

Introdução à Computação

Aulas 1.2:

- *Conceitos Básicos*
- *Visão Geral do Computador*
- *Hardware e Software*

Prof. Carlos Antônio Campos Jorge



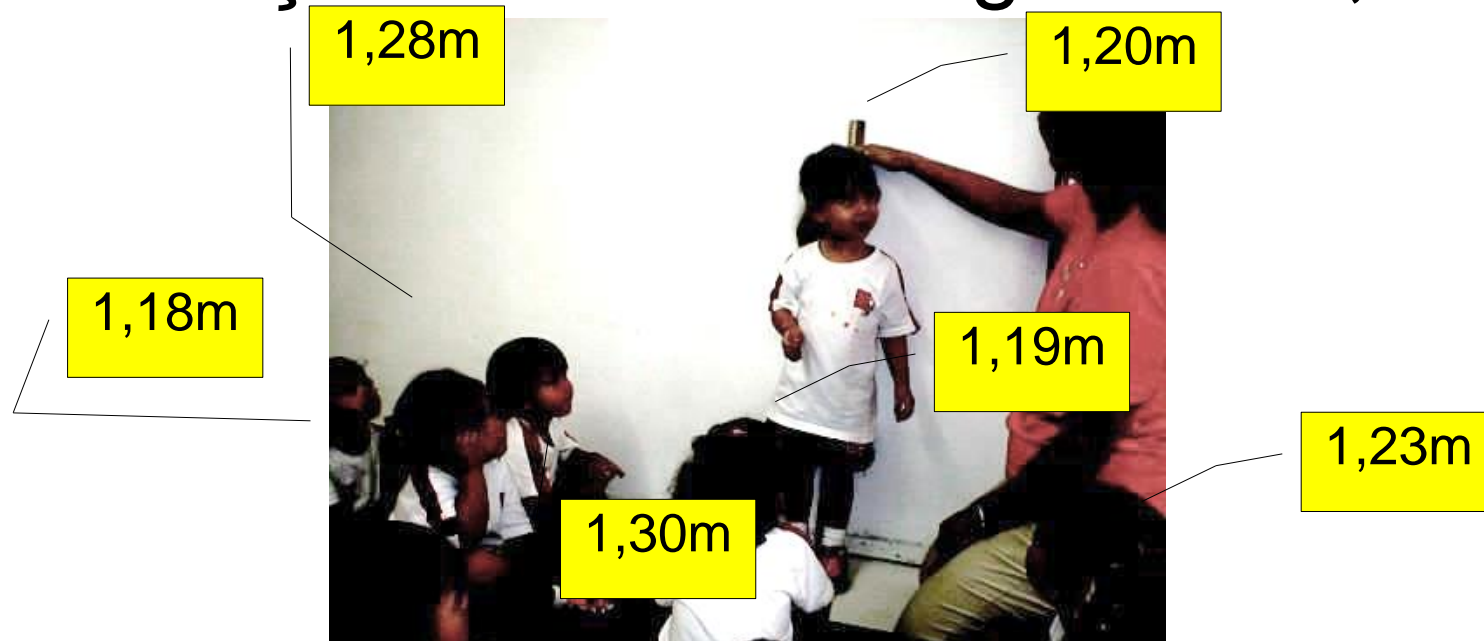
Conceitos básicos

- Dado: representação de uma idéia, evento. Fatos ou objetos não processados;
- Informação: dado com significado;



Introdução

- Dado: representação de uma idéia, evento. Fatos ou objetos não processados;
- Informação: dado com significado;



Introdução

- Dado: representação de uma idéia, evento. Fatos ou objetos não processados;
- Informação: dado com significado;

Altura
média:
1,23m



Processamento de Dados

- Série de **operações** que se aplica a um conjunto de dados (**entrada**) para obter informações ou resultados (**saída**).
- Elementos Básicos:
 - Dados **iniciais** - informações iniciais sujeitas a certas transformações.



Processamento de Dados

- **Transformações** (Processamento) - modificações efetuadas no conteúdo ou na forma dos dados iniciais.
 - Resultados **finais** - produto dos dados iniciais após as transformações.
- Exemplos:
 - **Dado Inicial:** Leite, Abacate e Açúcar.
 - **Transformação:** Mistura dos 3 ingredientes no liquidificador.
 - **Resultado Final:** Vitamina de abacate!



Processamento Eletrônico de Dados

- Processamento de dados com a utilização do **computador**.

COMPUTADOR



PROCESSAMENTO ELETRÔNICO DE DADOS

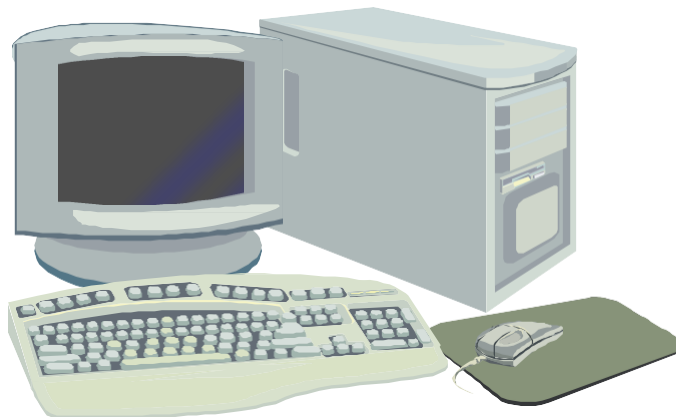


lê dados
processa dados
fornece resultados





O que é um *Computador* ?



