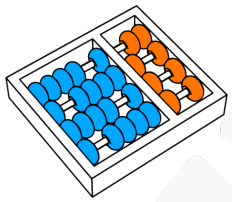
Santiago Valdés Ravelo
https://ic.unicamp.br/~santiago/
ravelo@unicamp.br

UNICAMP



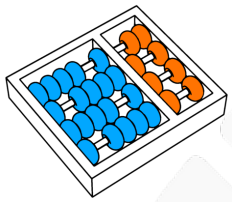
"É mais fácil viajar de avião, até mesmo pilotá-lo, do que entender por que ele pode voar."

John Von Neumann.



Pergunta

Como funciona um computador?

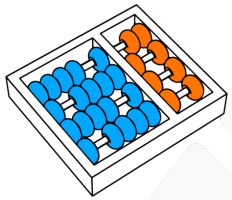


John Von Neumann (28/12/1903 – 08/02/1957)

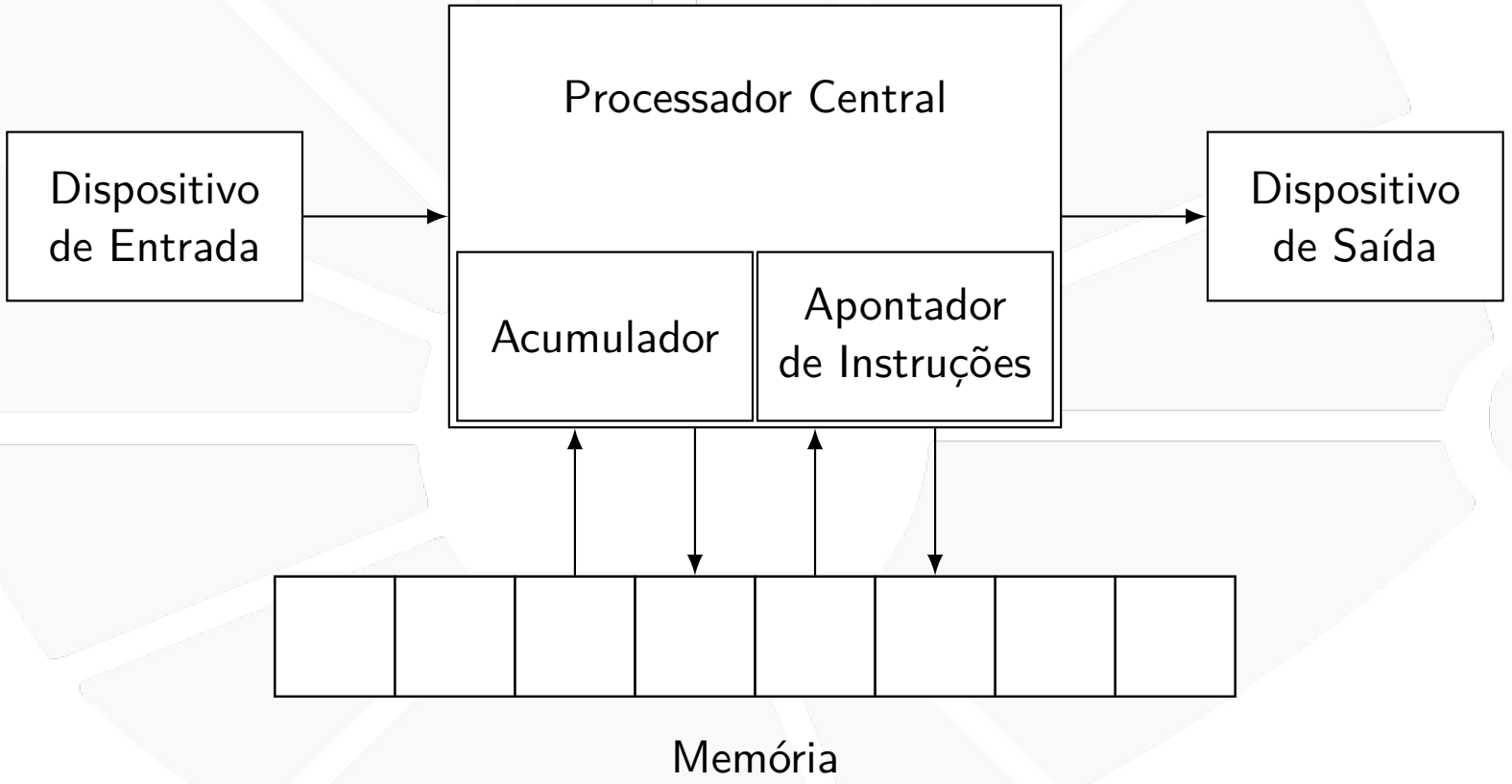
Principais áreas de atuação:

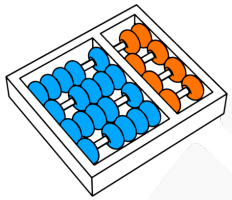
- Computação.
- Teoria dos números.
- Mecânica quântica.
- Teoria dos jogos.

