

SI100

Algoritmos e Programação de Computadores I

1º Semestre - 2018

Tópico 1: Introdução e Algoritmos

Unidades 1 e 2

Prof. Guilherme Palermo Coelho
guilherme@ft.unicamp.br

Roteiro

- Organização básica de um computador:
 - Processador;
 - Memória;
 - Periféricos;
- Algoritmos:
 - Definições;
 - Noções de correção e eficiência;
 - Formas de Representação;
- Exercícios;
- Referências.

ORGANIZAÇÃO DE UM COMPUTADOR

O que é um computador?

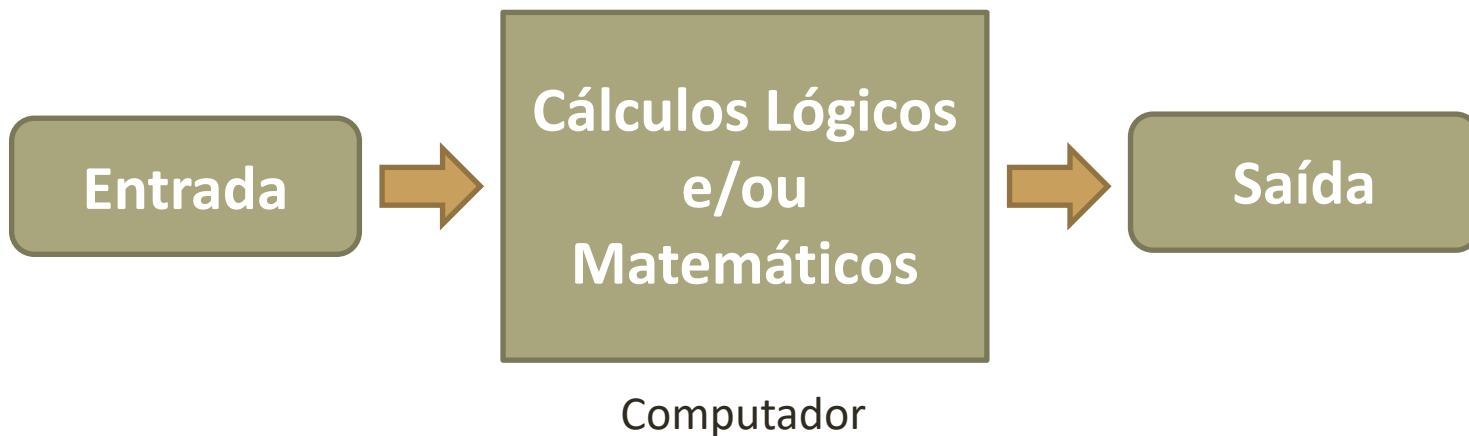
- Segundo o dicionário Merriam-Webster:
 - “*Dispositivo eletrônico programável para armazenamento, recuperação e processamento de dados*”;
- É uma máquina que executa um conjunto de instruções, fornecidas por um ser humano ou por outra máquina, com o intuito de desempenhar tarefas e resolver problemas;
 - Fornecer instruções para desempenhar uma tarefa → *programar o computador*;

O que é um computador?

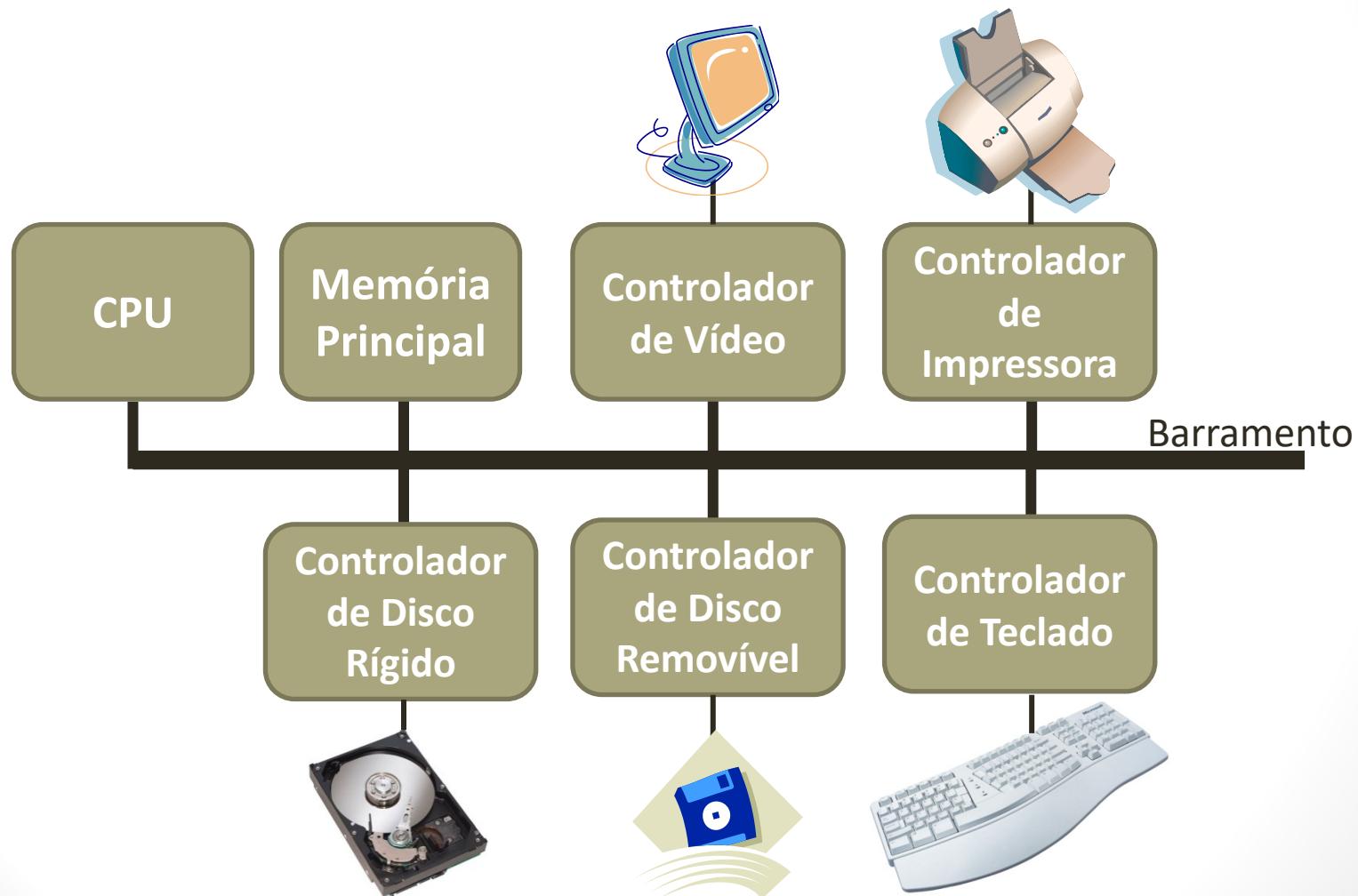
- Um computador pode realizar as mais diferentes tarefas:
 - Cálculos complexos:
 - Ex.: simulação aerodinâmica de veículos;
 - Geração de relatórios:
 - Ex.: movimentação bancária de um cliente;
 - Controle de outras máquinas:
 - Ex.: robôs;
 - Transmissão de informação:
 - Ex.: envio de e-mails, streaming de vídeos;

O que é um computador?