O que é um Computador?

- **Digital:** dados codificados no sistema binário;
- **Computador:** aceita dados e os processa eletronicamente.



O que é um Computador?

- É uma máquina constituída por uma série de componentes e circuitos eletrônicos, capaz de receber, armazenar, processar e transmitir informações.
- **Máquina programável**, capaz de realizar uma grande variedade de tarefas, seguindo uma sequência de comandos, de acordo com o que for especificado.
- *“Computadores são estúpidos, eles somente respondem perguntas.” Pablo Picasso*



Processamento Eletrônico de Dados

- Vantagens:
 - Processa grande **volume** de dados com rapidez;
 - Trata grandes **quantidades** de informações com segurança;
 - Realiza cálculos com **exatidão**;
 - Oferece grande **disponibilidade** de acesso às informações armazenadas;
 - Pode ser **programado**;



Benefícios dos Computadores

- **Produtividade**

- Funcionários usam seus computadores para executar suas tarefas mais rápido e melhor;
- Muitos processos podem ser controlados mais eficientemente por meio dos computadores.

- **Tomada de decisões**

- Ajuda os tomadores de decisões a identificar fatores financeiros, geográficos e logísticos.

- **Redução de custos**

- Ajuda a reduzir os custos com mão-de-obra, energia e papelada.



De que é formado um computador?

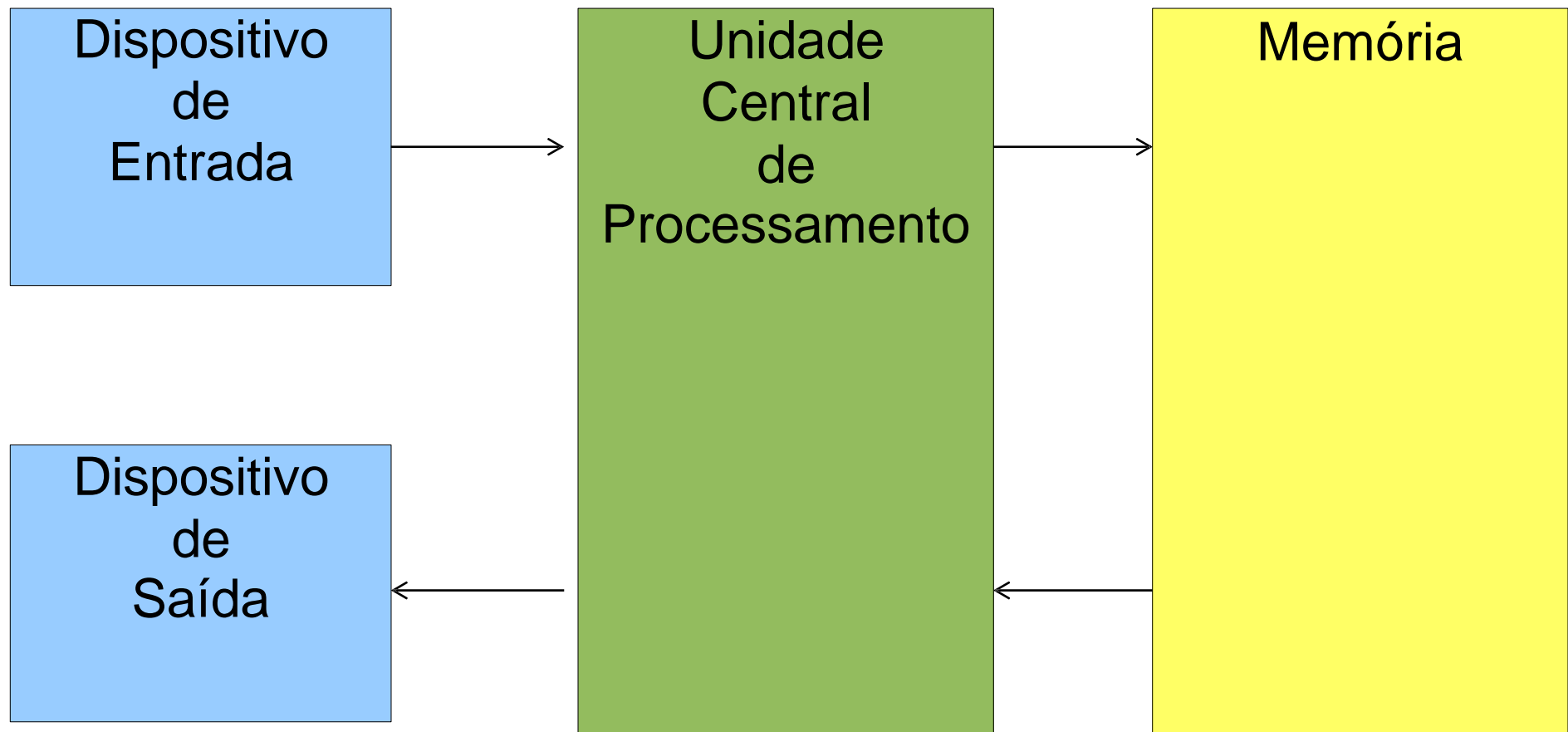


Computador

- O computador é um recurso formado por duas partes:
 - Parte **física**
 - Parte **lógica**
- A parte **física** é a que podemos ver e tocar e o **nome técnico** que se dá à essa parte é **HARDWARE**. Na parte lógica, não se pode tocar e ela é conhecida como **SOFTWARE**.
- $\text{COMPUTADOR} = \text{HARDWARE (física)} + \text{SOFTWARE (lógica)}$.