Arquitetura de Von Neumann



Modelo


Processador Central
Dispositivo de Entrada
Dispositivo de Saída
Acumulador
Apontador de Instruções
Memória



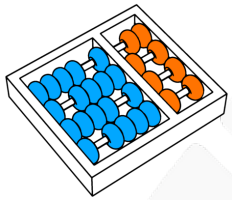
Dispositivos de Entrada

Responsabilidades:

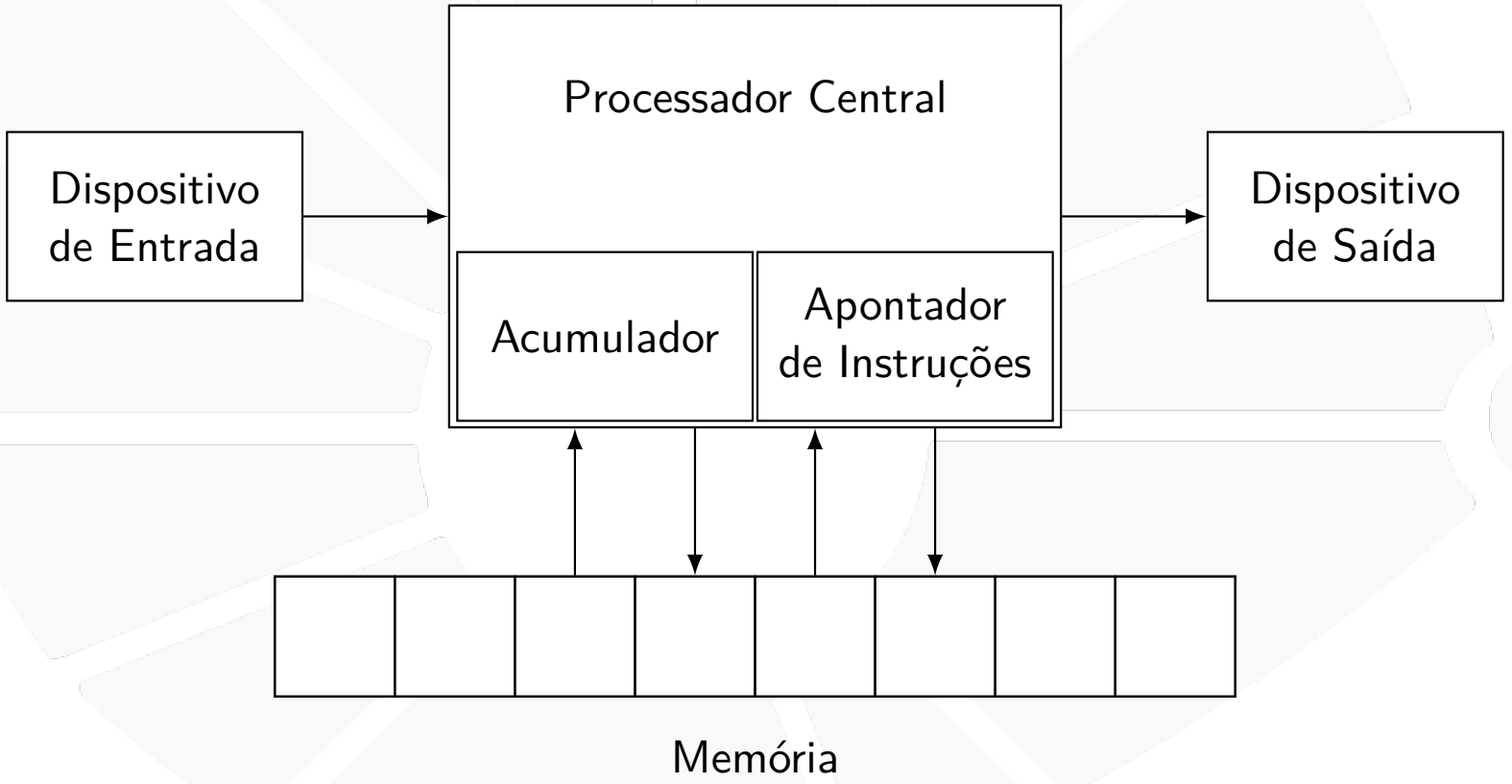
- Receber informações.

