# INTRODUÇÃO À COMPUTAÇÃO

Vinícius Godoy

# Introdução à Computação

Vinícius Godoy

IESDE BRASIL S/A
2018

CIP-BRASIL. CATALOGAÇÃO NA PUBLICAÇÃO
SINDICATO NACIONAL DOS EDITORES DE LIVROS, RJ

G535i Godoy, Vinícius

Introdução à computação / Vinícius Godoy. - 1. ed. - Curitiba [PR] : IESDE Brasil, 2018.
130 p. : il. ; 21 cm.
Inclui bibliografia
ISBN 978-85-387-6365-9

1. Computação. I. Título.

17-46726

CDD: 004
CDU: 004

Capa: IESDE BRASIL S/A.
Imagem da capa: v_alex/iStockphoto



**IESDE BRASIL S/A.**
Al. Dr. Carlos de Carvalho, 1.482. CEP: 80730-200
Batel – Curitiba – PR
0800 708 88 88 – www.iesde.com.br

Na atualidade, a informática está presente em nosso dia a dia e em praticamente todos os dispositivos que utilizamos.

Como profissional da informática, você está prestes a entrar de cabeça no futuro. Irá conhecer tecnologias que estão por vir, como a computação quântica, e falar diariamente sobre assuntos como realidade aumentada, aprendizagem de máquina e trabalhar com uma série de aplicativos ou dispositivos que serão conhecidos por seus familiares ou amigos só alguns anos à frente.

Este livro proporciona uma breve introdução a essa área fascinante. Nele, você conhecerá um pouco da história de como os computadores sugiram e evoluíram. Entenderá sua organização básica, seu sistema operacional e o que exatamente são bits e bytes.

As próximas páginas também irão lhe possibilitar uma visão sobre esse mercado em que você está entrando ou já faz parte. Você entenderá a importância da internet e da web e dos dispositivos móveis para os negócios e terá conhecimento das posições em que poderá atuar como profissional. Finalmente, o livro trará uma reflexão sobre o presente e o futuro da computação.

Esta obra foi elaborada para que você inicie seus estudos na área, acostume-se com o vocabulário utilizado e tenha uma base esclarecedora sobre a profissão escolhida por você – tudo com uma leitura leve, considerando os seus primeiros passos no universo da informática.

Desejamos a você muito sucesso nesse campo fascinante e uma excelente leitura!

## Vinícius Godoy

Mestre em Visão Computacional pela Pontifícia Universidade Católica do Paraná (PUCPR), especialista em Desenvolvimento de Jogos de Computadores pela Universidade Positivo (UP) e graduado em Tecnologia em Informática pela Universidade Tecnológica Federal do Paraná (UTFPR).

# Sumário


**1 História da informática** 9

**1.1** Antes da eletrônica 9

**1.2** Dos relés ao circuito integrado 13

**1.3** O presente 16

**2 A internet** 23

**2.1** A história da internet 24

**2.2** Como a internet funciona 29

**2.3** A internet em nossas vidas 33

**3 Mercado de informática** 41

**3.1** Desenvolvimento 42

**3.2** Informática gerencial 45

**3.3** Pesquisas científicas 48

**4 Organização física do computador - Hardware** 59

**4.1** Tipos de computadores 60

**4.2** Componentes 62

**4.3** Arquitetura 63

**5 Organização lógica do computador - O sistema operacional 69**

**5.1** Bits e bytes 70

**5.2** Bases numéricas 71

**5.3** Tipos de dados 75

**6 Bits, bytes e hexadecimal - A informação no computador 91**

**6.1** Breve histórico 92

**6.2** Funcionamento 94

**6.3** Arquiteturas 98

**7 Linguagens de programação 103**

**7.1** Linguagens 104

**7.2** Do texto ao software 107

**7.3** Paradigmas 110

**8 O futuro da informática 119**

**8.1** O ensino 120

**8.2** Realidade virtual 123

**8.3** Computação quântica 125

# 1 História da informática

Neste livro, você aprenderá a terminologia básica da área de computação e entenderá como se situar no mercado. Aprenderá o funcionamento básico do dispositivo fascinante que é parte da vida de qualquer cientista da computação: o computador.

Iniciaremos esta jornada explorando a história dos computadores, tema deste capítulo. Veremos as várias pessoas e transformações que ocorreram nesses dispositivos até a atualidade.

## 1.1 Antes da eletrônica

Vídeo

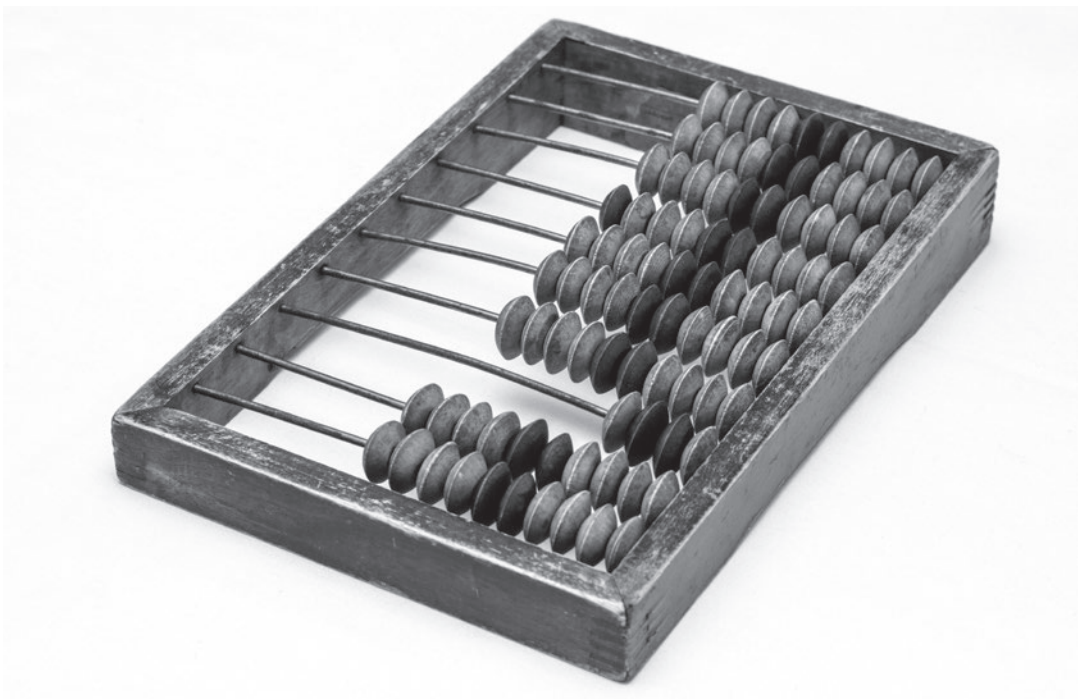
Pode parecer inacreditável, mas a história da computação tem mais de 7 mil anos! Você pode estar se perguntando: mas como isso é possível, uma vez que computadores dependem de eletricidade?

# 1 História da informática

O computador é uma máquina capaz de: armazenar dados; realizar cálculos e operações lógicas; obedecer a um conjunto prévio de instruções.

Obviamente, os primeiros dispositivos ainda não eram computadores completos. A história da informática começa no primeiro dispositivo criado pelo ser humano para auxiliar em seus cálculos, o **ábaco** (Figura 1).

**Figura 1 – Ábaco russo.**



Fonte: Alexander_Fagaulin/iStockphoto.