Hardware – Modelo Simplificado de um Computador



Dispositivos de Entrada

- Seu objetivo é obter dados que serão colocados na memória para que sejam posteriormente usados pelo processador em cálculos aritméticos ou lógicos.



Dispositivos de Saída

- Seu objetivo é obter dados da memória do computador e mostrá-los ou passá-los ao usuário.



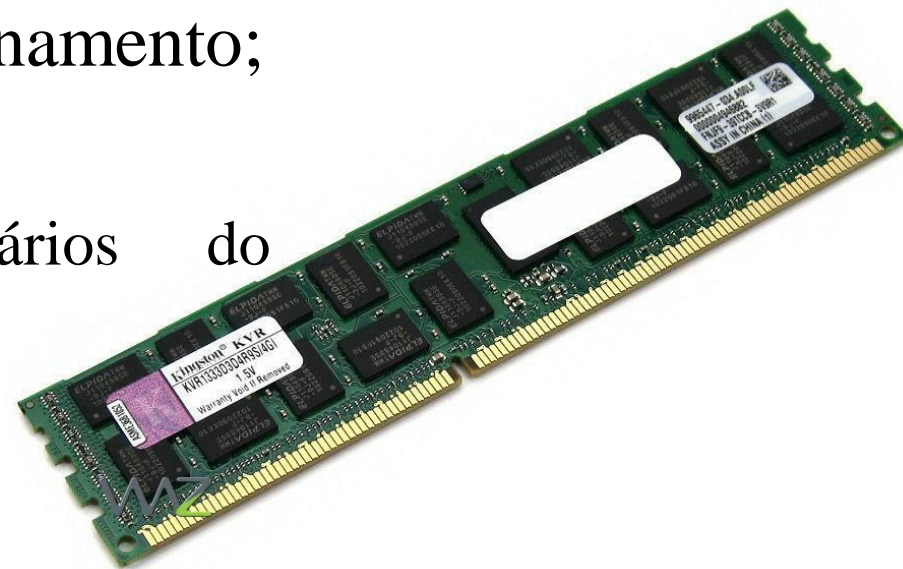
Memória

- Dados e informações são fornecidos como entrada e podem ser obtidos como saída.
- Mas também podem ser guardados (armazenados).
- Memória é onde são armazenados os dados.



Memória RAM

- RAM (*Random Access Memory*)
 - Memória temporária;
 - Utilizada para desenvolver e executar programas;
 - É volátil: Uso restringe-se ao período em que o equipamento está em funcionamento;
 - Armazenar programas e dados;
 - Guardar resultados intermediários do processamento.

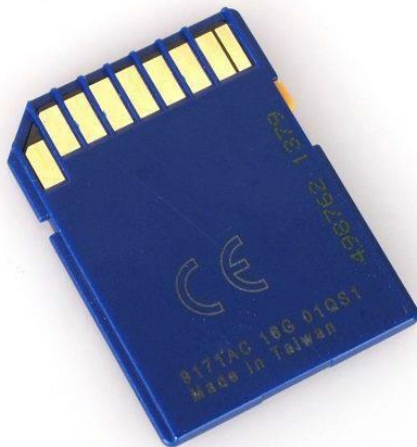
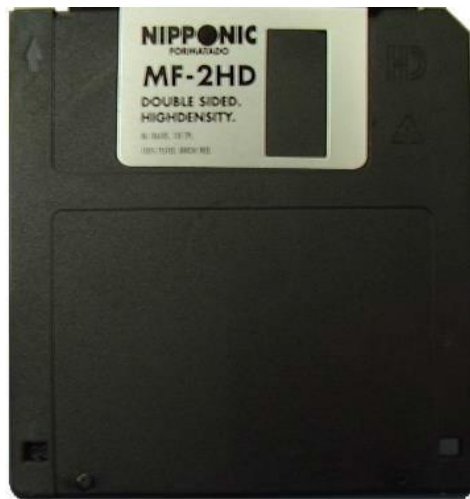


Memória ROM

- ROM (*Read Only Memory*)
 - Tipicamente menor que a RAM;
 - Não depende de energia para manter o seu conteúdo;
 - Memória permanente;
 - Informações não podem ser apagadas (casos especiais);
 - Geralmente vem gravada do fabricante;
 - Apenas de leitura;
 - Rotina de inicialização do computador, reconhecimento do hardware, identificação do sistema operacional, contagem de memória;
- Orienta o computador nas 1ªs operações.