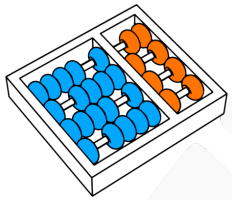
Alguns exemplos:

- Teclado.
- Mouse.
- HD (Disco rígido) / SSD (Disco de Estado Sólido).
- Tela Touch.
- Rede.
- Microfone.
- Câmera.
- Sensores (temperatura, batimento cardíaco, etc).



Modelo


Processador Central
Dispositivo de Entrada
Dispositivo de Saída
Acumulador
Apontador de Instruções
Memória



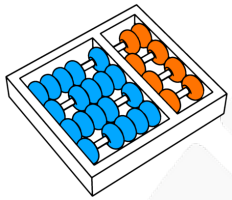
Dispositivos de Saída

Responsabilidades:

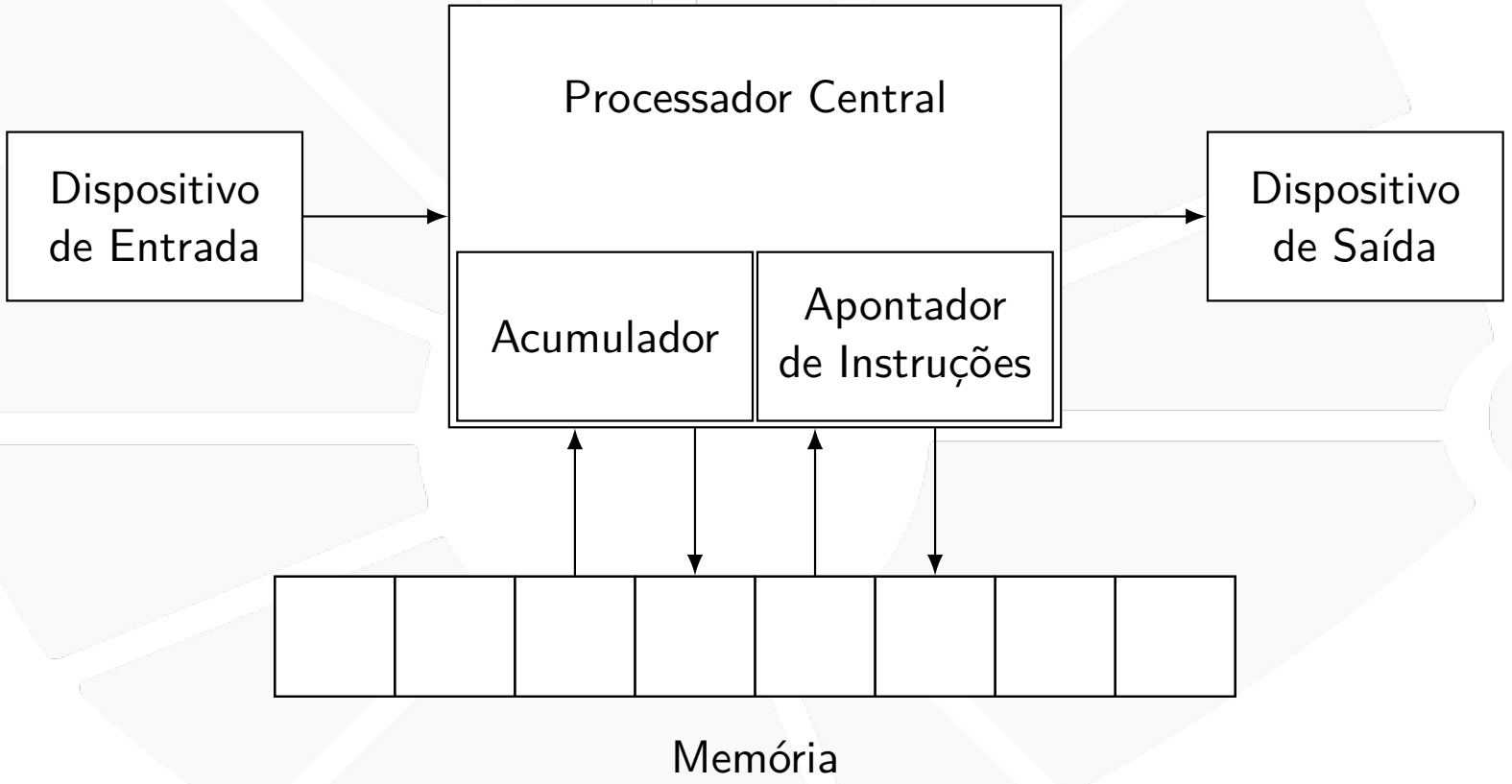
- Exibir resultados.