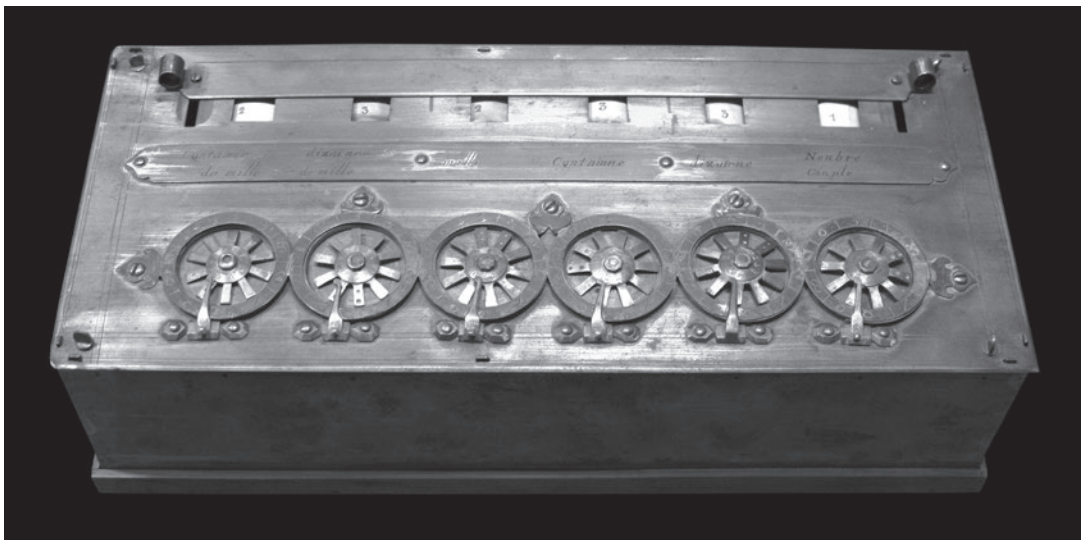
Não se sabe exatamente onde o ábaco foi criado, mas o primeiro registro que temos notícia é cerca de 5500 a. C., na Mesopotâmia. Esse dispositivo simples já apresenta uma série de características dos computadores modernos: auxiliar em cálculos aritméticos e ter memória, representada pelas suas contas.

Em 1638, o matemático renascentista William Oughtred estudava os trabalhos de John Napier sobre logaritmos e precisava de uma forma de multiplicar números muito grandes com facilidade. Ao estudar várias propriedades matemáticas desses números, Oughtred projetou uma régua com números e operações pré-marcadas, que ficou conhecida como *régua de cálculo*. Esse dispositivo permaneceu em uso nos cursos de engenharia até o surgimento das calculadoras (WAZLAWICK, 2016).

Embora a régua fosse útil, ela só realizava operações predefinidas. A primeira calculadora capaz de realizar somente somas e subtrações foi criada em 1642 por Blaise Pascal e foi batizada em sua homenagem com o nome de *Pascalina* (Figura 2). Essa calculadora

funcionava de maneira similar ao medidor de quilometragem de um carro. Se você quisesse somar 2+3, bastaria girar uma roda duas vezes para direita, e, em seguida, mais duas vezes para direita e o marcador mostraria o resultado 5. Girar em sentido oposto faria a subtração.

**Figura 2 – Pascalina.**



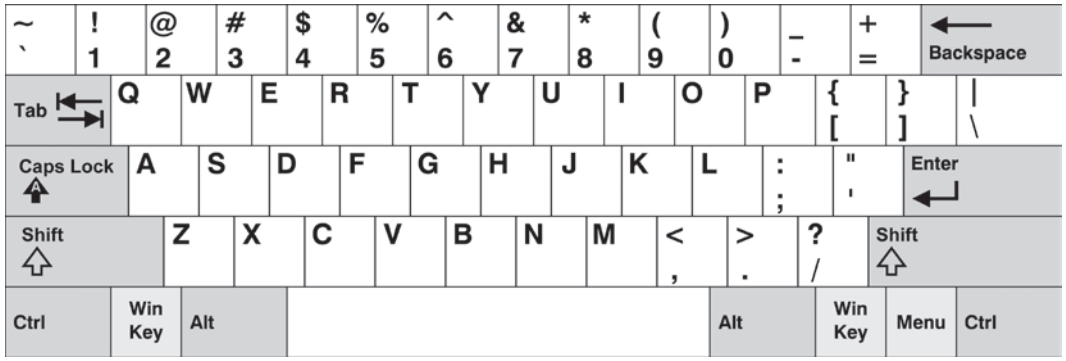
Fonte: Rama/Wikimedia Commons.

Dadas as suas limitações e dificuldade de manuseio, ela nunca se tornou um dispositivo realmente popular. Mas, em 1672, o alemão Gottfried Wilhelm Leibniz conseguiu expandir a calculadora de Pascal e criar a primeira calculadora mecânica capaz de realizar as quatro operações básicas e, até mesmo, calcular a raiz quadrada (WAZLAWICK, 2016).

Em 1847, o matemático George Boole demonstrou que era possível combinar variáveis, por meio das preposições lógicas *e*, *ou* e *não*. Ele então criou símbolos para essas preposições e as associou à teoria dos conjuntos, demonstrando inclusive que operações matemáticas comuns poderiam ser escritas de maneira lógica. Por exemplo, (x-1) poderia ser escrito como o conjunto de todas as cores do mundo, menos o branco. Esse trabalho formalizou o que ficou conhecido como *álgebra booleana* (CARVALHO; LORENA, 2016).

Em 1867, surgiu a primeira máquina de escrever comercial, criada por Christopher Soulen, Carlos Glidden e Samuel Soulen. A máquina era constituída por um conjunto de baquetas mecânicas, associadas a cada tecla. Os primeiros protótipos, entretanto, enganchavam os tipos gráficos, não permitindo que se digitasse muito rapidamente. Para resolver esse problema, em 1878, Soulen decidiu associar as teclas de uso mais frequente, como as vogais, aos dedos de menor agilidade, o que impedia uma digitação veloz. A máquina então tornou-se um fenômeno de vendas e tornou-se um padrão em 1910. Essa disposição do teclado ficou conhecida como *QWERTY* (Figura 3) e é utilizada até a atualidade (WAZLAWICK, 2016).

**Figura 3 – Teclado QWERTY norte-americano.**




Fonte: Wikimedia Commons.

Houve até uma tentativa de mudança, proposta por August Dvorak, em 1930, que associava os dedos rápidos a teclas mais usadas. Essa disposição exige menos esforço, acelera a digitação e é recomendada por alguns ergonomistas. Contudo o padrão de Dvorak nunca se propagou. Versões desse teclado foram propostas para diversos países, inclusive o Brasil (Figura 4).

**Figura 4 – Teclado Dvorak brasileiro.**


Fonte: Wikimedia Commons.

No ano de 1880, o censo americano levou sete anos e meio para ser realizado. A maior demora estava em totalizar os dados manualmente. Isso motivou Herman Hollerith a desenvolver uma máquina eletromecânica para ajudar nos cálculos do censo dos anos 1890, já que o prazo máximo estipulado em lei para tal tarefa não poderia ultrapassar dez anos. Essa máquina, depois foi aperfeiçoada por Hollerith para cálculos contábeis, foi vendida para empresas interessadas em processar folhas de pagamentos. Por isso, até a atualidade chamamos os comprovantes de *holerites*. Com o auxílio dessa máquina, o censo levou apenas dois anos para ser computado. A máquina utilizava cartões perfurados para a entrada de dados e sua organização serviu de inspiração para a primeira Unidade Lógico Aritmética (ULA) dos futuros computadores (CHRISTIAN; GRIFFITHS, 2017).

Nenhuma dessas máquinas, entretanto, dispunha de uma das principais características de um computador: obedecer a um conjunto de instruções, ou, em outros termos, ser *programável*.

Essa invenção só surgiu em 1822, idealizada pelo inglês Charles Babbage (WAZLAWICK, 2016). Tratava-se de um computador mecânico chamado de *máquina analítica* e foi criado

originalmente para automatizar o processo de geração de tabelas de cálculo polinomial. O engenheiro militar italiano Luigi Menabrea escreveu um trabalho sobre essa máquina, que posteriormente foi traduzido para o inglês por Augusta Ada King-Noel, condessa de Lovelace, conhecida como *Ada Lovelace*.

Ada Lovelace conheceu Charles Babbage em 1833, incluindo notas próprias nesse trabalho de tradução. Ada reconheceu o potencial da máquina e sua capacidade de executar um conjunto de instruções, escrevendo então o primeiro programa de computador, que realizava o cálculo das sequências de Bernoulli. Com esse fato, Lovelace torna-se a primeira programadora da história e Babbage passa a ser considerado o pai da computação (WAZLAWICK, 2016).