Dispositivos de Armazenamento Externo

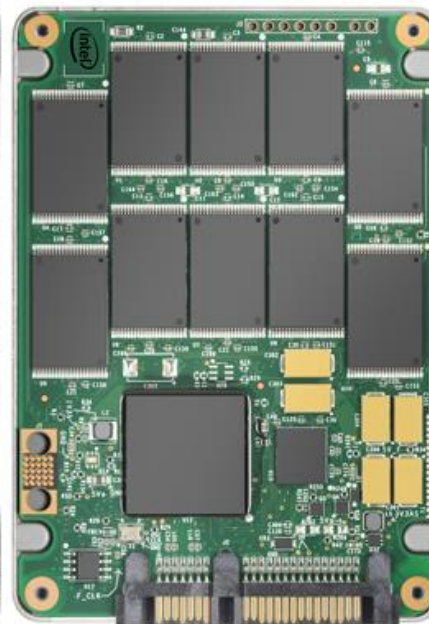


Dispositivos de Armazenamento Interno – HD/SSD

- O disco rígido ou HDD (Hard Disk) e estado sólido ou SSD (Solid State Disk) são usados para guardar não só seus arquivos como também todos os dados do seu sistema operacional, sem o qual você não conseguiria utilizar o computador.



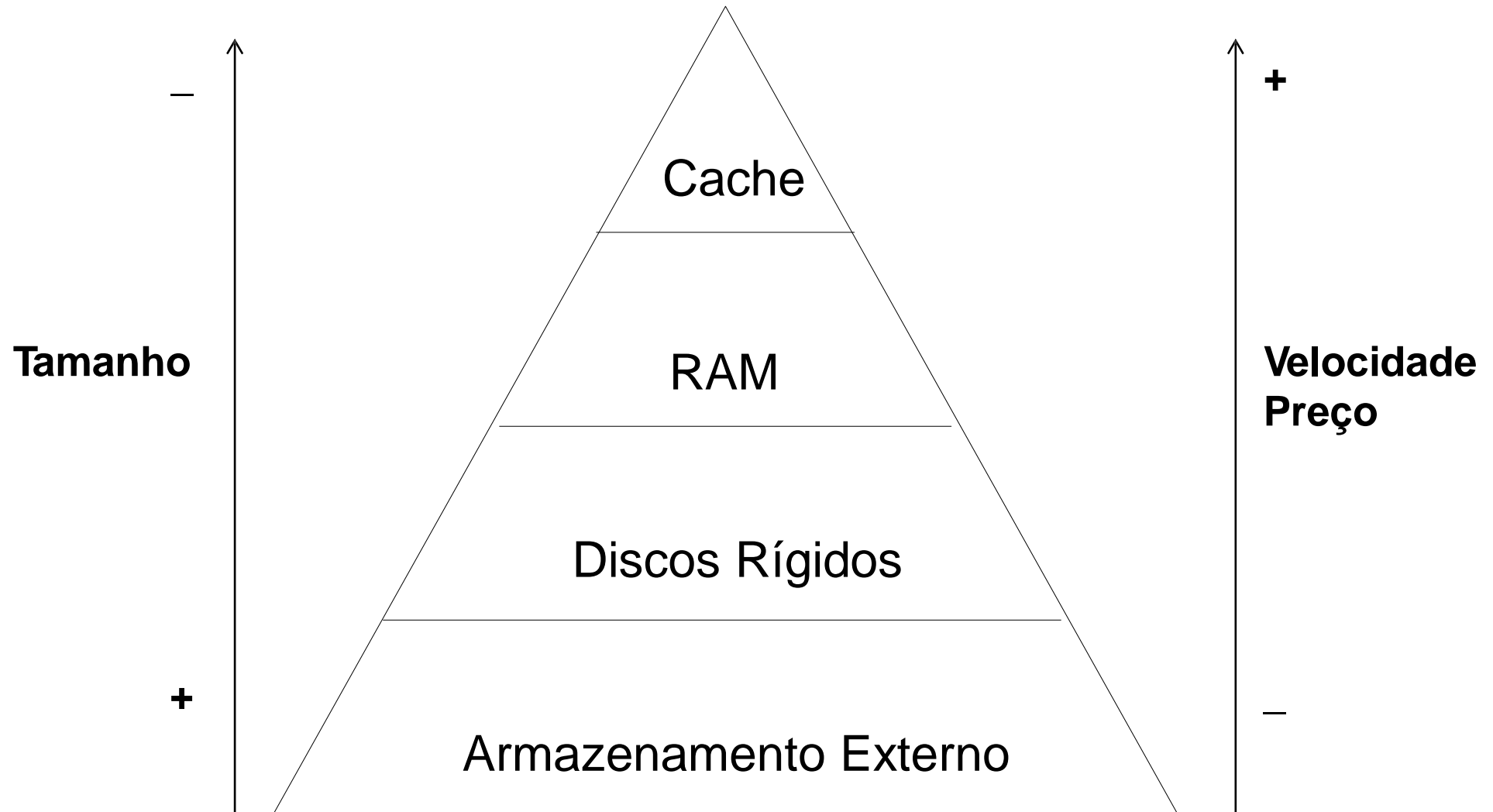
HDD



SSD



Hierarquia de memória



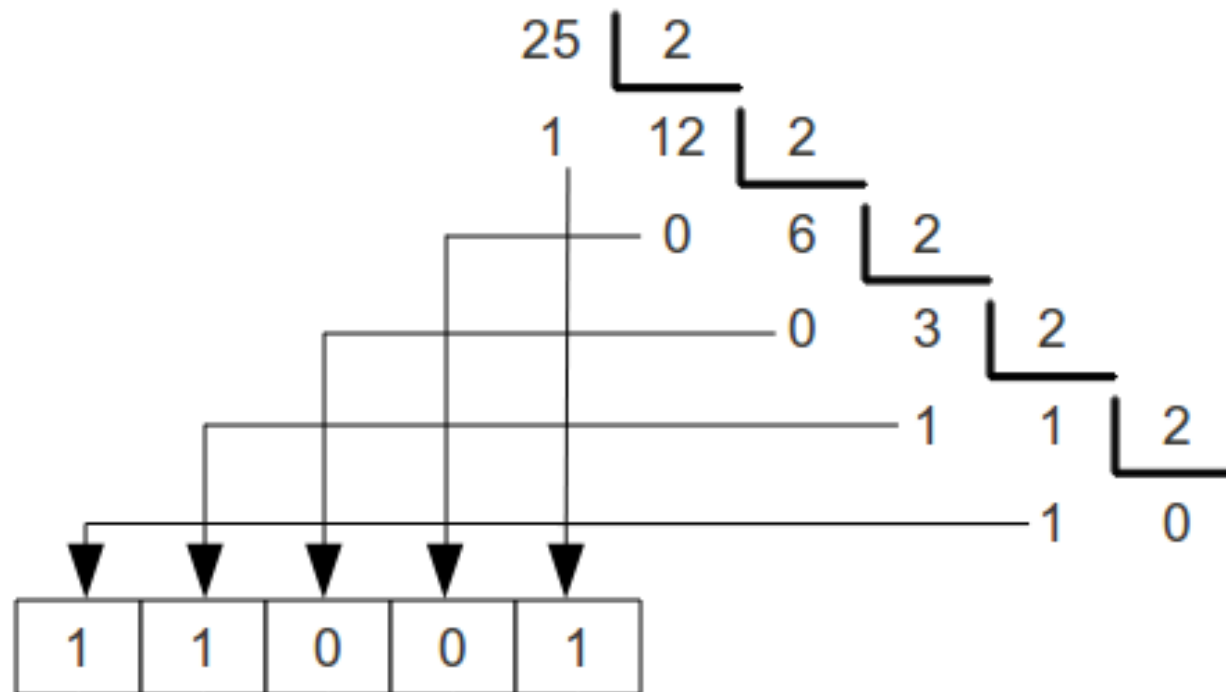
Armazenamento

- Nos computadores, os dados são armazenados na forma binária, isto é, como zeros e uns. Portanto, a memória é uma sequência de zeros e uns.
- BIT (Binary digIT): 0 ou 1.
- Esses bits estão divididos em grupos de 8, denominados bytes.
- Os bytes são divididos em grupos de 1, 2, 3, 4, ..., denominados de palavras. O tamanho da palavra depende, particularmente, do computador.
- Existem computadores com palavras de 8 bits, 16 bits, 24 bits, 32 bits, e 64 bits de palavra.
- 8 ou 16 bits: computadores para propósitos específicos.



Codificação

- Representar números
 - Converter da base 10 para a base 2 (binário).




Unidade de Armazenamento

Bit (b*)	1 unidade
Byte (B*)	8 bits
Kilobyte (KB)	1024 Byte
Megabyte (MB)	1024KB