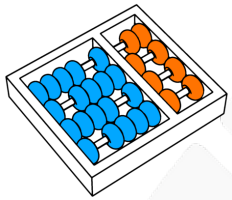
Alguns exemplos:

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



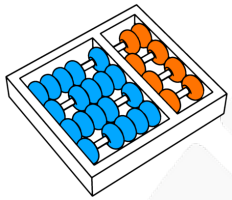
Modelo


Processador Central
Dispositivo de Entrada
Dispositivo de Saída
Acumulador
Apontador de Instruções
Memória

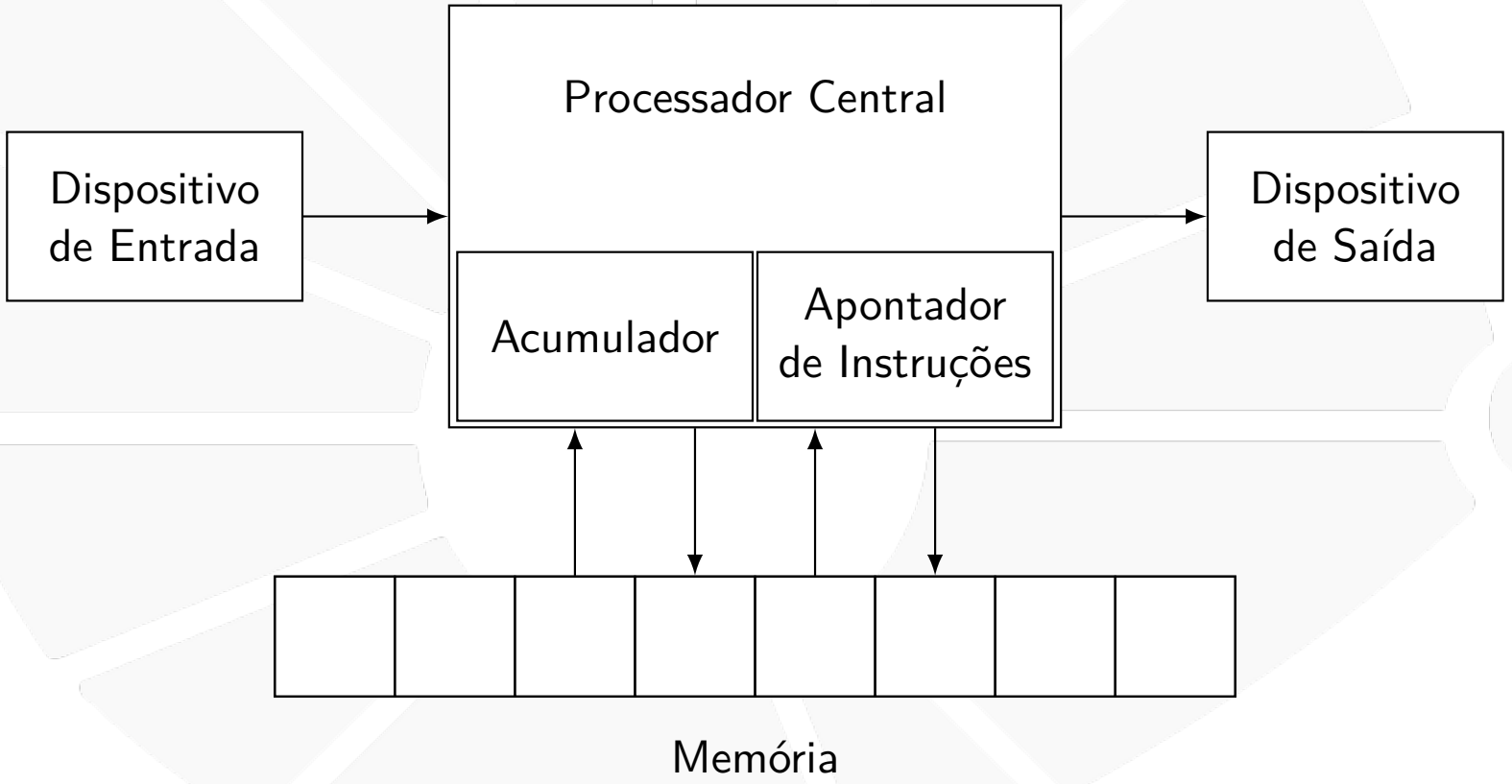


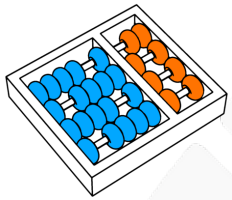
Memória