## 1.2 Dos relés ao circuito integrado

Vídeo

O uso de energia elétrica surgiu em meados do século XIX, porém, somente em 1835 revolucionou os dispositivos mecânicos, com a invenção dos relés. Eles nada mais são do que interruptores de energia capazes de ligar e desligar com a presença de uma corrente elétrica. Foram utilizados para criar as primeiras centrais telefônicas, substituindo as telefonistas por comutação automática.

Porém somente com o surgimento da válvula, em 1906, que foram criados os primeiros computadores realmente eletrônicos. Entretanto o primeiro dispositivo a utilizar lógica binária surgiu somente em 1931, com Vannevar Bush (WAZLAWICK, 2016).

### 1.2.1 Primeira geração (1940-1959)

A primeira geração de computadores eletrônicos foi largamente impulsionada pela guerra, já que realizavam cálculos de balística e, posteriormente, decodificavam mensagens inimigas.

Howard Aiken desenvolveu o computador Mark I, que foi construído entre os anos de 1939 e 1944 pela Universidade de Harvard. Esse computador foi utilizado pela marinha americana até 1959, quando foi desativado. Ele utilizava relés eletromecânicos, pesava cerca de 5 toneladas, era lento e barulhento. Os números de até 23 dígitos eram armazenados em 3 mil rodas giratórias, além de ter 1.400 interruptores giratórios controlados e conectados por cerca de 700 quilômetros de cabos. Uma das grandes características da arquitetura de Harvard era usar espaços de memória separados para o programa e para os dados.

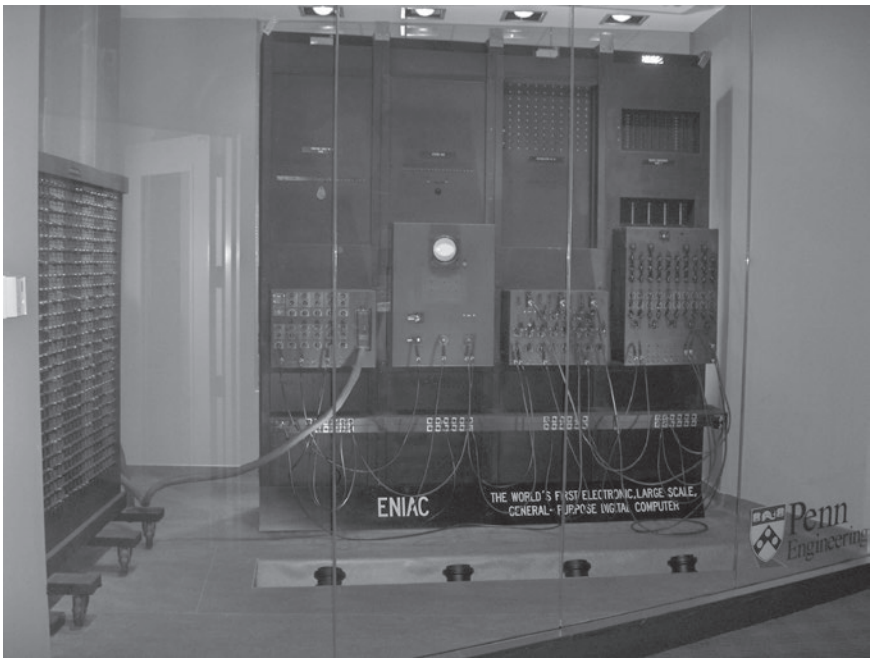
Em 1943, Allan Turing trabalhou na construção e programação de um dos mais importantes computadores da história, o Colossus (CARVALHO; LORENA, 2016). Foi o primeiro computador 100% valvulado, portanto, eletrônico. Turing também utilizou memória e dados em um mesmo local.

Anos antes, em 1936, Turin demonstrou que um conjunto de instruções e dados poderiam ser utilizados em seu computador para resolver qualquer cálculo. Esse conceito foi batizado de *Turin completude*. O Colossus utilizava 1.500 válvulas e conseguia ler textos por

um leitor óptico. Foi utilizado na guerra para decodificar mensagens nazistas e inspirou o filme *O jogo da imitação* (2014).

Outro computador importante foi o *Eniac* (Figura 5), cuja construção terminou em maio de 1944. Com 17.468 válvulas, 1.500 relays, 70 mil resistores, 10 mil capacitores e 5 milhões de pontos de solda feitos à mão, custou 500 mil dólares. Pesava 27 toneladas, ocupava uma área de 63 m² e consumia 150 kW de energia. Porém, na época, foi descrito como um computador mais rápido que o pensamento (CARVALHO; LORENA, 2016).

**Figura 5 – Eniac.**




Fonte: Wikimedia Commons.

Uma de suas desvantagens era o fato de que precisava ser calculado com ligações de cabos, o que era difícil e sujeito a erros. Isso impulsionou o matemático John Von Neumann a resolver o problema, criando uma arquitetura que permitia utilizar dados e programa em um mesmo local, o que possibilitava ao computador alterar a própria programação. Essa arquitetura ficou conhecida como *Arquitetura de Von Neumann*, utilizada até os dias atuais (WAZLAWICK, 2016).

## 1.2.2 Segunda geração (1953-1963)

Na segunda geração, um novo componente substituiu as válvulas: o transistor, criado por William B. Shockley, no Bell Labs, em 1947. Além de menor, os transistores podiam ser chaveados mais rapidamente e não queimavam, o que tornou a ferramenta ideal para criar os minicomputadores – ainda assim, eram máquinas gigantescas que ocupavam salas inteiras.

O primeiro computador dessa geração foi o IBM 7030, também conhecido como *Stretch*, em 1961. Durante sua construção, a IBM fez promessas de que o computador seria muitíssimo rápido, capaz de executar 4 milhões de instruções por segundo (MIPS). Porém, após

lançado, o computador só foi capaz de atingir a velocidade de 1,2 MIPS e, embora ainda fosse considerado o mais rápido para a época, fez com que seu preço final caísse de 13,5 milhões de dólares para 7,8 milhões, clientes desistissem e o produto fosse retirado do mercado. Ainda assim, a construção desse computador criou avanços importantes, como o conceito de interrupções, multiprogramação, memória protegida e várias das bases da aritmética de ponto flutuante (CARVALHO; LORENA, 2016).

Outro computador importante dessa geração foi o PDP-8 (Figura 6), produzido pela Digital Equipment Corporation (DEC). Sua importância histórica está no fato de que foi considerado o primeiro minicomputador de sucesso comercial. Era comercializado pelo preço de 18.500 dólares. Ele permitiu que praticamente todas as faculdades, centros de pesquisa e grandes empresas americanas possuíssem um computador. Em torno dele, foram organizados os primeiros cursos de computação e foram formados os primeiros programadores profissionais (CHRISTIAN; GRIFFITHS, 2017).

**Figura 6 – PDP-8 aberto, exibindo seus módulos de memória.**


Fonte: Morn/Wikimedia Commons.

## 1.2.3 Terceira geração (1963-1970)

A terceira geração inicia-se pelo uso de circuitos integrados. Um dos mais populares computadores da época é o IBM System 360 modelo 91, entregue em 1964. Seu uso ficou famoso, pois foi utilizado pela Nasa e, posteriormente, pela Universidade de Columbia. Surgiu para ser concorrente de outro computador, o CDC 6600, feito pela Control Data Corporation e entregue ao laboratório do Cern na Suíça, em 1965.

O CDC 6600 também foi um computador importante, já que seu projeto foi precursor da filosofia de design RISC, usada até a atualidade (WAZLAWICK, 2016).