Gigabyte (GB)	1024 MB
Terabyte (TB)	1024 GB
Petabyte (PB)	1024 TB
Exabyte (EB)	1024 PB
Zettabyte (ZB)	1024 EB
Yotabyte (YB)	1024 ZB



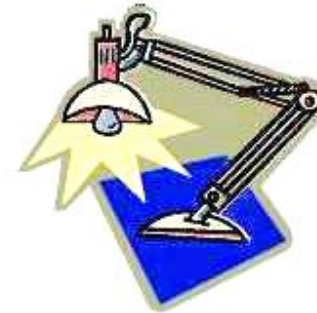
Unidade Central de Processamento

- CPU (Central Processing Unit) ou Processador é responsável pela execução de cálculos, decisões lógicas e instruções que resultam em todas as tarefas que um computador pode fazer;
 - Realiza o processamento;
 - Cérebro do computador.
- 



Computador

Energia



Lâmpada

Está para

Gasolina



Carro

Assim como ...



Programa

Está para



Computador



De onde vem o programa?



Mágica???



De onde vem o programa?



De onde vem o programa?



Pessoas



Programadores



Como programas são feitos?

- ♦ A linguagem de um computador é baseado em impulsos elétricos (0 = desligado, 1 = ligado);



Como programas são feitos?

◆ Linguagem de Máquina

- ◆ 01010101
- ◆ 0001011111101100
- ◆ 10000011111000100111110100
- ◆ 100010010100010111111100
- ◆ 00110011111001101
- ◆ 100010010100010111111100
- ◆ 1100011101000101111100001000000
- ◆ 10000011010001011111010000001010

- ◆ 111111110100010111111000
- ◆ 10000011011111011111100000000110
- ◆ 0111010111110011
- ◆ 10001011111100101
- ◆ 01011101
- ◆ 11000011



Como programas são feitos?