- De maneira simplificada:



Organização básica de um computador



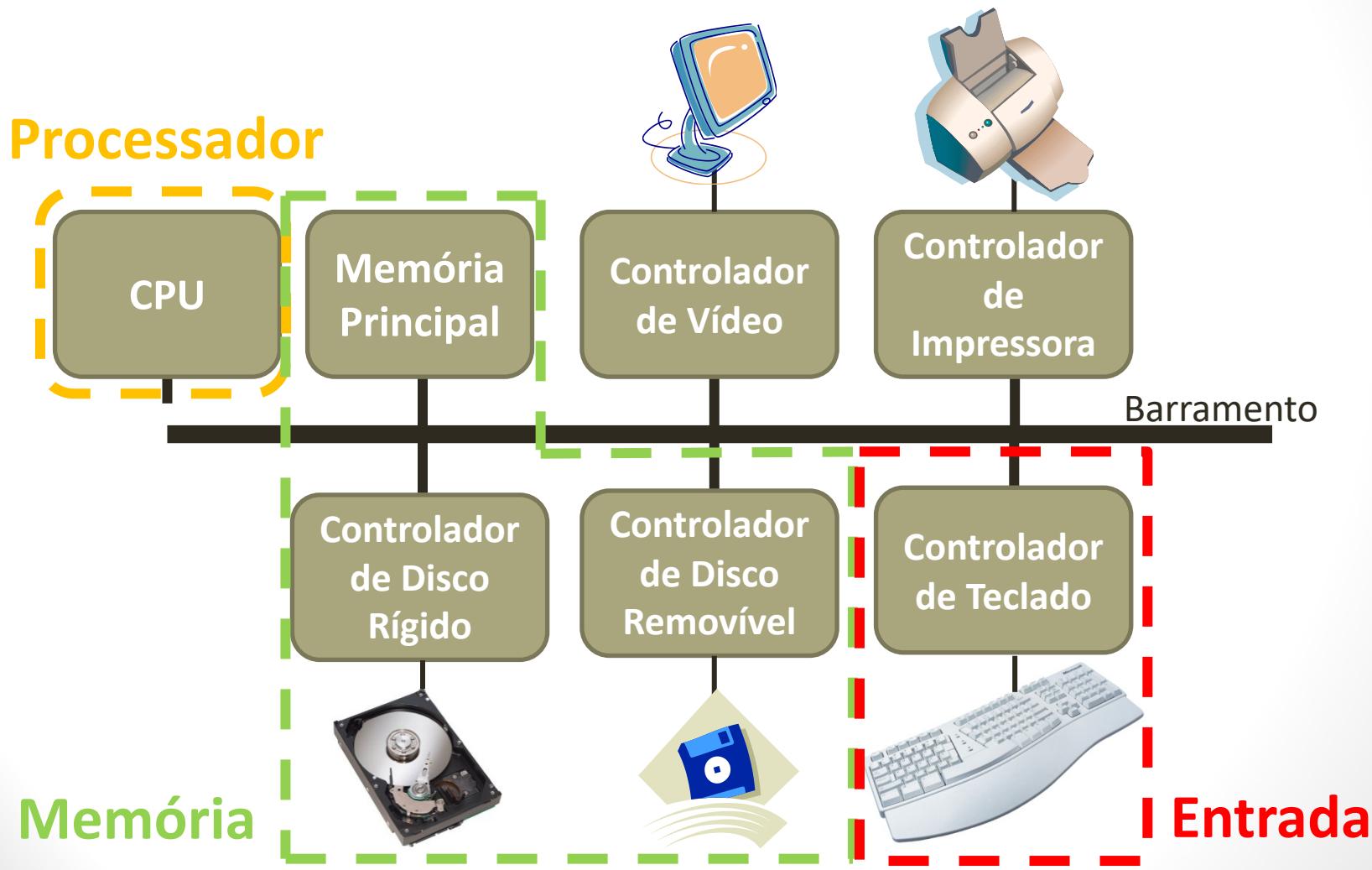
Organização básica de um computador



Organização básica de um computador



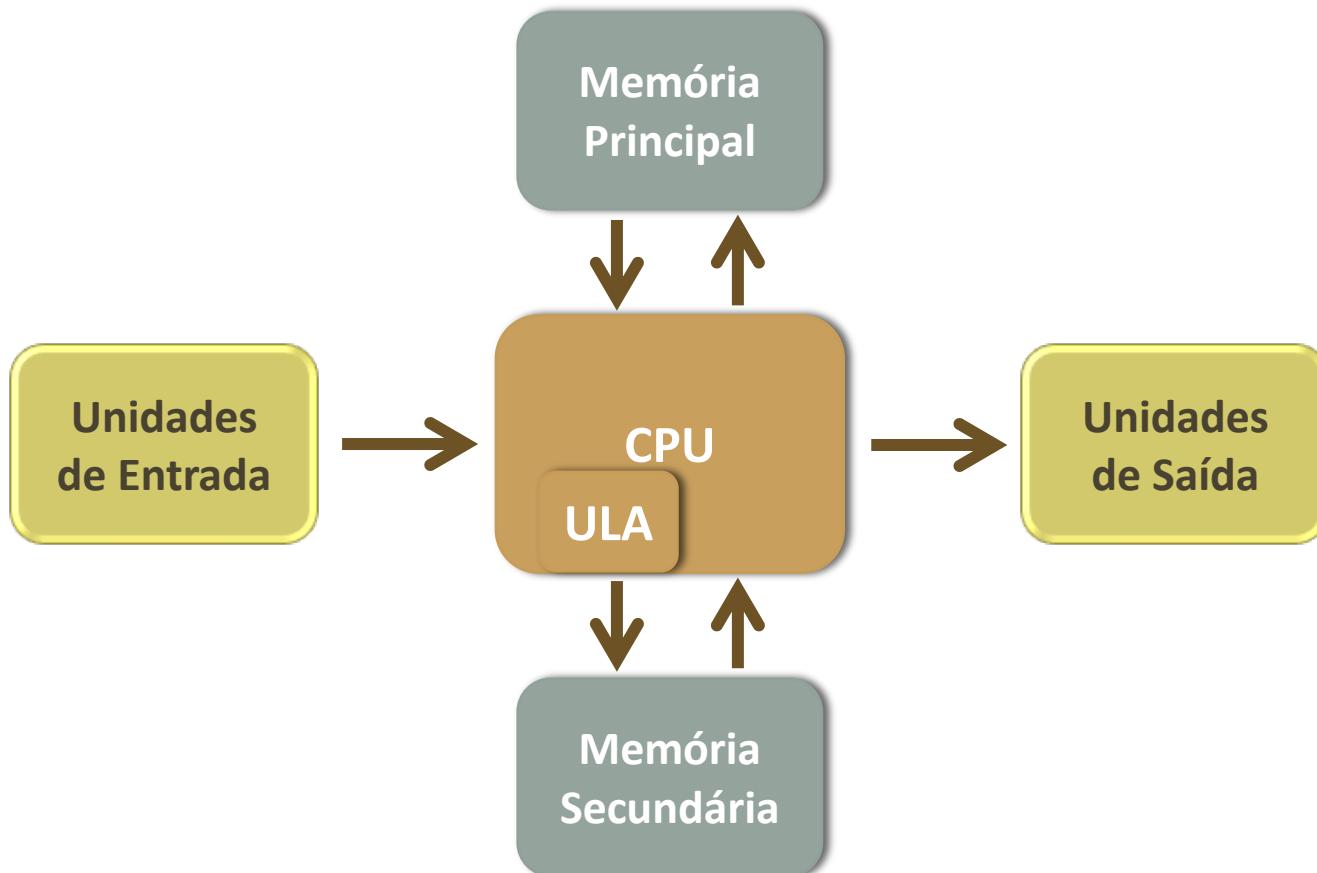
Organização básica de um computador