Vários fatos importantes também ocorreram nessa geração (HILTZIK, 2009):

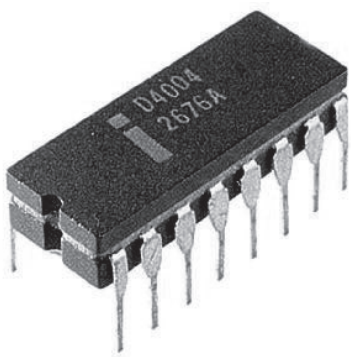
- 1963 – Douglas Engelbart patenteia o mouse, que é vendido pela Xerox para a Apple por US$ 40.000,00.
- 1964 – Paul Baran, pesquisador norte-americano, projeta e cria a primeira rede de computadores interligada por fios.
- 1968 – Engelbart cria um sistema operacional com mouse, teclado e janelas (que serviu como base para o Windows e o sistema do Apple II).
- 1968 – Engelbart cria um padrão de hipertexto, que futuramente serviu de inspiração para o protocolo HTML.
- 1969 – Engelbart e Leonard Kleinrock desenvolvem a *Arpanet*, rede que viria a se tornar a internet.
- 1969 – A AT&T cria o sistema operacional *Unix*: portável, multitarefa, time-sharing e monolítico.

Vídeo

## 1.3 O presente

Em 1972 surgiu o Intel 8008, um chip cujo conjunto de instruções ainda é utilizado em muitos cursos de eletrônica básica. Sua versão melhorada – a 8088, lançada em 1974 (Figura 7) – foi utilizada nos primeiros computadores pessoais.

**Figura 7 – Intel 4004.**




Fonte: Wikimedia Commons.

Também surgiu em 1927 o primeiro videogame, chamado *Odyssey* e fabricado pela Magnavox, obtendo recorde de vendas para um computador pessoal na época: 300 mil unidades (HANSEN, 2016). No mesmo ano, a Atari lança o primeiro game para fliperama: o jogo Pong, reescrita de um game de 1958 chamado *Tennis for Two*, criado por William A. Higinbotham para ser jogado em um osciloscópio. O videogame doméstico Atari foi lançado em 1973.

Entre 1973 e 1975 foi criada a placa de rede ethernet, que se tornou padrão da indústria na atualidade. Ela foi criada por Robert Metcalfe e David Boggs no laboratório PARC da Xerox. Em 1976, um artigo científico com um modelo experimental da placa de 3 MB foi publicado. Porém a Xerox mostrou pouco interesse em produzir essas peças, o que fez com que Metcalfe saísse da companhia em 1979 para promover a adoção da tecnologia (HILTZIK, 2009).

Em 1977, é inaugurada no mercado a grande concorrente da Intel, a AMD, produzindo um clone do chip 8080. No mesmo ano, a Apple criou o computador pessoal mais bem-sucedido da época, chamado de *Apple II*. Ele já dispunha de interface gráfica e áudio. Foi muito adquirido por escolas americanas de classe média e alta.

No ano de 1981, os computadores começavam a ganhar cores com a placa de vídeo CGA. Era a primeira a suportar resolução gráfica de 320x200 pixels e colocar 4 de 16 cores ao mesmo tempo na tela. Entretanto, apesar do sucesso, era uma placa deplorável se comparada ao Commodore 64 ou ao Atari da época (HANSEN, 2016).

No ano de 1982, foi lançado, nos EUA, o 80286 (WAZLAWICK, 2016). Esse computador foi o percursor do Pentium e de toda a linha x86. Já era capaz de endereçar 16 MB de RAM e já incluía recursos avançados, como memória protegida e preempção, que só viriam a ser utilizados pelo Windows 95, mas que já se via em sistemas Unix. O 285 trabalhou em velocidades de 8 a 25 MHz e foi utilizado até o início dos anos 1990.

Enquanto isso, o Brasil vivia em um período de mercado fechado e ditadura militar. Em 1982, popularizou-se por aqui o computador CP-500 da Prológica (CARVALHO; LORENA, 2016). Ele era a réplica do computador TSR-80, fabricado nos Estados Unidos na década de 1970, seu último lançamento foi em 1987, com velocidade de 4 MHz e só durou tanto tempo graças à reserva de mercado.

Em 1984 a IBM lançou a placa gráfica EGA, capaz de reproduzir 16 cores simultâneas e fornecer animações mais suaves do que as CGA. A placa rapidamente dominou o mercado, mostrando o potencial da tecnologia gráfica. Ela também impulsionou fortemente a indústria de games, uma vez que praticamente todos os lares americanos já contavam com um computador (HANSEN, 2016).

Em 1985, surgiu o Microsoft Windows 1.0. O Windows ainda era um software separado do sistema operacional. Na época, o DOS (Disk Operating System) era vendido por US$ 99.

Em 1985, já com o fim das barreiras impostas pelos militares, entrou no Brasil o computador MSX. Ele foi o primeiro a propor que se tornasse o centro multimídia da casa. Foi importado pela Sharp com o nome de HotBit e pela Toshiba com o nome de Expert. E, assim como videogames, tinha entrada para cartuchos, placa de vídeo de 16 cores, áudio e utilizava a televisão como monitor.

No mesmo ano, foi desenvolvido o processador Intel 80386. Dois anos depois, a IBM lançou a placa VGA, capaz de reproduzir 65.536 cores simultâneas e imagens de até 1024x768 pixels (HANSEN, 2016). Em um erro estratégico incrível, a IBM acreditou que essa placa tinha qualidade tão alta que teria pouca aplicação e praticamente retirou-se desse mercado.

Pouco tempo depois, outros fabricantes aderiram a um padrão conhecido como *VESA* – uma nova geração de placas de vídeo com qualidade muito superior.

Em 1991, Linus Torvalds lançou o sistema Linux (WAZLAWICK, 2016). Tratava-se de uma alternativa grátis ao Windows, de código aberto e baseada nos sistemas Unix. Rapidamente, espalhou-se por servidores e atualmente lidera as iniciativas de software livre, inclusive implantadas no governo brasileiro.

Em 1993, surgiu o processador Pentium (FONSECA FILHO, 2007). Ele rapidamente evoluiu para atingir a barreira de 3,9 GHz. Contudo esbarrou em limitações físicas do silício, já que microchips mais rápidos começavam a perder bits na forma de ondas magnéticas. A partir daí, os fabricantes tiveram de apostar em outros recursos, como o aumento do paralelismo, com mais núcleos de processamento, e tentar resolver problemas de performance em outras áreas do computador (memória, barramento, discos rígidos etc.). A Tabela 1 resume essa evolução:

**Tabela 1 – CPUs Intel.**

| CPUs Intel (2004-2013) | | | | | | |
|---|---|---|---|---|---|---|
| **Nome** | **Ano** | **Núcleo** | **Arquitetura** | **Nanômetros** | **Clock** | **Largura de dados** |
| Pentium 4 | 2004 | 1 | Prescott | 180 nm | 3.6 GHz | 32 bits |
| Pentium D | 2005 | 2 | Smithfield | 90 nm | 3.2 GHz | 32 bits |
| Core 2 Duo | 2007 | 2 | Conroe/ Allendale/ Wolfdale | 65 nm | 3 GHz | 64 bits |
| Core i3, i5 e i7 (1ª geração) | 2008 | 3 | Nehalem | 45 nm | 2.6 GHz | 64 bits |
| Core i3, i5 e i7 (2ª geração) | 2011 | 2 a 4 | Sandy Bridge | 32 nm | 3.9 GHz | 64 bits |
| Core i3, i5 e i7 (3ª geração) | 2012 | 2 a 4 | Ivy Bridge | 22 nm | 3.9 GHz | 64 bits |
| Core i3, i5 e i7 (4ª geração) | 2013 | 2 a 4 | Haswell | 22 nm | 3.9 GHz | 64 bits |

Fonte: ROCKETZ, 2014.

Atualmente, vivemos em uma era na qual processadores estão em praticamente todos os dispositivos: automóveis, celulares e até relógios. Os equipamentos incluíram diversos novos sensores como acelerômetros, câmeras, giroscópios e acesso a redes GPS, além de estarem totalmente integrados por redes sem fio e internet. Começamos a ver aplicações incríveis de inteligência artificial, como o carro do Google, que se dirige sozinho. Videogames

tornaram-se verdadeiras centrais de entretenimento caseiras, dando acesso a vídeos em streaming, realidade virtual e utilizando TVs de 4k.

Pode parecer assustador, mas esse ambiente é um campo fértil para profissionais da informática, como você.

## Ampliando seus conhecimentos

Deixamos você aqui com um trecho da introdução do livro *Rise of the Robots: Technology and the threat of a jobless future* (2015), de Martin Ford.