◆ Linguagem de Montagem

- LOOP:
- LARP AR1
- LRLK AR1,apontador
- ADRK TAMANHO_CONSTANTE
- ADRK fimcon_r̄x
- LAC *
- BZ NAOPASSARAM10MS;
- ZAC
- SACL *
- LARP AR1
- LRLK AR1,apontador+CONSTANTE_A
- ADRK controle
- LAC *
- BZ LOOP;NAO DECORRIDO TEMPO: FICA NO LOOP
- NAOPASSARAM10MS:
- SACL *
- LARP AR1
- B LOOP



Dificuldade

- Um programa era difícil, longo e principalmente caro de o construir;
- Era também difícil de ser entendido por outros programadores;
- Essa complexidade levou à necessidade de desenvolver novas técnicas e ferramentas.



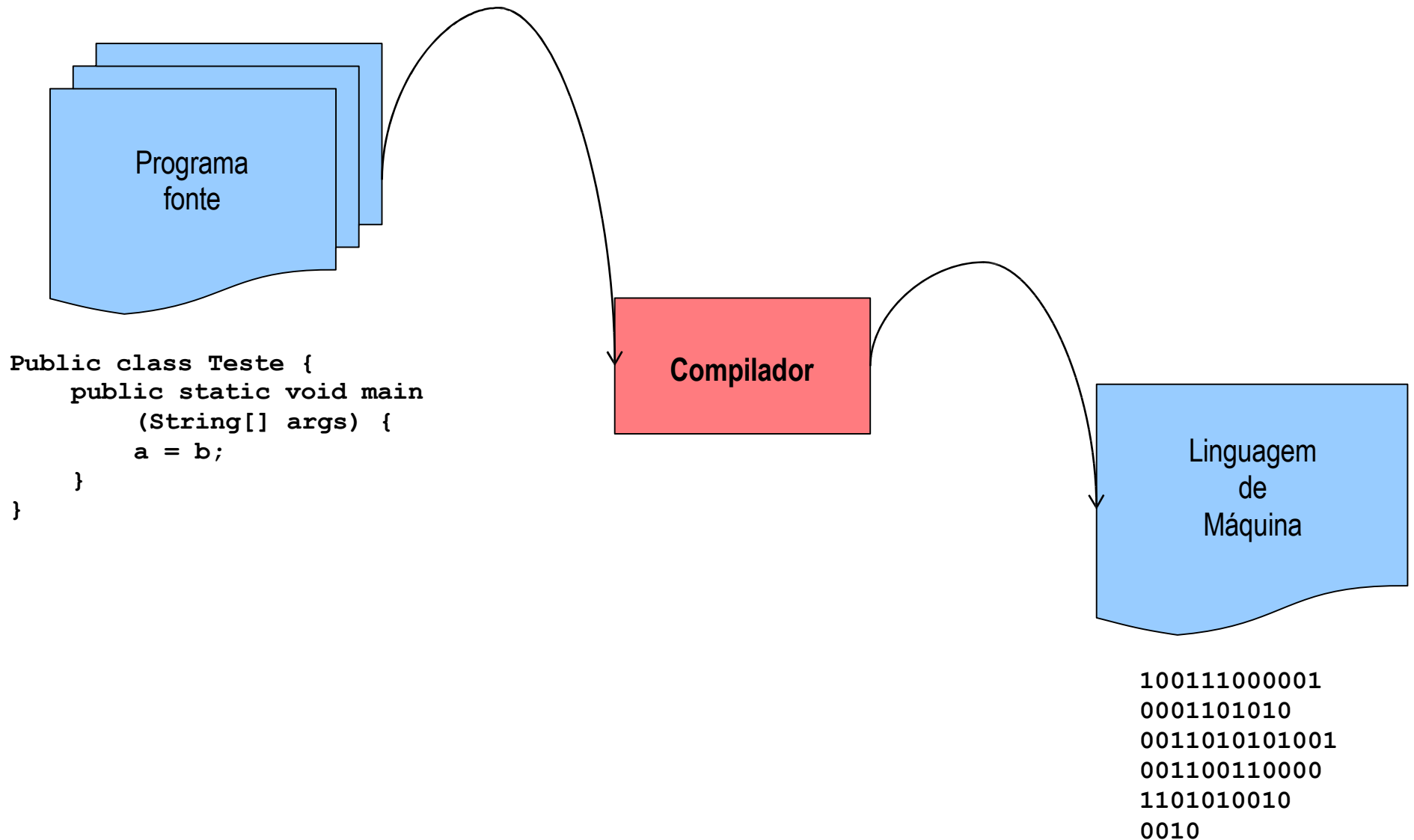
Linguagens de Alto Nível

- **Exemplo em C:**

```
#include <stdio.h>
int main()
{
    int a = 3, b = 7;
    printf("a + b = %d\n", a+b);
    return 0;
}
```