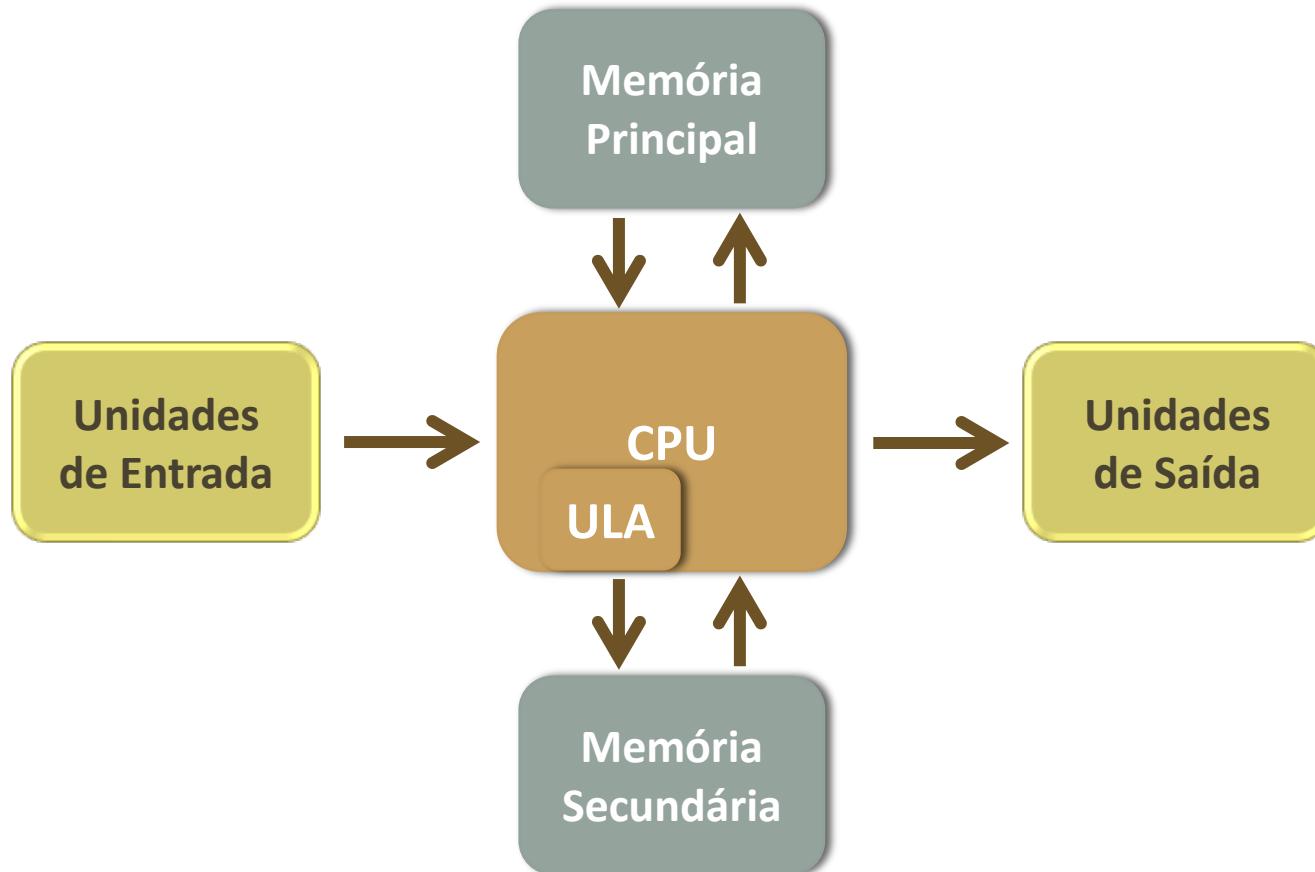
Organização básica de um computador



Organização básica de um computador

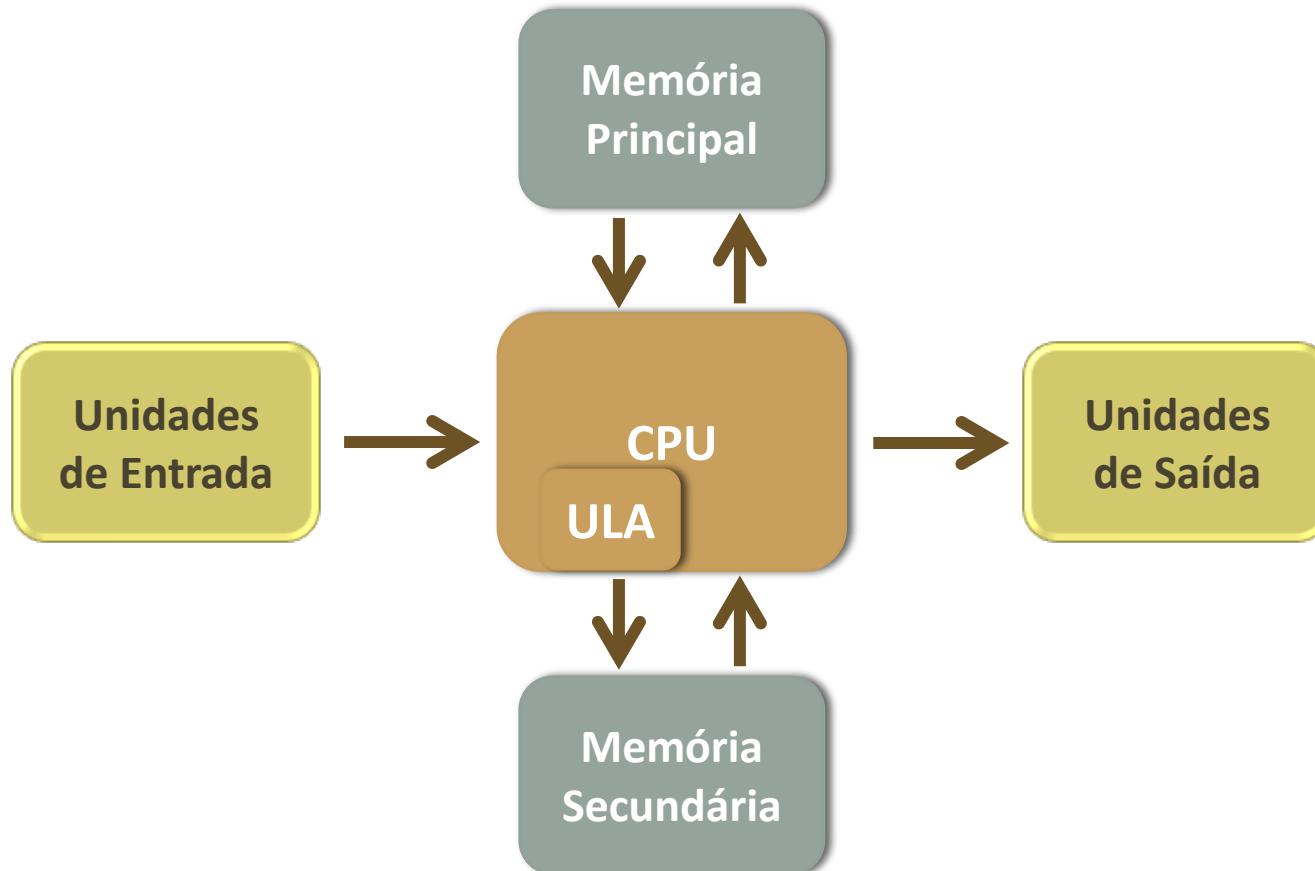


Organização básica de um computador



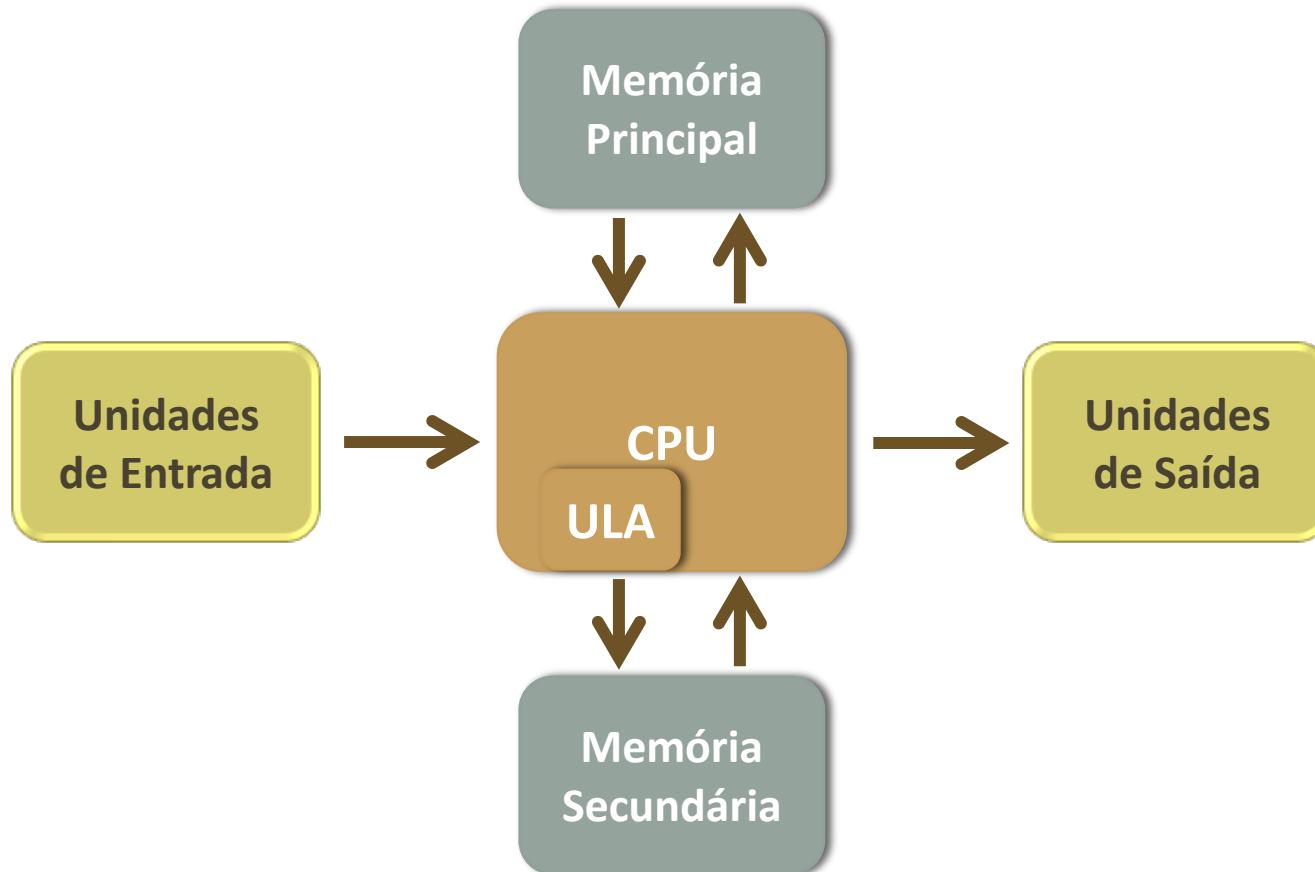
- **Unidade Central de Processamento (CPU – Central