### Aumento dos robôs[1]

*(FORD, 2015)*

Essa é uma era que será definida por uma mudança fundamental no relacionamento entre trabalhadores e máquinas. Essa mudança vai desafiar um dos nossos pressupostos mais básicos sobre tecnologia: que *máquinas são ferramentas* que aumentam a produtividade dos trabalhadores. No lugar, máquinas estão se tornando elas mesmas os trabalhadores, e a linha entre capacidade de trabalho e capital está se tornando mais tênue do que nunca.

Todo esse progresso é, obviamente, conduzido pela incansável aceleração na tecnologia dos computadores. Enquanto a maioria das pessoas está familiarizada com a lei de Moore – uma regra geral de que a tecnologia dobra de velocidade a cada 16 ou 18 meses – poucas assimilaram totalmente as implicações desse progresso exponencial.

Imagine que você entre em um carro e comece a dirigir à velocidade de 5 milhas por hora (mph). Você dirige por um minuto, então acelera e dobra sua velocidade para 10 mph, dirige outro minuto, dobra sua velocidade novamente, e assim por diante. O que é impressionante não é dobrar a velocidade, mas a quantidade de terra que você percorrerá quando esse processo ocorrer por um tempo. No primeiro minuto, você terá percorrido 134 metros. No terceiro minuto, a uma velocidade já de 20 mph, você terá percorrido 536 metros. No décimo quinto minuto, já a uma velocidade de 80 mph, você já teria dirigido por mais de 1.500 quilômetros. No décimo sexto minuto, você já precisaria de um carro mais rápido e de uma pista de corrida.

Agora, imagine o quão rápido você estaria indo – e quanto progresso você faria até o minuto final – se você dobrasse sua velocidade 27 vezes. Essa é

1 Tradução do autor.

aproximadamente a quantidade de vezes que a velocidade dos computadores dobrou desde a invenção do circuito integrado, em 1958. A revolução que está a caminho está acontecendo não só por causa da aceleração em si, mas também porque *essa aceleração está ocorrendo há tanto tempo* que podemos esperar em um determinando ano é inimaginável.

[...]

Em 2008, quando a crise mundial ocorreu, comecei a pensar seriamente na implicação dessas dobradas consistentes na velocidade na computação e na probabilidade de que ela dramaticamente alteraria os empregos no mercado e a economia geral ao longo dos anos e décadas. Como resultado, escrevi meu primeiro livro *Lights in the Tunel: Automation, Acelerating Technology and the Economy of the Future* que foi publicado em 2009.

Nesse livro, mesmo eu próprio escrevendo sobre a importância da tecnologia em aceleração, subestimei o quão rapidamente as coisas evoluíriam. Por exemplo, eu observei que fabricantes de automóveis estavam criando sistemas para evitar colisões e prevenir acidentes e sugeri que "ao longo do tempo esses sistemas podem evoluir numa tecnologia capaz de dirigir os carros sozinha". Bem, o que ocorreu é que o "ao longo do tempo" acabou não sendo tão longo tempo assim! Depois de um ano da publicação, o Google anunciou o primeiro carro capaz de se autodirigir no trânsito. E desde lá, três estados americanos – Nevada, Califórnia e Flórida – aprovaram leis que permitem que carros autônomos compartilhem as vias de forma limitada. Enquanto escrevo o governo britânico está à beira de publicar uma lei que permite que veículos autônomos sejam testados em ruas pela Inglaterra.

Também escrevi sobre o progresso sendo feito no campo da inteligência artificial. Na época, a história do computador "Deep Blue" da IBM e sobre como ele superou o campeão de xadrez Gary Kasparov em 1997, foi provavelmente a maior demonstração de uma inteligência artificial em ação. Novamente, fui pego de surpresa quando a IBM anunciou o sucessor do Deep Blue, o Watson – uma máquina que participou de um desafio muito mais desafiador, o show de televisão Jeopardy!

Xadrez é um jogo com um conjunto definido de regras, algo que se espera que o computador seja bom. Já o Jeopardy! é algo completamente diferente: um jogo que requer uma quantidade imensa de conhecimento, habilidade de entender uma linguagem, inclusive com piadas e ironias. O sucesso do Watson no Jeopardy! não é só impressionante, mas também

altamente prático e, de fato, a IBM já anunciou o uso do Watson em campos como a medicina e o atendimento ao consumidor.

Eu aposto que praticamente todos nós nos surpreenderemos com o progresso para os próximos anos ou décadas. Essas surpresas não estarão confinadas à natureza dos avanços tecnológicos por si só: o impacto que o progresso em aceleração tem na economia, como ele se equilibrará com o mercado de empregos está desafiando muito do conhecimento convencionado sobre como tecnologia e economia se misturam.

## Atividades

**1.** Cite quais componentes diferenciam a primeira, segunda, terceira e atual gerações de computadores.

**2.** O padrão de teclado DVORAK apresenta diversas vantagens sobre o padrão QWERTY. Sobre esses padrões é correto afirmar que:

a. O padrão DVORAK substituiu o QWERTY e está disponível em todos os teclados atuais.
b. O padrão QWERTY demonstrou que a consolidação de um padrão torna difícil a entrada de outro na indústria, mesmo que tecnicamente melhor.
c. O padrão DVORAK é nacionalizado, por isso, é mais fácil de produzir, já que atende à indústria local.
d. O padrão QWERTY foi criado para resolver o problema do teclado que enganchava os tipos gráficos ao utilizar o padrão DVORAK.

**3.** Qual é o nome da primeira pessoa a realizar programação de computadores na história:

a. Ada Lovelace.
b. Charles Babbage.
c. John von Neumann.
d. Alan Turing.

## Referências

CARVALHO, A. C. P. L. F.; LORENA, A. C. **Introdução à Computação**. São Paulo: LTC, 2016.

CHRISTIAN, B. C; GRIFFITHS, T. **Algoritmos para viver**: a ciência exata das decisões humanas. São Paulo: Companhia da Letras, 2017.

FONSECA FILHO, C. **História da computação**: o caminho do pensamento e da tecnologia. Porto Alegre: EDIPUCRS, 2007.

FORD, M. **Rise of the robots**. Nova York: Basic Books, 2015.

HANSEN, D. **Game On! Video Game History**: from Pong and Pac-man to Mario, Minecraft and More. [S.l.]: Ed. Feiwel and Friends, 2016.

HILTZIK, M. A. **Dealers of Lightining**: Xerox PARC and the dawn of the computer age. Nova York: Harper Collins, 2009.

ROCKETZ. **Evolução dos computadores nos últimos 10 anos**. Disponível em: <https://rocketz.com.br/store/articles/evolucao-dos-processadores-nos-ultimos-10-anos>. Acesso em: 8 dez. 2017.

WAZLAWICK, R. **História da Computação**. Rio de Janeiro: Elsevier, 2016.

## ☑ Resolução

a. Primeira geração: válvula. Segunda geração: transistor. Terceira geração: circuito integrado. Atual: microchips.
b. Alternativa "b". A prova disso está no fato de a proposta do Dvorak ser tecnicamente superior, mas não ter conseguido entrar no mercado.
c. Alternativa "a".

# 2
# A internet

Uma das tecnologias que mais transformaram o mundo nos últimos anos foi, definitivamente, o surgimento da rede mundial de computadores: a internet. Mais do que simplesmente uma tecnologia, ela democratizou o acesso à informação, permitiu o surgimento de novos negócios, criou novos empregos e conectou pessoas no mundo todo.

Neste capítulo, estudaremos essa rede, como ela surgiu e como mudou o mundo e o seu funcionamento técnico.

Vídeo

# 2.1 A história da internet

A comunicação digital é muito anterior à internet. Antes da rede mundial de computadores, já existiam redes de telégrafos e até mesmo de telefonia, permitindo comunicação global. Neste capítulo, vamos descobrir quem foram as pessoas e ideias que decidiram ligar computadores e dar início ao que chamamos de *rede mundial de computadores*.