Linguagens de Alto Nível



Linguagens de Alto Nível

[2.PAK](#)
[20-GATE](#)
[473L Query](#)
[51forth](#)
[A+](#)
[A++](#)
[A#.NET](#)
[A# \(Axiom\)](#)
[A-0](#)
[ABAP](#)
[ABC](#)
[Actors](#)
[ADA](#)
[ADVPL](#)
[ALGOL](#)
[APL](#)
[Assembly](#)
[Autocoder](#)
[ABLE](#)
[ABSET](#)
[ABSYS](#)
[ACC](#)
[Accent](#)
[Action!](#)
[ACS](#)
[ActionScript](#)
[ADS/Online](#)
[AdvPL Afnix](#)
[AIMMS](#)
[Alan](#)
[Aldor](#)
[Alef](#)
[Algae](#)

[Alice](#)
[Alphard](#)

[AppleScript](#)
[Argos](#)
[ARS++](#)
[AspectJ](#)
[ATLAS](#)
[Atlas Autocode](#)
[ATOLL](#)
[Aubit-4GL](#)
[Autocoder](#)
[Autolt](#)
[AutoLISP](#)
[Averest](#)
[AWK](#)
[Axiom](#)
[Axiom-XL](#)
[B](#)
[BACI](#)
[BASIC](#)
[Bash](#)
[BC](#)
[BCPL](#)
[BeanShell](#)
[BETA](#)
[Bigwig](#)
[Bistro](#)
[BLISS](#)
[Blitz Basic](#)
[BALM](#)
[Blue](#)
[Boo](#)
[Bourne shell](#)
[Boxx](#)
[BPEL](#)
[Brainfuck](#)
[BUGSYS](#)
[BuildProfessional](#)

[Cw](#)
[Caché](#)
[CamI](#)
[Clarion](#)
[Clean](#)
[ColdFusion](#)
[Clipper](#)
[CLOS](#)
[COBOL](#)
[Common Lisp](#)
[CPL](#)
[CPython](#)
[C shell](#)
[D](#)
[Datalog](#)
[Dataflex](#)
[dBase](#)
[Delphi](#)
[DisCo](#)
[Dynamic C](#)
[Eiffel](#)
[Euphoria](#)
[Fortran](#)
[Forth](#)
[FoxPro](#)
[Flow-Matic](#)
[GAP](#)
[G-Portugol](#)
[Graforth](#)
[Groove](#)
[Haskell](#)
[Haskell.NET ou H#](#)
[IAL](#)
[Icon](#)
[Informix-4GL](#)
[Inger](#)

[J#](#)
[JavaScript](#)
[JCL](#)
[Joiner](#)
[Jovial](#)
[Just BASIC](#)
[Kid's](#)
[Limbo](#)
[Lingo](#)
[Lisp](#)
[Logic Basic](#)
[Logo](#)
[Lotusscript](#)
[Lua](#)
[Lua.NET](#)
[Matlab](#)
[Meta4](#)
[Miranda](#)
[mIRC Scripting](#)
[ML](#)
[Modula](#)
[Modula-2](#)
[MonoBASIC](#)
[Mortran](#)
[Mumps](#)
[Nemerle](#)
[Object Pascal](#)
[O2](#)
[Oberon](#)
[Objective-C](#)
[Ook# .NET](#)
[Pascal](#)
[Perl](#)
[PerlSharp](#)
[Phalanger](#)
[PHP](#)

[PostScript](#)
[Powerbuilder](#)
[Prolog](#)
[Python](#)
[Python.NET](#)
[POV-Ray](#)
[QUEL](#)
[QT](#)
[R \(estatística\)](#)
[RealBasic](#)
[RPG](#)
[RPL](#)
[Ruby](#)
[Ruby/.NET Bridge](#)
[SasI](#)
[Scheme](#)
[Scriptol](#)
[sh](#)
[Simula](#)
[Smalltalk](#)
[Snobol 4](#)
[SQL](#)
[T](#)
[TACL](#)
[TACPOL](#)
[TADS](#)
[Transaction](#)
[Tcl](#)
[Transact SQL](#)
[teco](#)
[TELCOMP](#)
[Telon](#)
[Tempo](#)
[Titanium](#)
[TI-Basic](#)
[Today](#)

[TTCN](#)
[Turing](#)
[TUTOR](#)
[Tutorial D](#)
[TXL](#)
[Ubercode](#)
[Ultra 32](#)
[Unicon](#)
[Uniface](#)
[UnrealScript](#)
[Visual Basic](#)
[VBScript](#)
[Visual Basic for Applications](#)
[Visual Basic .NET](#)
[Visual Objects](#)
[Visual Smalltalk](#)
[Water](#)
[WATFIV](#)
[Whitespace](#)
[Winbatch](#)
[WinDev](#)
[Windows PowerShell](#)
[WML](#)
[X10](#)
[XBL](#)
[xbScript](#)
[xHarbour](#)
[XL](#)
[XOTcl](#)
[XPL](#)
[XPL0](#)
[XQuery](#)
[XSLT](#)
[YAFL](#)
[Yellowra Ada](#)
[Yorick](#)



Tarefa menos fácil do que parece!



Facebook: 12 meses, mais de 400.000 linhas de código, uma equipe de mais de 60 pessoas.



Microsoft Office: +1.000.000 linhas de código.