Processing Unit):** responsável pelo gerenciamento do sistema como um todo.

Organização básica de um computador



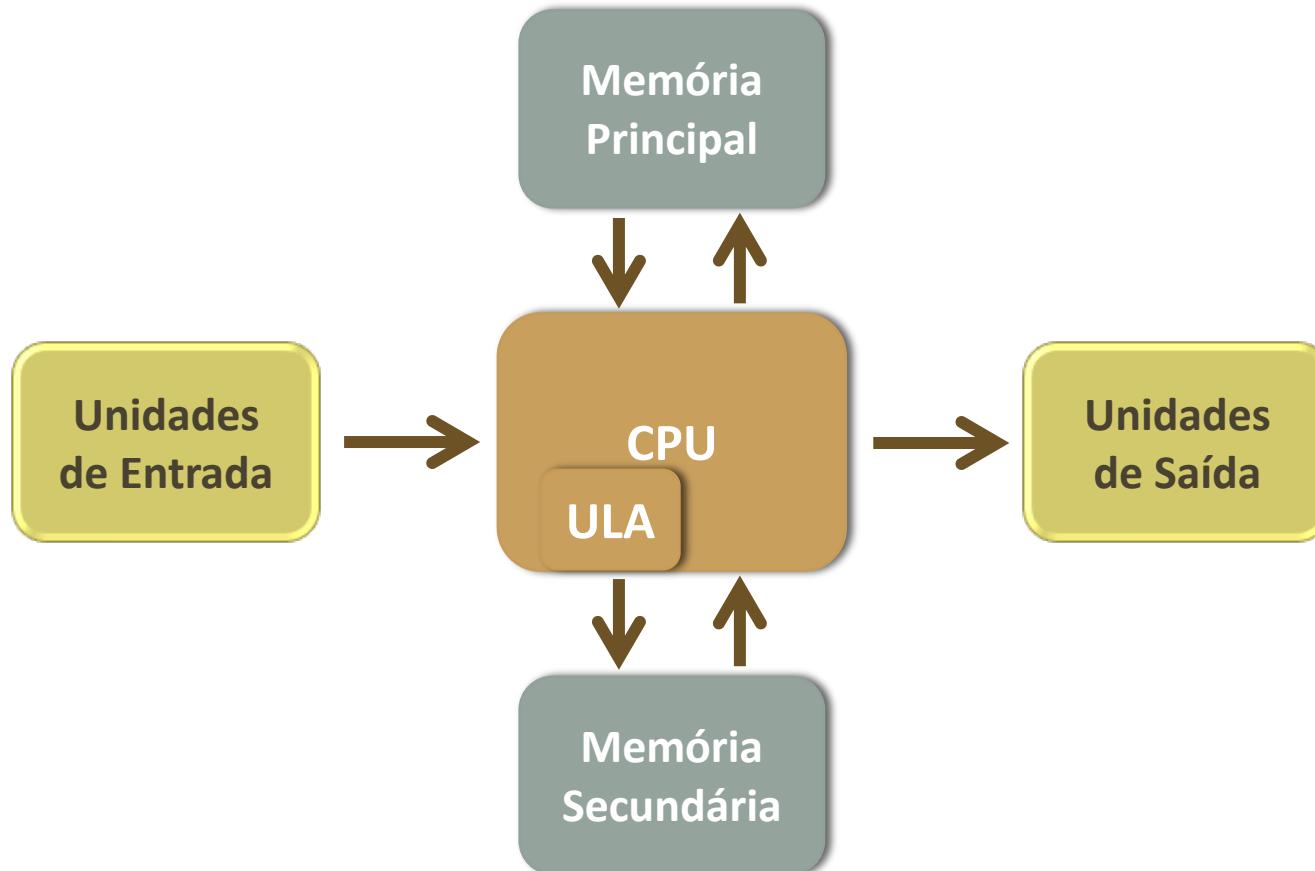
- **Unidade Lógica e Aritmética (ULA):** responsável por cálculos matemáticos. Muitos processadores englobam a ULA na CPU.