## 2.1.1 O surgimento da Arpanet (1962-1972)

A comunicação digital é muito anterior à internet. Antes da rede mundial de computadores, já existiam redes de telégrafos e até mesmo de telefonia, permitindo comunicação global.

Os primeiros trabalhos que se tem notícia sobre interações sociais por redes de computadores são de agosto de 1962, quando Joseph Carl Robnett Licklider começou a discutir o conceito do que chamou de Galactic Network em suas pesquisas no Instituto de Tecnologia de Massachussets (MIT). Ele imaginou uma rede globalmente interconectada, que permitiria a qualquer um acessar programas ou dados, de qualquer lugar (HAFNER, 1999). Esse pesquisador tornou-se chefe da Agência de Projetos de Pesquisa Avançados de Defesa (Darpa), ligada ao departamento de defesa americano. Lá ele convenceu seus sucessores, Ivan Sutterland e Bob Taylor, da importância desse conceito.

Pouco tempo antes, em julho de 1961, Leonard Kleinrock publicou no MIT o primeiro artigo científico sobre a tecnologia de comutação de pacotes[1]. Também escreveu um livro que foi publicado em 1964. Foi então que convenceu Lawrence Roberts – arquiteto da Darpa – da viabilidade técnica ao usar pacotes em vez de circuitos, o que viria a ser um marco no conceito de redes de computadores (LEINER et al., 2017).

Em 1965, Roberts decidiu criar a primeira rede de computadores de longo alcance (PACKET, 2017). Com a ajuda de Thomas Merril, eles interligaram um computador TX-2 em Massachusetts com um Q-32 na Califórnia, usando uma conexão telefônica discada e de baixa velocidade. O resultado desse experimento provou que computadores poderiam trabalhar juntos, inclusive acessar dados e programas remotamente. Porém perceberam que a rede telefônica seria completamente inadequada para esse trabalho, o que reforçou ideia de Kleinrock, de que a troca por meio de pacotes de dados seria ideal (TANEMBAUM; WETHERALL, 2011).

No fim de 1966, Roberts foi contratado pelo Darpa para atuar como engenheiro-chefe (PACKET, 2017). Lá, começou seu trabalho e desenvolveu o plano de uma rede de computadores chamada *Arpanet*, publicando um artigo sobre ela em uma conferência, em 1967, promovida pela American Federation of Information Processing Societies (AFIPS)[2]. Nessa mesma conferência, outra rede baseada em pacotes foi apresentada pelos pesquisadores Donald

1 Método de envio de mensagens ou dados em pacotes de tamanho uniforme e de processamento e roteamento de pacotes em vez de fluxos de bits (SAWAYA, 1999, p. 339).

2 Organização que representava o interesse de diversas sociedades profissionais de informática. Foi extinta em 1990.

Davies e Roger Scantlebury, ambos do National Physical Laboratory (NPL). Scantlebury contou a Roberts sobre seus trabalhos, além de apresentar também o trabalho de Paul Baran, no instituto RAND, que trabalhava em uma rede utilizando pacotes para a transmissão de voz de maneira segura para o exército. Curiosamente, as pesquisas do MIT (1961-1967), RAND (1962-1965) e NPL (1964-1967) ocorreram paralelamente sem que os cientistas se conhecessem até então.

Em agosto de 1968, a estrutura geral e as especificações da Arpanet foram refinadas. Roberts deixou seu cargo e uma comunidade foi fundada para dar sequência ao projeto (HAFNER, 1999). Uma requisição de cotação (RFQ)[3] foi criada pelo Darpa para o desenvolvimento de um dos componentes-chave da rede: os roteadores de pacotes, na época chamados de *processadores de mensagens (IMPs)*. A empresa Bolt Beranek and Newman (BBN) ganhou a RFQ com um time liderado por Frank Heart, em dezembro de 1968.

Além da BBN, Bob Kahn trabalhou no design da arquitetura da rede. A topologia da rede e a parte econômica foram projetados e otimizados por Roberts, junto com Howard Frank e seu time na Network Analysis Corporation. Por fim, um sistema para medição da rede foi desenvolvido por Kleinrock na UCLA. Foi também a UCLA o local selecionado para ser o primeiro ponto de rede da Arpanet, iniciando as atividades em setembro de 1969, quando a BBN instalou o primeiro IMP e conectou o primeiro computador.

O segundo computador da rede foi o do Instituto de Pesquisa de Stanford (SRI), que trabalhava em um projeto de Douglas Engelbart. Além do mouse, Engelbart trabalhou em um projeto chamado *o aumento do intelecto humano*, que incluía um modelo de hipertexto conhecido como *on-line System (NLS)* (HILTZIK, 2009). O SRI também ficou responsável por incluir tabelas que mapeavam nomes de computadores aos seus endereços na rede, além de um diretório de requisições de comentários (RFCs), que se tornaram documentos padrão para sugerir mudanças na arquitetura da Arpanet e, futuramente, da internet.

Somente um mês depois, a primeira mensagem foi enviada do laboratório de Kleinrock ao SRI. Em seguida, dois outros computadores foram adicionados à rede: a Universidade de Santa Barbara e a de Utah. Ao fim de 1969, esses quatro computadores representavam as fundações do que viria a ser a internet. Mesmo nesse estágio inicial, os pesquisadores já se preocupavam tanto em trabalhar na infraestrutura da rede quanto em explicar como a rede deveria funcionar sobre essa infraestrutura – prática que foi mantida até a atualidade (WARD, 2015).

Foram adicionados vários computadores à rede nos anos seguintes, e o trabalho prosseguiu até a criação do primeiro protocolo ponto a ponto, em dezembro de 1970, pelo Network Working Group (NWG), liderado por Steve Crocker. O protocolo foi chamado de *Network Control Protocol (NCP)* e implementado entre 1971 e 1972, finalmente viabilizando aos usuários da rede criar suas próprias aplicações (HAFNER, 1999).

Em 1972, em uma conferência internacional de computação, chamada *ICCC*, Bob Khan realizou a primeira grande demonstração pública da Arpanet (LEINER et al., 2017). Foi também nesse ano que o correio eletrônico foi introduzido. Em março, Ray Tomlinson

3 Processo similar às licitações no Brasil.

criou o primeiro software de e-mail com as funcionalidades de leitura e envio. O software foi melhorado pelo próprio Roberts, que incluiu funcionalidades de arquivar, encaminhar e responder as mensagens. O e-mail foi a principal ferramenta da rede por mais de uma década, iniciando aí uma mudança de paradigma fundamental: a rede agora ligava pessoas a pessoas (WARD, 2015).

## 2.1.2 O surgimento da internet (1972-1973)

Em 1973, já havia uma série de redes de computadores operando separadamente nos EUA. Kahn, que havia se juntado ao Darpa, em 1972, começou a estudar uma forma de como interconectá-las. Embora a Arpanet fosse uma rede mundial, era pouco flexível, pois controlava muitos aspectos da comunicação e inviabilizava arquiteturas diferentes para situações específicas. Por exemplo, era muito difícil integrar uma rede de rádio à Arpanet, sem que sua arquitetura sofresse mudanças radicais (WARD, 2015).

A internet atual contém um conceito-chave que surgia naquela época: o de uma arquitetura aberta de rede (TANEMBAUM; WETHERALL, 2011). Nessa abordagem, a escolha de uma arquitetura de rede não é ditada por uma regra específica, mas escolhida livremente por um provedor e integrada a outras redes por uma meta chamada de *inter-rede*. Dessa forma, redes individuais podem ser projetadas e desenvolvidas para atender às necessidades específicas de seus usuários, não havendo qualquer requisito específico a respeito de seu tipo ou de sua geografia.

Originalmente, Kahn trabalhava em um conceito para comunicação em rádio, e percebeu que a chave para essa forma de comunicação funcionar era um protocolo confiável entre as duas pontas, que conseguia reduzir problemas na comunicação na presença de ruídos ou falhas de sinal (WARD, 2015). Foi então que ele iniciou a concepção de um protocolo para transmissão de pacotes apenas para a rede de rádio, evitando que tivesse de trabalhar com vários sistemas operacionais e permitindo que continuasse a usar a NCP.