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



Modelo

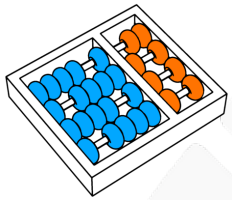

Processador Central
Dispositivo de Entrada
Dispositivo de Saída
Acumulador
Apontador de Instruções
Memória



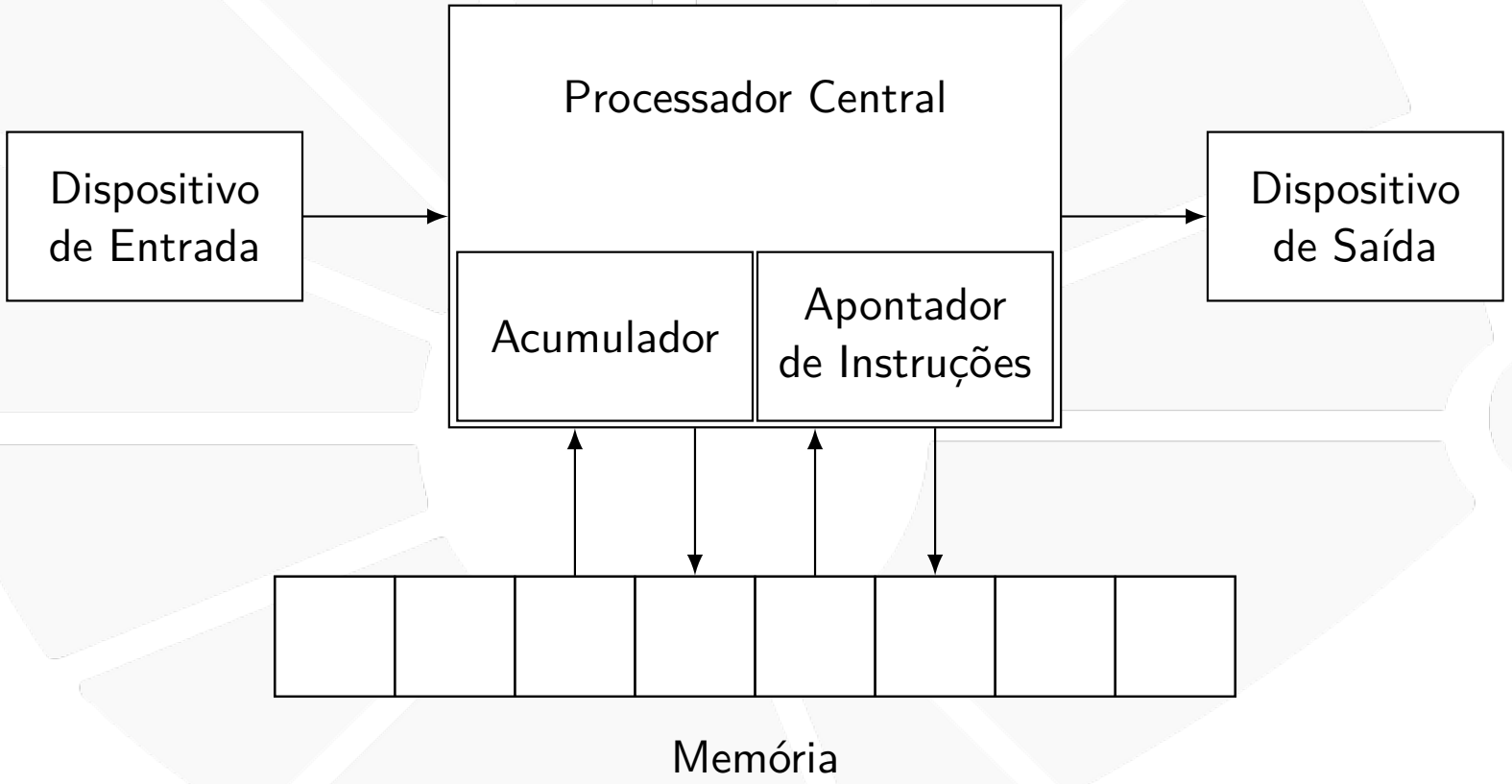
Processador central

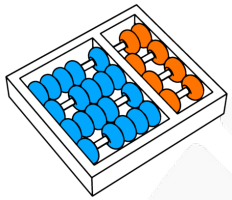
Coordena o funcionamento do computador:

- 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.