Organização básica de um computador



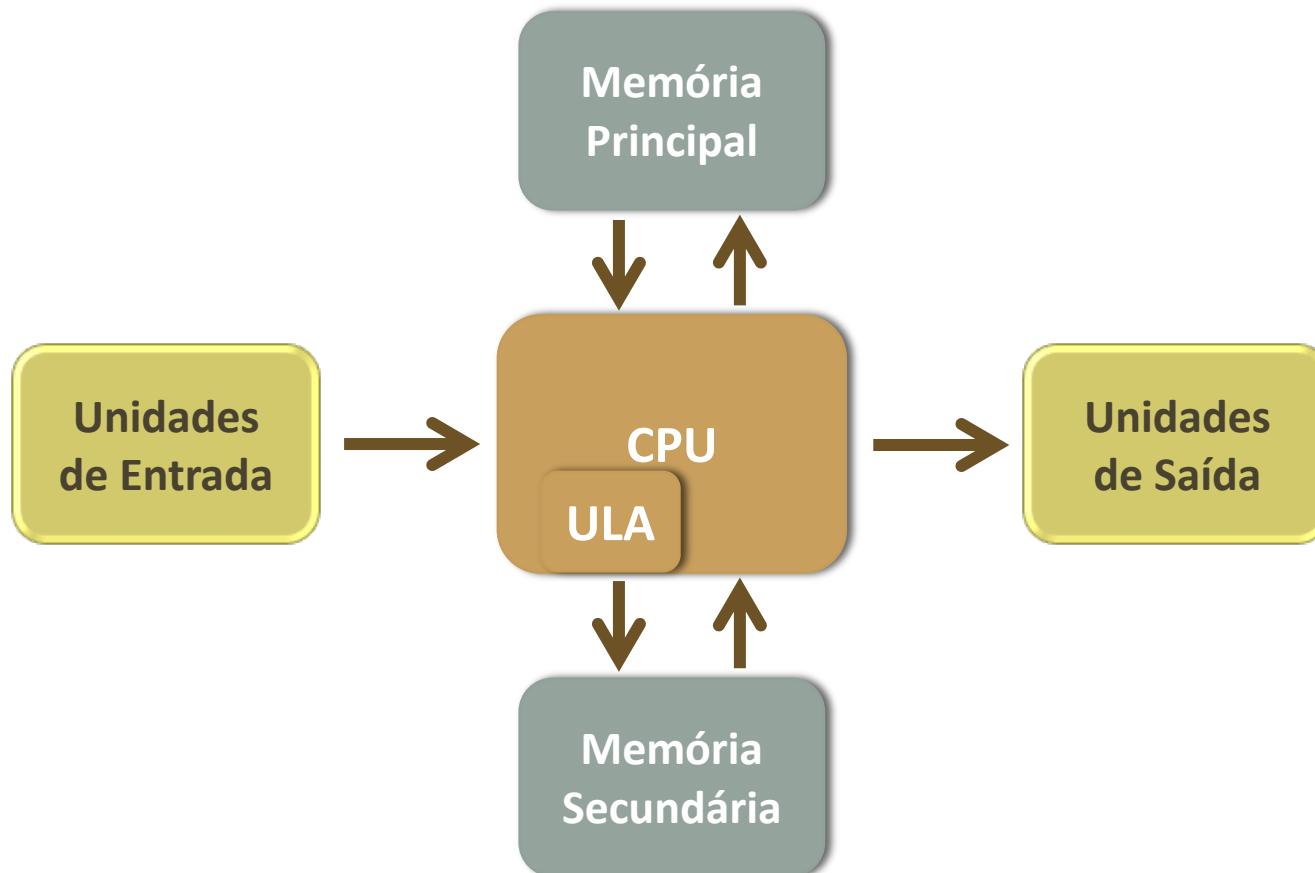
- **Unidades de Entrada:** permitem ao computador obter dados externos. Ex.: teclado, mouse, *webcam*, microfone...

Organização básica de um computador



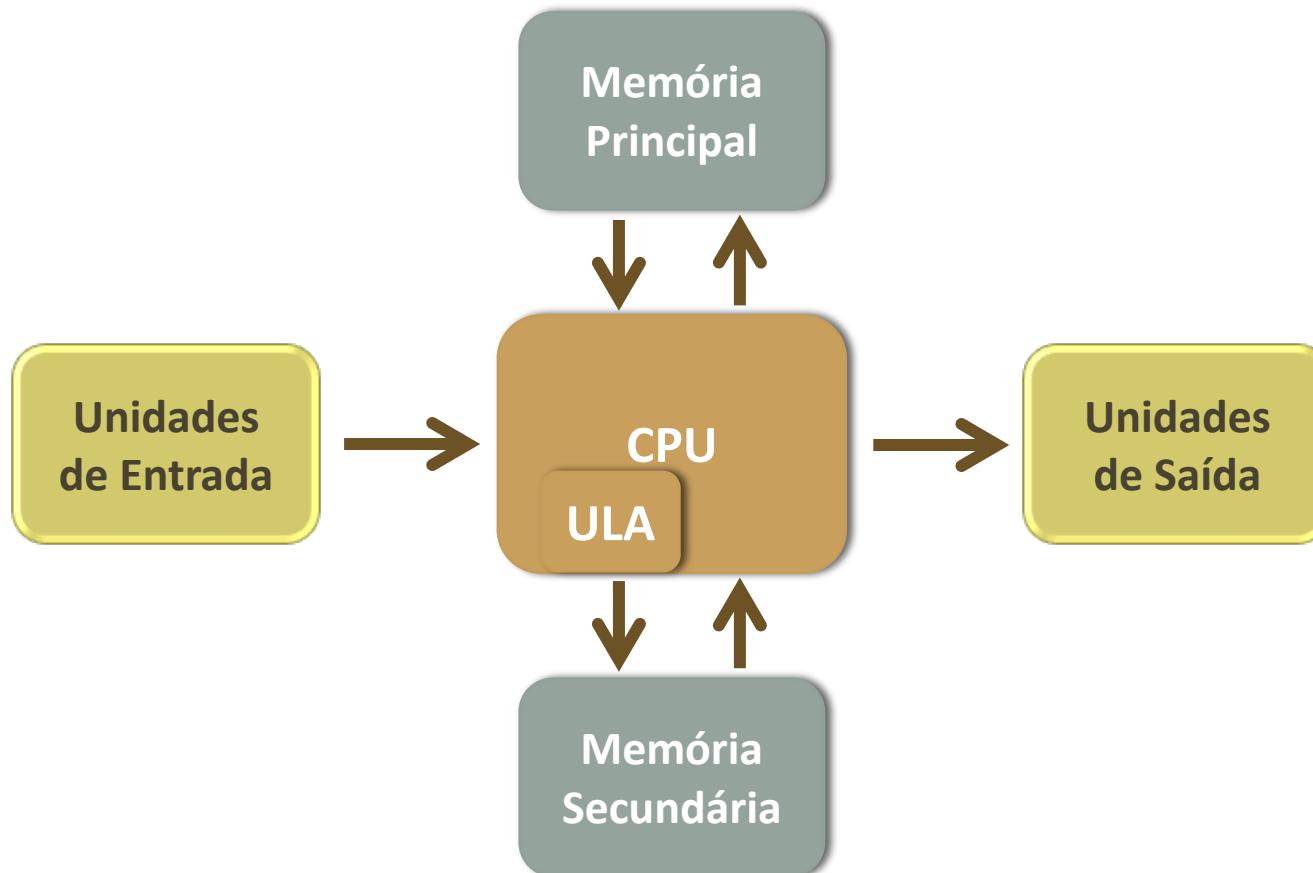
- **Unidades de Saída:** usadas para exibir os resultados da computação. Ex.: monitor, impressora...

Organização básica de um computador



- **Memória Principal** (RAM – *Random Access Memory*): usada para armazenar instruções e informações enquanto o computador está ligado.

Organização básica de um computador



- **Memória Secundária:** usada para armazenar instruções e informações por prazo indeterminado, independente de o computador estar ligado ou não. Ex.: HDs, DVD, pen-drives...

Armazenamento na Memória Principal

- **Memória de um computador:**
 - Só é capaz de armazenar ***bits*** (sequências de 0s e 1s);
 - Cada bit pode armazenar dois valores: 0 ou 1;
 - 0 = desligado ou desativado;
 - 1 = ligado ou ativado;
 - É dividida em unidades de armazenamento chamadas ***bytes***;
 - Cada byte é composto por 8 bits;
 - Cada posição de memória (byte) possui um *endereço próprio*;