Entretanto, Kahn percebeu que a NCP não possuía uma forma de enviar uma mensagem a uma rede ou máquina que estivesse ligada a um IMP, portanto, precisava ser alterada. Além disso, o NCP também contava que a infraestrutura da Arpanet garantiria a confiabilidade da transmissão de dados; se uma perda de pacotes ocorresse, tanto as aplicações quanto a própria NCP falhariam. Ou seja, partia-se do pressuposto que na Arpanet o canal de comunicação era extremamente confiável, uma situação muito diferente do que Kahn percebia nas redes de rádio.

Por tudo isso, Kahn decidiu desenvolver uma nova versão do protocolo que permitisse uma arquitetura de rede aberta. Posteriormente, esse protocolo tornou-se o *Transmission Control Protocol/Internet Protocol* (TCP/IP), definindo, de acordo com Leiner et al. (2017), as primeiras regras de conexão:

- Cada rede separada deve se manter por si mesma e nenhuma mudança interna deve ser feita para conectar-se à internet.
- As comunicações serão feitas com base no melhor esforço. Se um pacote não chegar ao seu destino final, ele deve ser retransmitido pela origem.
- Caixas pretas serão utilizadas para conectar as redes. Posteriormente, essas caixas foram chamadas de *gateways* e de *roteadores*. Não há nenhuma informação retida por esses equipamentos sobre o fluxo de dados que passa por eles, para que se mantenham simples e evitem a necessidade de criar mecanismos de prevenção de falhas complexas.
- Não haverá controle global no nível de operações.

No início de 1973, Kahn solicitou a ajuda de Vint Cerf para trabalhar no detalhamento do projeto do protocolo. Cerf atuou diretamente no design e desenvolvimento do NCP original e já possuía conhecimento relacionado a interfaces com diferentes sistemas operacionais (LEINER et al., 2017).

A união dos dois mostrou-se extremamente produtiva, e a primeira versão do protocolo foi distribuída em uma reunião especial do International Networking Group (INWG), realizada na universidade de Sussex, em setembro de 1973. Cerf havia sido convidado para coordenar esse evento e utilizou-se disso para convocar uma reunião com os membros do INWG que estavam fortemente presentes na conferência (WARD, 2015).

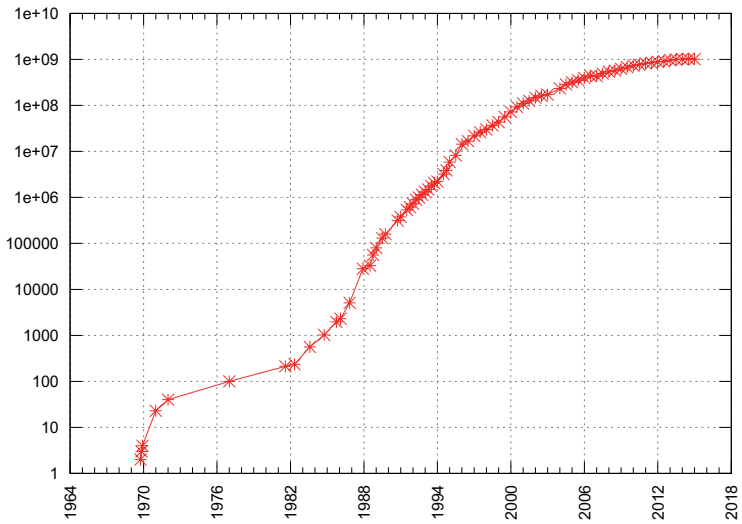
O papeis originais de Cerf e Kahn sobre a internet descreviam um protocolo, chamado *TCP*, que forneceu todos os serviços de transporte e qualidade na internet. A ideia original de Kahn era que esse protocolo suportasse uma série de serviços de transporte, desde uma sequência confiável de dados (como um circuito virtual) até uma sequência de baixa confiabilidade, em que pacotes poderiam se perder ou trocar de ordem. Entretanto a primeira implementação do protocolo incluiu apenas os circuitos virtuais, o que funcionou bem para transferência de arquivos e para aplicações de login.

A fragilidade dessa abordagem ficou evidente ao surgirem aplicações de transmissão de voz, deixando claro que Kahn estava certo, isto é, era necessária uma versão do protocolo no qual a perda de pacotes pudesse ser tratada pelo aplicativo, e não pelo protocolo em si. Para esse tipo de aplicação foi desenvolvido um segundo protocolo, chamado de User Dataquam Protocol (UDP) (WARD, 2015).

## 2.1.3 Outros fatos importantes

Várias redes de computadores começaram a se unir à internet. A partir dos anos 1980, o tamanho da rede se multiplicaria por 10 a cada intervalo de aproximadamente cinco anos. No gráfico a seguir (Figura 1), pode-se perceber esse crescimento assustador:

**Figura 1 – Número de hosts mundialmente ligados à internet.**




Fonte: Wikimedia Commons.

Nesse período, a maior parte das inovações ocorreram pelos próprios usuários da rede. Destacamos algumas em ordem cronológica (WAZLAWICK, 2016):

- 1973-1975 – Robert Metcalfe, no laboratório Xerox PARC, em Palo Alto, desenvolveu um sistema que substitui a transmissão de rádio por transmissão a cabo de forma extremamente eficiente. Robert deixou a companhia para fundar a 3com e batizou esse sistema de *Ethernet*.
- 1979 – Tom Truscott e Steve Bellovin criam um programa em Unix para transferir arquivos e notícias entre computadores em rede, com a finalidade de trocar informações entre as universidades de Duke e da Carolina do Norte por meio de rede discada. O sistema cresceu e ficou conhecido como *Usenet*.
- 1982-1983 – Arpanet foi dividida em duas: Milnet, para uso militar, e a Arpanet, para uso primariamente acadêmico. Isso permitiu que equipamentos de internet fossem comercializados e criou a necessidade de associar endereços de redes a nomes mais intuitivos, dando origem ao protocolo DNS (Domain Name System). Surgiram então os domínios .com (comercial), .org (organizações), .gov (governo) .edu (educacional) e .mil (militar).
- 1985 – O primeiro domínio comercial é registrado.
- 1990 – A Arpanet é considerada obsoleta. Tim Berners-Lee desenvolve o primeiro browser de internet, chamado *World Wide Web (WWW)*. Também faz a primeira comunicação utilizando o protocolo HTTP, dando origem à WWW como é conhecida atualmente.
- 1992 – Tim Berners-Lee cria o protótipo do HTML, protocolo que se tornaria o padrão para exibir texto na internet.
- 1993 – Surge o primeiro serviço legal de download de MP3, chamado *Internet Underground Music Archive* (IUMA).

- 1994 – O Yahoo surge como o primeiro mecanismo de busca da internet. No mesmo ano, Jeff Bezzos cria os planos para a Amazon.com.
- 1995 – Surgem o e-Bay, Internet Explorer, a Amazon e Java, considerada a primeira linguagem de programação voltada para a internet. Também é feito o primeiro stream de vídeo pela ESPN Sports Zone.
- 1996 – Surge o primeiro jogo multijogador massivo (MMO) para internet, chamado *Ultima Online*, responsável por popularizar o gênero. Seu lançamento também promove a adoção em massa da internet na indústria de jogos.
- 1997 – Surge o Mosaic, da NCSA, o primeiro navegador gráfico da história. Considerado um dos responsáveis pela rápida expansão da internet. Seus criadores, posteriormente, foram contratados pela Netscape, dando origem ao Netscape Navigator.
- 1998 – Surge o Google, que rapidamente se torna o buscador mais usado da internet.
- 1999 – Inicia-se a polêmica sobre pirataria na rede devido ao software *Napster*, criado por Shawn Fanning. Também surge o primeiro vírus capaz de se autocopiar e se redistribuir por e-mail.
- 2000 – A editora Boomis, de Jimmy Walles, lança a Nupedia, uma enciclopédia livre para a língua inglesa, cujo editor era Larry Sanger. Porém, devido ao sucesso da Wikipédia, fundada em 15 de janeiro de 2001, decidiu encerrar suas atividades.