Modelo

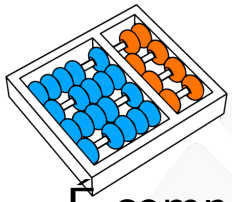

Processador Central
Dispositivo de Entrada
Dispositivo de Saída
Acumulador
Apontador de Instruções
Memória



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

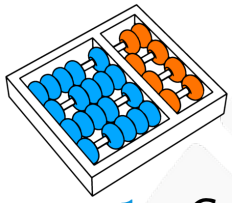
É composto de:

- Processador.
- Apontador de instruções.
- Acumulador.
- Memória de 16 posições.
- Teclado.
- Tela.

Preciso de 22 **voluntários!**

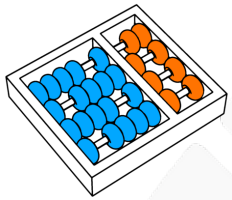
Abreviaturas:

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



Programa a executar

1. Carregue zero no AC
2. Armazene o AC no end 16
3. Leia um número e armazene no end 15
4. Escreva [end 15]
5. Carregue no AC [end 15]
6. Se [AC] = 0, desvie para end 13
7. Carregue no AC [end 16]
8. Some [AC] ao [end 15] e armazene o 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 end 05
13. Escreva [end 16]
14. Pare
15.
16.



Exemplo de duas linguagens diferentes

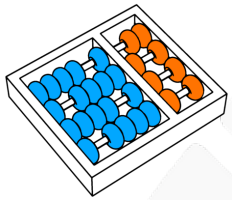
Algoritmo

```
1  soma ← 0
2  leia num
3  escreva num
4  enquanto num ≠ 0
5      soma ← soma + num
6      leia num
7      escreva num
8  escreva soma
```

Python

```
1  soma = 0
2  num = int(input())
3  print(num)
4  while(num != 0):
5      soma = soma + num
6      num = int(input())
7      print(num)
8  print(soma)
```

Ambiente computacional



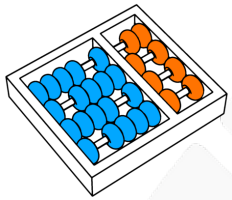
Hardware e Software

HARDWARE é toda a parte física da computação:

- Processador, memória, periféricos, etc.

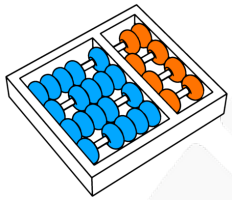
SOFTWARE é toda a parte lógica da computação:

- 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!

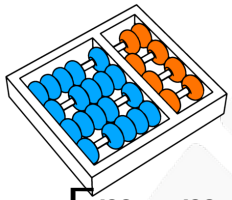


Ordens de magnitude

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. . .

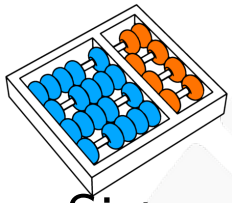
- Dizemos kibibyte, mebibyte, gibibyte, etc.
- Às vezes kiB é escrito como KB.
- Ainda há confusão se é múltiplo de 1000 ou 1024 por razões históricas.



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.



Organização

Sistema Operacional:

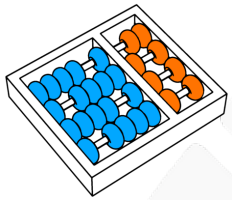
- Conjunto de programas que gerencia o Hardware.
- Ex: Linux, macOS, Windows, Android, iOS.
- Permite que outros programas usem recursos de maneira segura e organizada .

Aplicativos:

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

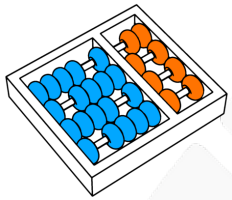
Compiladores/Interpretadores:

- 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.

Funcionamento de um computador

MC102 - Algoritmos e Programação de Computadores

03/24

1

Santiago Valdés Ravelo
https://ic.unicamp.br/~santiago/
ravelo@unicamp.br

UNICAMP