Armazenamento na Memória Principal

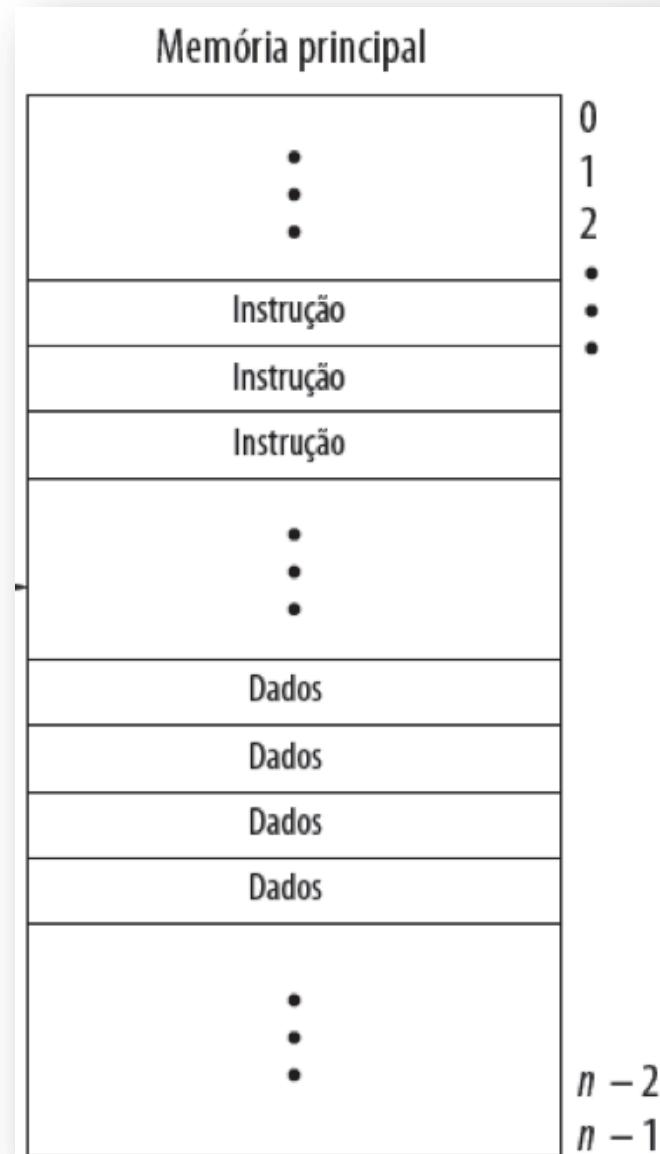
- Nada além de zeros e uns pode ser armazenado em memória:
 - Programas, textos, imagens, vídeos etc. são armazenados com o uso de uma **codificação numérica** na forma binária.
- Se só podemos armazenar números, como armazenar, por exemplo, um texto?
 - Cada caractere (inclusive os de pontuação e formatação) recebe um código numérico único:
 - Ex.: A = 65, B = 66, ...
 - Sequência de caracteres (texto) é convertida em uma sequência de números → pode ser armazenada em memória.

Armazenamento na Memória Principal

- A mesma estratégia pode ser usada para representar *programas* na memória:
 - Cada ***programa executável***: sequência de ***instruções*** que o processador interpreta.
 - Cada ***instrução***: representada por uma sequência de códigos numéricos (***binários***).
- Programas (sequência de instruções) são armazenados em memória secundária mas, para serem executados, devem ser ***transferidos para memória principal***.

Armazenamento na Memória Principal

- Ilustração de dados e programas armazenados em um módulo de memória principal com n posições:



Definição de termos

- **Dados:** qualquer tipo de informação ou instrução que possa ser manipulada por um computador;
- **Bit:** unidade básica para armazenamento, processamento e comunicação de dados;
- **Byte:** conjunto de 8 bits;
- **Palavra (*word*):** conjunto de n bytes;
- **Comandos:** instruções que levam o computador a executar tarefas;
- **Programa:** sequência de instruções com alguma finalidade;

Definição de termos

- **Arquivo:** conjunto de bytes que contém dados. Ex.: programa, imagem, música, vídeo, lista de caracteres, ...
- **Software:** conjunto de programas com um propósito em comum;
- **Hardware:** módulos físicos do computador;
- **Sistema Operacional:** conjunto de programas que gerencia e aloca recursos de *hardware* e *software*;

ALGORITMOS

Etapas para desenvolvimento de um programa

- O desenvolvimento de um programa de computador obrigatoriamente deve passar por *três etapas*:
 - **Análise**: nesta etapa estuda-se o **enunciado** do problema que se quer resolver, para que sejam definidos:
 - Os dados de entrada;
 - O processamento que deverá ser feito;
 - Os dados de saída;
 - **Elaboração do algoritmo**: definição da sequência de passos que deverá ser realizada para que o objetivo seja atingido;
 - **Codificação**: consiste na *transformação* do algoritmo em códigos de uma *linguagem de programação* escolhida.

Algoritmos

- Correspondem às “*receitas*” para se resolver um problema ou desempenhar uma tarefa;
- Ou seja, um algoritmo é a **sequência de passos** que se deve adotar para resolver um determinado problema;
- Podem ser especificados de diversas formas, inclusive em linguagem natural (português, inglês, etc.).



- Numa tigela misture bem a farinha, o açúcar, o cacau em pó, o fermento e o sal.
- Adicione o café expresso quente, o óleo, os ovos, a essência de baunilha e o iogurte líquido.
- ...

Algoritmos

- **Exemplo 1:** troca de uma lâmpada;
 - 1) Buscar uma escada;
 - 2) Posicionar a escada embaixo da lâmpada;
 - 3) Pegar uma lâmpada nova;
 - 4) Subir na escada;
 - 5) Retirar a lâmpada velha;
 - 6) Colocar a lâmpada nova;
 - 7) Descer da escada;
 - 8) Guardar a escada.



Este algoritmo é **linear** e **sequencial**!

Algoritmos – Exercício 01

Problema: Transportar a galinha, o milho e a raposa

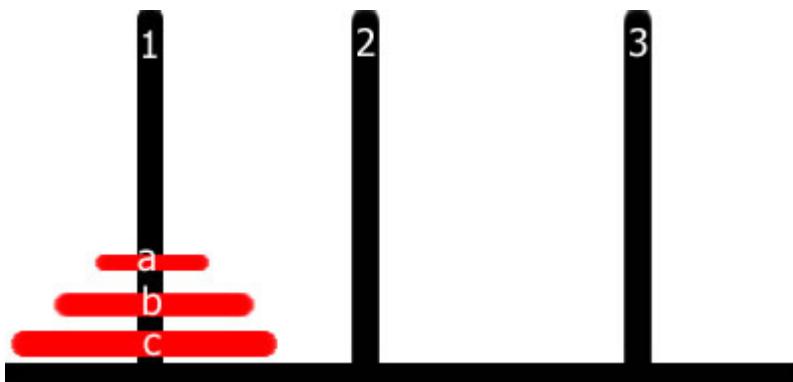
- Você tem um barco e precisa transportar uma galinha, uma raposa e uma saca de milho para outra margem de um rio. Para isto, as seguintes restrições se aplicam:
 - Em seu barco, só é possível transportar um item por vez;
 - Se a galinha ficar sozinha com o milho, ela comerá o milho;
 - Se a raposa ficar sozinha com a galinha, ela comerá a galinha;
 - Todos devem chegar inteiros do outro lado do rio.



Algoritmos – Exercício 02

Problema: Torre de Hanoi

- Suponha que você possua três pinos **1**, **2** e **3**, sendo que no pino **1** estão três discos de tamanhos diferentes, em ordem crescente de tamanho (vide figura). O objetivo é levar os três discos para o pino **3**, usando o pino **2** como auxiliar e obedecendo as seguintes restrições:
 - Pode-se mover um disco por vez;
 - Não se pode colocar um disco maior sobre um menor.



Algoritmos

- Nem todos os algoritmos são lineares e sequenciais;
- Se voltarmos no exemplo da troca da lâmpada (slide 28), percebemos que **sempre** trocaremos a lâmpada, independentemente de ela estar queimada ou não;
 - Isto seria um desperdício de lâmpadas;
 - Podemos adicionar um **teste de verificação**, para indicar se a lâmpada está queimada ou não;
 - A troca da lâmpada só será efetivamente realizada se o teste retornar **verdadeiro** para lâmpada queimada.

Algoritmos

- **Exemplo 2:** troca de uma lâmpada com verificação;

- 1) Buscar uma escada;
- 2) Posicionar a escada embaixo da lâmpada;
- 3) Pegar uma lâmpada nova;
- 4) Acionar o interruptor;
- 5) Se a lâmpada **não acender**, então
 - 6) Subir na escada;
 - 7) Retirar a lâmpada velha;
 - 8) Colocar a lâmpada nova;



Note que os passos 6, 7 e 8 só serão executados se a lâmpada não acender (condição 5 for **verdadeira**).