Windows: 6 anos, US\$ 16 bilhões e 50 milhões de linhas de código.



Software

- De nada adianta a existência do hardware sem o software. É como se fosse um automóvel sem combustível apropriado.
- O que faz com que seja possível operacionalizar o hardware é o software.
- Um software é um programa que possui sequências de atitudes lógicas (passos) a serem tomadas em cada situação previamente determinada.



Software Básico

- Exerce o papel de tomar o primeiro contato com o hardware, além de realizar o elo entre a máquina e os demais softwares.
- Realiza a administração/controle básico do funcionamento do equipamento. Exemplo: Sistema Operacional.

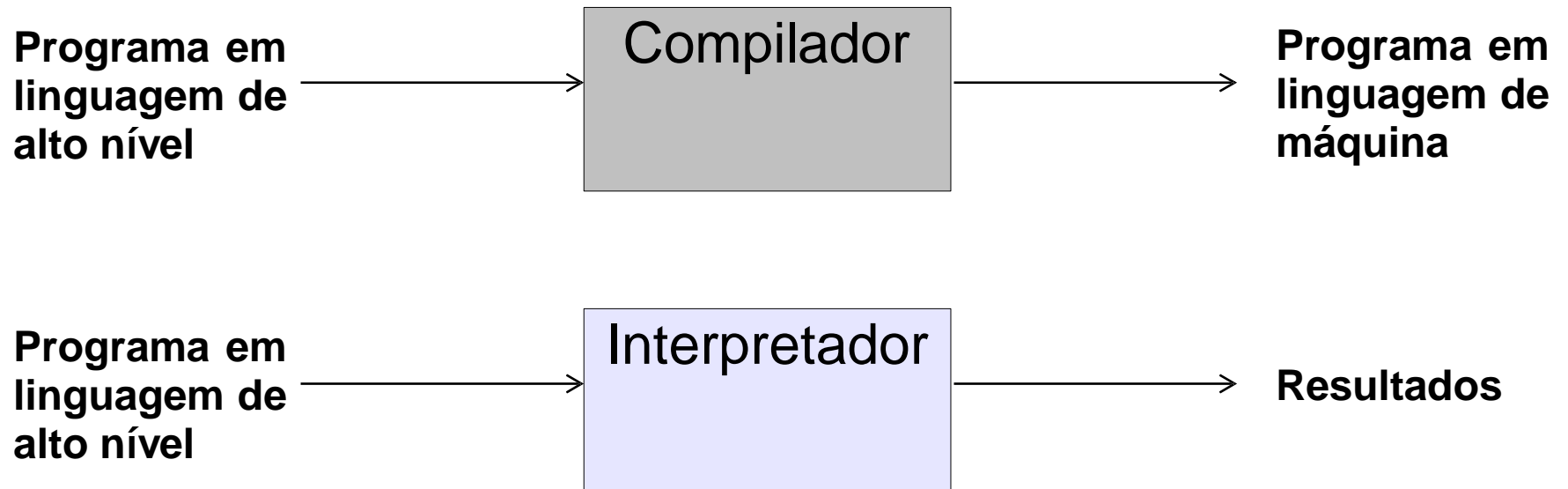


Software de Desenvolvimento

- São softwares utilizados pelos profissionais de computação para desenvolver programas.
 - **Compilador:** Tradução de linguagem de alto nível para linguagem de máquina antes do programa começar a executar;
 - **Interpretador:** Tradução de linguagem de alto nível para linguagem de máquina durante execução do programa.



Software de Desenvolvimento



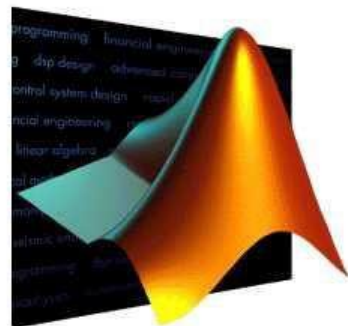
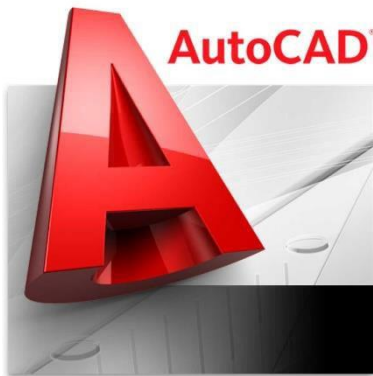
Softwares Aplicativos

•São softwares voltados para um objetivo previamente definido, ou seja, voltados para atender determinado tipo de necessidade do usuário de informática. Exemplos: processadores de texto (Microsoft Word, BrOffice Writer, Latex), planilhas eletrônicas (Microsoft Excel, BrOffice Calc), editores de gráficos (Inkscape, Gimp, Paint).



Softwares Aplicativos Específicos

- São uma especificidade dos softwares aplicativos. São softwares moldados às necessidades específicas de determinada clientela, ramo ou atividade. Exemplos: desenho de plantas, matemática computacional, edição de imagens, etc.



Referências

FORBELLONE, A. L. V. e EBERSPACHER, H. F. Lógica de Programação – A Construção de Algoritmos e Estrutura de Dados. 3ª Edição. Prentice Hall, 2005.