Algoritmos

- A solução apresentada no Exemplo 2 (slide 32), apesar de adequada, não prevê a possibilidade de que a nova lâmpada **pode não acender**;
 - Podemos acabar trocando uma lâmpada queimada por outra também queimada!
- Podemos refinar o algoritmo do Exemplo 2 de forma que, enquanto não colocarmos uma lâmpada que funcione, devemos repetir a operação de troca;

Algoritmos

- **Exemplo 3:** troca de uma lâmpada com verificação e repetição;
 - 1) Buscar uma escada;
 - 2) Posicionar a escada embaixo da lâmpada;
 - 3) Acionar o interruptor;
 - 4) Enquanto a lâmpada **não acender**, faça
 - 5) Pegar uma lâmpada nova;
 - 6) Subir na escada;
 - 7) Retirar a lâmpada velha;
 - 8) Colocar a lâmpada nova;
 - 9) Descer da escada;



Algoritmos

- No Exemplo 3 (slide 34), temos uma **estrutura de repetição** combinada com uma **estrutura de teste condicional**;
 - **Teste condicional**: verificar se a lâmpada acendeu;
 - **Estrutura de repetição**: repetir os passos de troca de lâmpada até que o teste condicional seja verdadeiro.
- O teste condicional é a **condição de parada** das repetições do algoritmo;
 - As repetições são encerradas no instante em que uma lâmpada que não esteja queimada é colocada no soquete.

Algoritmos

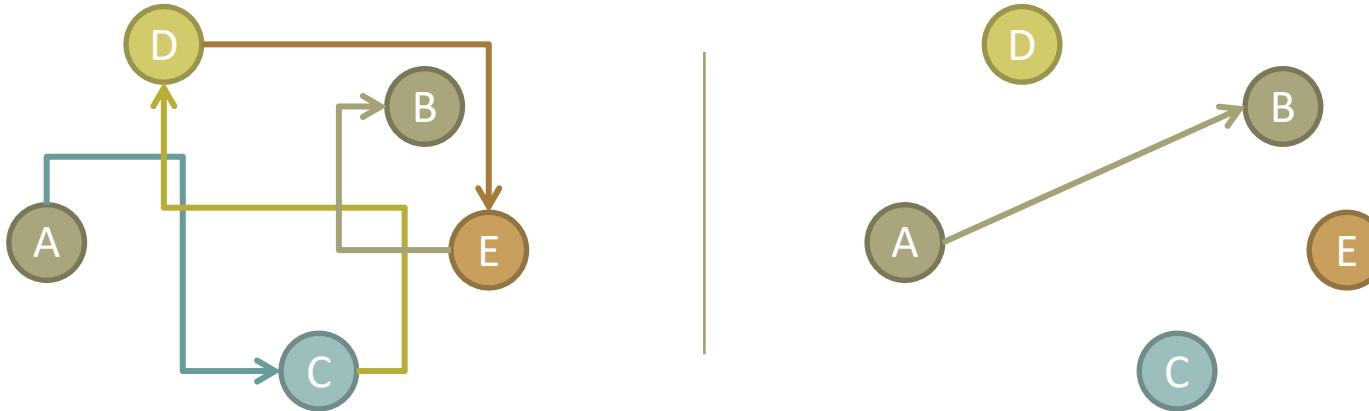
- Nestes exemplos de troca de lâmpadas → qualquer pessoa pode usar sua experiência e trocar uma lâmpada sem a necessidade de definir previamente um algoritmo para isto;
- No entanto, computadores não são pessoas!
 - Não têm conhecimento prévio;
 - Não adquirem experiência;
- Sendo assim, sempre que for necessário elaborar um programa computacional, é **essencial definir *a priori*** um algoritmo detalhado para o problema:
 - Detalhando ações a serem tomadas e prevendo obstáculos que poderão ser enfrentados (além de como transpô-los);
 - Ou seja, deve-se descrever uma **sequência finita de passos que garantam a solução do problema.**

Algoritmos: Corretude e Eficiência

- **Definição computacional:** um algoritmo é um procedimento computacional que recebe valores de entrada e produz valores de saída.
- **Algoritmo correto:** sempre termina e, para qualquer instância de entrada, produz uma saída correta.
- **Exemplo:** ordenação de uma sequência de números (vetor)
 - **Entrada:** uma sequência de números;
 - **Saída:** uma sequência ordenada dos mesmos números de entrada.
 - Se a entrada for: (32, 55, 1, 199);
 - A saída deverá ser: (1, 32, 55, 199).

Algoritmos: Corretude e Eficiência

- **Algoritmo eficiente:**
 - Ex.: ir do ponto A ao ponto B.



- Existem diferentes algoritmos para resolver um mesmo problema;
- Diferentes algoritmos podem ter características diferentes: tempo computacional, uso de memória, número de acessos a disco, etc.

Algoritmos: Corretude e Eficiência

- Voltando ao Exemplo 2 (reproduzido abaixo), note que o algoritmo proposto não é muito eficiente;
- **Exemplo 2:**
 - 1) Buscar uma escada;
 - 2) Posicionar a escada embaixo da lâmpada;
 - 3) Pegar uma lâmpada nova;
 - 4) Acionar o interruptor;
 - 5) Se a lâmpada **não acender**, então
 - 6) Subir na escada;
 - 7) Retirar a lâmpada velha;
 - 8) Colocar a lâmpada nova;



Algoritmos: Corretude e Eficiência

- É possível reescrevê-lo tornando-o mais eficiente;

- **Exemplo 4:**

- 1) Acionar o interruptor;
- 2) Se a lâmpada **não acender**, então
 - 3) Buscar uma escada;
 - 4) Posicionar a escada embaixo da lâmpada;
 - 5) Pegar uma lâmpada nova;
 - 6) Subir na escada;
 - 7) Retirar a lâmpada velha;
 - 8) Colocar a lâmpada nova;



Se a lâmpada não estiver queimada, nenhuma ação é tomada.

Algoritmos: Forma de Representação

- É importante destacar mais uma vez que **um algoritmo é uma linha de raciocínio**;
- Pode ser representado de diversas formas:
 - Até o momento, usamos a forma **textual**, escrevendo cada algoritmo como um **pseudocódigo em português coloquial**;
 - Está sujeita a **ambiguidade**!
 - “Sentado na varanda, o menino avistou um mendigo.”
 - “A polícia cercou o ladrão do banco na rua Santos.”
 - “Onde está a cachorra da sua mãe?”

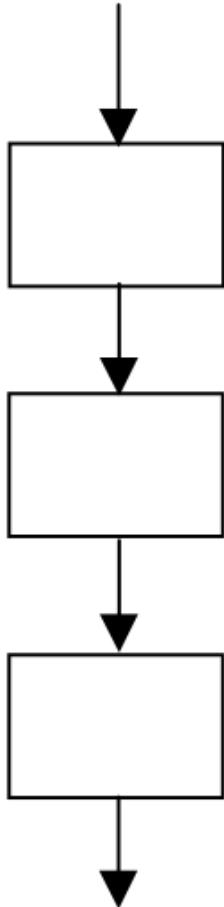
Algoritmos: Forma de Representação

- É importante destacar mais uma vez que **um algoritmo é uma linha de raciocínio**;
- Pode ser representado de diversas formas:
 - Também é possível representar algoritmos **graficamente** (em diagramas);
 - A representação gráfica de algoritmos tende a ser mais **fiel à linha de raciocínio** e também mais **clara** que a representação textual;
 - Por outro lado, é necessário conhecer as convenções de cada representação.

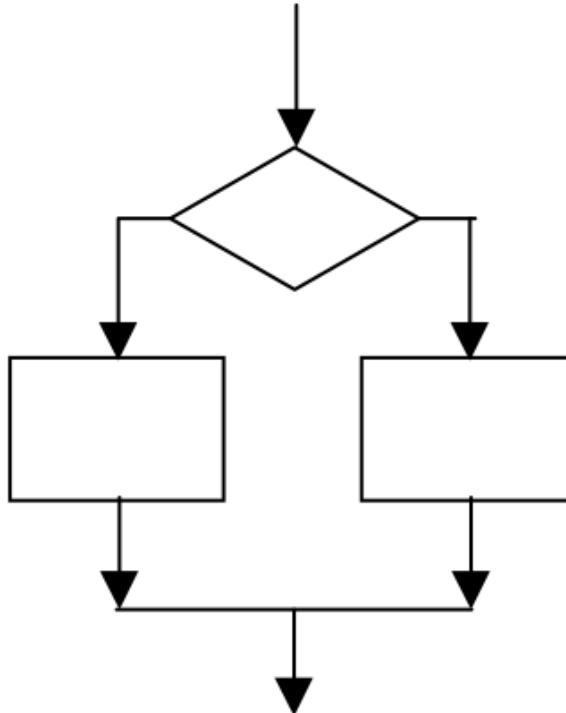
Algoritmos: Fluxogramas

- É a forma mais comum de representação gráfica de algoritmos;
- Notação:
 - **Retângulo**: representa um passo ou um módulo do algoritmo;
 - **Seta**: indica o próximo comando (passo) a ser executado;
 - **Losango**: indica uma condição que interfere no fluxo do algoritmo ou programa;
 - **Retângulo arredondado**: indica início e fim da execução.

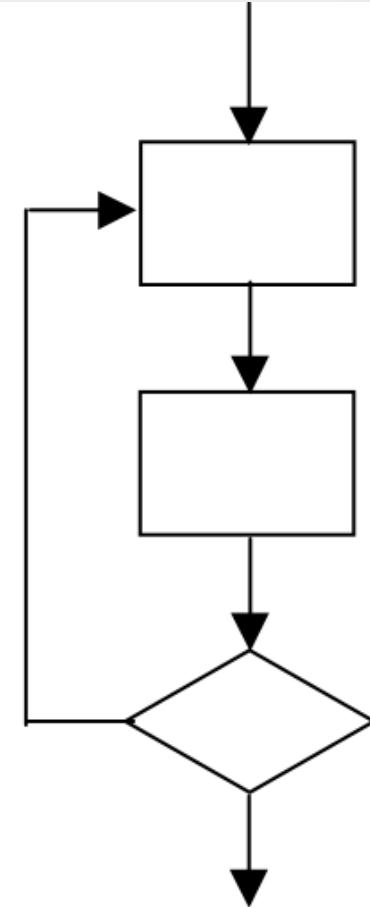
Algoritmos: Fluxogramas



ESTRUTURA DE
CONTROLE SEQÜENCIAL



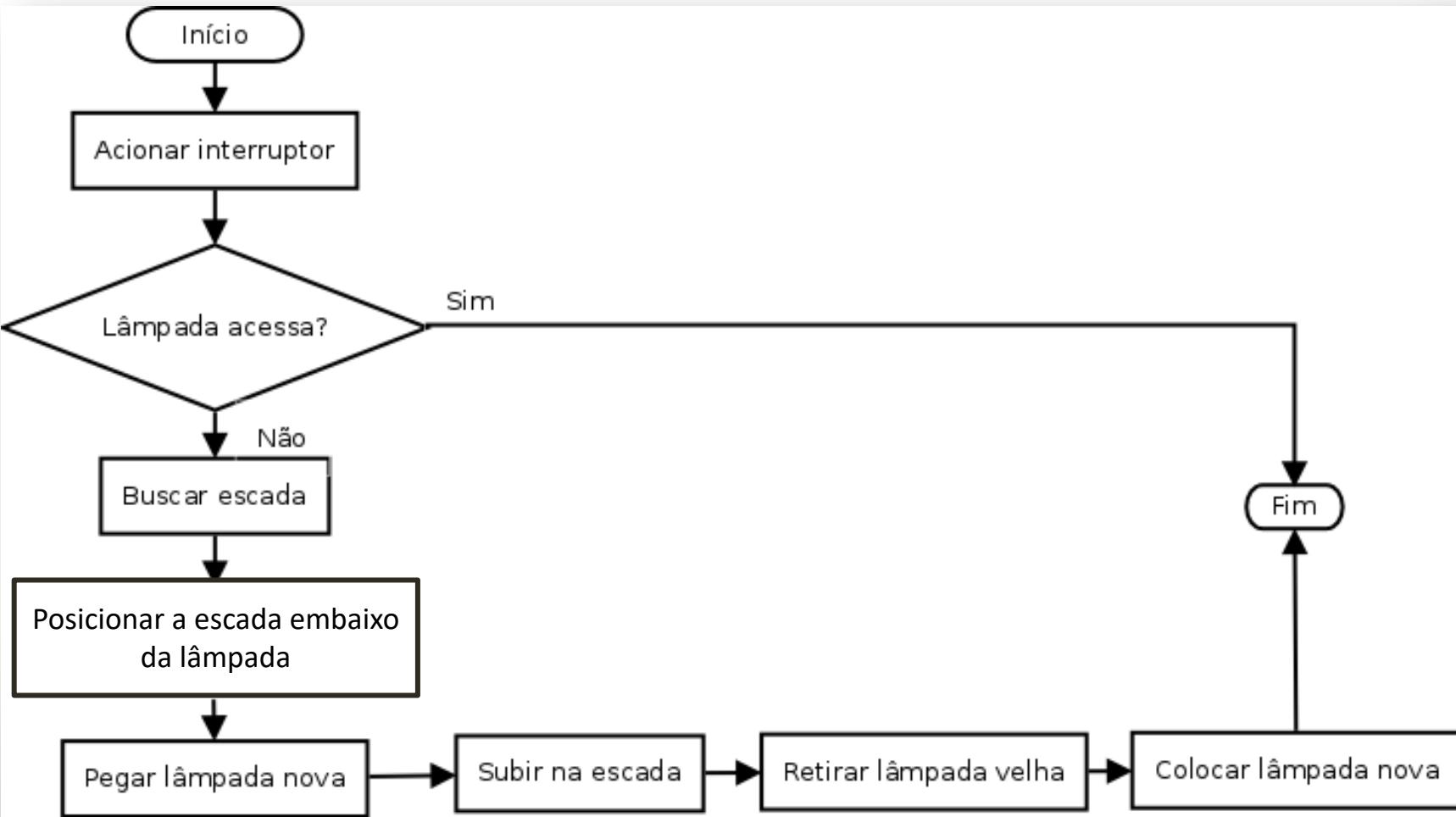
ESTRUTURA DE
CONTROLE CONDICIONAL



ESTRUTURA DE
CONTROLE REPETITIVA

Algoritmos: Fluxogramas

- Voltando ao Exemplo 4 (slide 40):



EXERCÍCIOS

Exercícios

1. Tendo como base os algoritmos para troca de uma lâmpada vistos em aula, escreva um algoritmo para resolver o problema de troca de lâmpadas em dez soquetes de um salão. Apresente seu algoritmo em pseudocódigo e em fluxograma.
2. Modifique o algoritmo do exercício 1 para que apenas as lâmpadas queimadas sejam substituídas.
3. Elabore um algoritmo para um programa computacional que compare dois números X e Y quaisquer. Este programa deve receber os valores de X e Y, compará-los e escrever uma frase indicando qual dos números é o maior e qual é o menor. Não se preocupe com detalhes computacionais, apenas com a lógica. Apresente o algoritmo na forma de pseudocódigo.

Exercícios

4. Modifique o algoritmo do exercício 3 para considerar também o caso em que X e Y sejam iguais. Apresente o algoritmo modificado em um fluxograma.
5. Modifique o algoritmo visto em aula (slide 30), para o problema da Torre de Hanói, de forma a mover **quatro** discos do pino 1 para o pino 3, usando o pino 2 como auxiliar.
6. Considere uma calculadora comum, de quatro operações, que esteja com as teclas de multiplicação e divisão inoperantes. Escreva algoritmos que resolvam os problemas abaixo usando **apenas** as operações de adição e subtração:
 - 12×4 ;
 - $10 \div 2$.

REFERÊNCIAS

Referências

- FORBELONE, A. L. V., EBERSPÄCHER, H. F., Lógica de Programação – A construção de algoritmos e estruturas de dados, 3a Edição, São Paulo, Pearson Prentice Hall, 2005.
- ASCENCIO, A. F. G., CAMPOS, E. A. V., Fundamentos da Programação de Computadores – Algoritmos, Pascal e C/C++, Pearson Prentice Hall, 2003.
- CELES, W., CERQUEIRA, R., RANGEL, J. L.. Introdução a Estruturas de Dados. Campus, 2004.