Vídeo

## 2.2 Como a internet funciona

Na seção anterior, vimos que uma das grandes chaves para seu desenvolvimento da internet foi a ideia de **trocar pacotes de dados**. Agora, iremos entender exatamente o que são esses pacotes e como eles navegam pela rede.

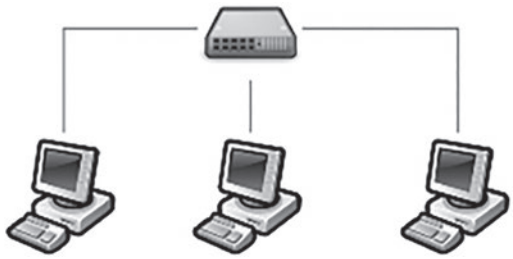
### 2.2.1 Redes Ethernet

Antes de estudarmos a internet, vamos discutir a respeito de uma rede mais simples, a Ethernet. Tratam-se das redes com cabos, comuns em escritórios e em boa parte das residências.

Nessas redes, as máquinas ligam-se entre si por um fio. Em cada computador existe um circuito de saída, ligado ao circuito de entrada de outro computador da rede, fisicamente. Quando um computador deseja enviar uma mensagem, ele simplesmente energiza esse circuito e esse sinal é recebido em outro computador. Como em uma ligação direta, os cabos de entrada e saída cruzaram-se e ganharam o nome de *cabos crossover* ou simplesmente *cabo cross.*

Para ligar vários computadores, utiliza-se um equipamento de rede conhecido como *hub* (TANEMBAUM; WETHERALL, 2011). Um hub têm várias portas de entrada. Esse equipamento recebe um sinal por essa porta, amplifica-o e distribui para todas as portas de saída. Dessa forma, nenhum cabo ligado a ele precisa ser cruzado. O hub, portanto, simula que várias máquinas estejam ligadas pelo mesmo cabo (Figura 2).

**Figura 2 – Rede ethernet.**



Fonte: Elaborada pelo autor.

Para cada computador é associado um endereço, chamado *endereço MAC* (Media Access Control). Quando um computador se comunica, todos os outros recebem o sinal, mas como cada placa de rede tem um endereço MAC único, ela é capaz de filtrar só as mensagens que são para si.

O que acontece se dois computadores enviarem informação ao mesmo tempo? Nesse caso, a energia dos dois sinais irá se somar. Quem detecta esse fato é a placa de rede, que automaticamente sorteia um intervalo de tempo para então retransmitir. Esse **protocolo de funcionamento** é chamado de *CSMA/CD (Carrier Sense Multiple Access/Collission Detection)*. Um **protocolo de comunicação**, portanto, nada mais é do que o conjunto de todas as regras e procedimentos necessários para que ocorra a comunicação (TANEMBAUM; WETHERALL, 2011).

Perceba que nesse modelo é a placa de rede a grande responsável pela comunicação. Ela dispõe dos sensores para detectar se um sinal chegou, se o emissor enviou um sinal, e, acima de tudo, é ela que faz o **controle de qualidade** do canal de comunicação.

Por exemplo, ligar uma placa de rede ethernet a uma rede wireless é um trabalho complexo. Afinal, são tecnologias totalmente diferentes, por isso, seriam necessários conversores.

Imagine que temos uma terceira rede, digamos, de rádio. Já teríamos de ter conversores ethernet/wireless, ethernet/rádio, rádio/wireless, rádio/ethernet. E se tivéssemos centenas de tipos de redes? Teríamos milhares de tipos de conversores, o que torna o sistema todo inviável.

Este foi o dilema enfrentado por Kahn e Cerf, em 1973, que levou ao surgimento do TCP/IP (Transmission Control Protocol/Internet Protocol) (LEINER et al., 2017).

## 2.2.2 Os protocolos da internet

Na internet, em vez de se trabalhar com sinais e circuitos, trabalha-se com pacotes de dados. São eles que controlam:

- quem enviou a informação;
- para onde ela vai;
- o quanto ela já viajou;
- dígitos verificadores, para saber se ela estragou.

Assim, os circuitos são responsáveis apenas por encaminhar os dados de um ponto a outro da rede, mas são os computadores ligados a essas redes os responsáveis por garantir a qualidade da transmissão (TANEMBAUM; WETHERALL, 2011).

Cada máquina ligada à internet tem um **endereço de rede**, também conhecido como *endereço IP*. Por exemplo, o servidor do Facebook tem o endereço 157.240.12.35.

Agora, vamos supor que você possua duas máquinas ligadas à internet e deseja enviar um e-mail da máquina A para a B. O primeiro passo será dado pelo software do e-mail, que irá selecionar o texto e as informações pertinentes (como destinatário, assunto etc.) e codificar em um formato que outro software de e-mail entenda. Isso é chamado de *camada da aplicação*, já que não está relacionado com a transmissão da mensagem em si, mas sim com informações que só interessam aos aplicativos de e-mail (TANEMBAUM; WETHERALL, 2011).

Em seguida, são adicionadas informações de transporte pelo protocolo TCP. Ou seja, qual é a porta de saída desse dado e informações como um checksum[4] da mensagem. Nessa etapa, a mensagem também pode ser dividida em vários pacotes de dados menores, onde serão adicionadas informações, como a ordem desses pacotes.

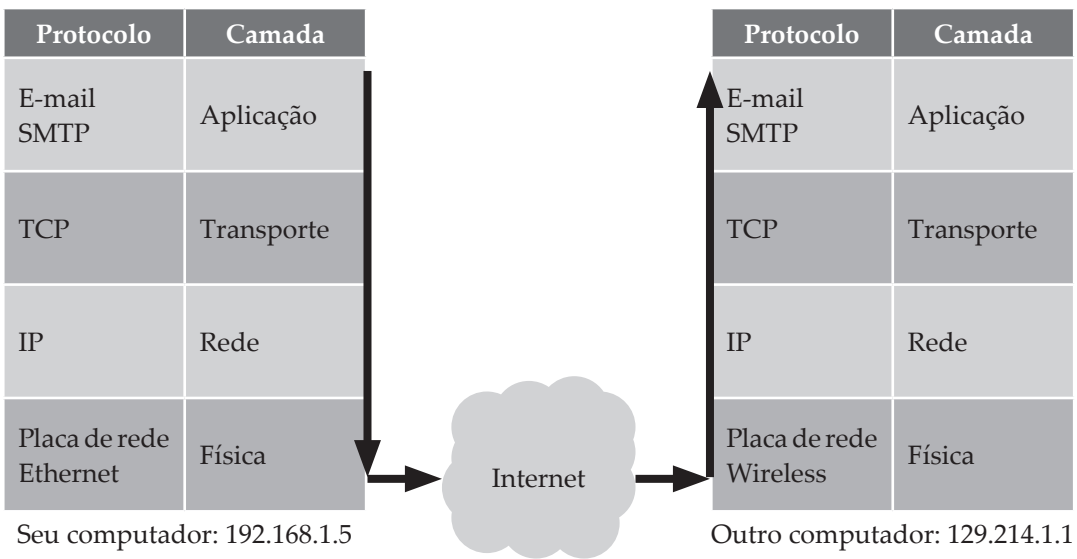
A porta de saída nada mais é do que um número, que identificará a aplicação dos dois lados do canal. O controle desses dados ocorre no que chamamos de *camada de transporte*.

Em seguida, são adicionadas informações de roteamento. Ou seja, o endereço da máquina de destino, um contador de quantas vezes aquele pacote já saltou de uma máquina para outra etc. Toda essa comunicação é então gerenciada pela **camada de rede**.

Por fim, em algum momento esses dados precisam ser transmitidos por um meio físico, seja uma rede de cabos, wireless ou satélite. Quem faz a conversão e transmissão final desses dados é o hardware da **camada física** (TANEMBAUM; WETHERALL, 2011).

Quando a informação chega em outro computador, ela toma o sentido inverso.

**Figura 3 – Protocolos para o transporte de dados.**




Fonte: Elaborada pelo autor.

4 "Cálculo da soma ou total de verificação. É um dígito ou uma série de dígitos calculados do bloco de sucessivos caracteres. O dígito é calculado na mesma hora em que o dado é armazenado e finalizado" (SAWAYA, 1999, p. 77).

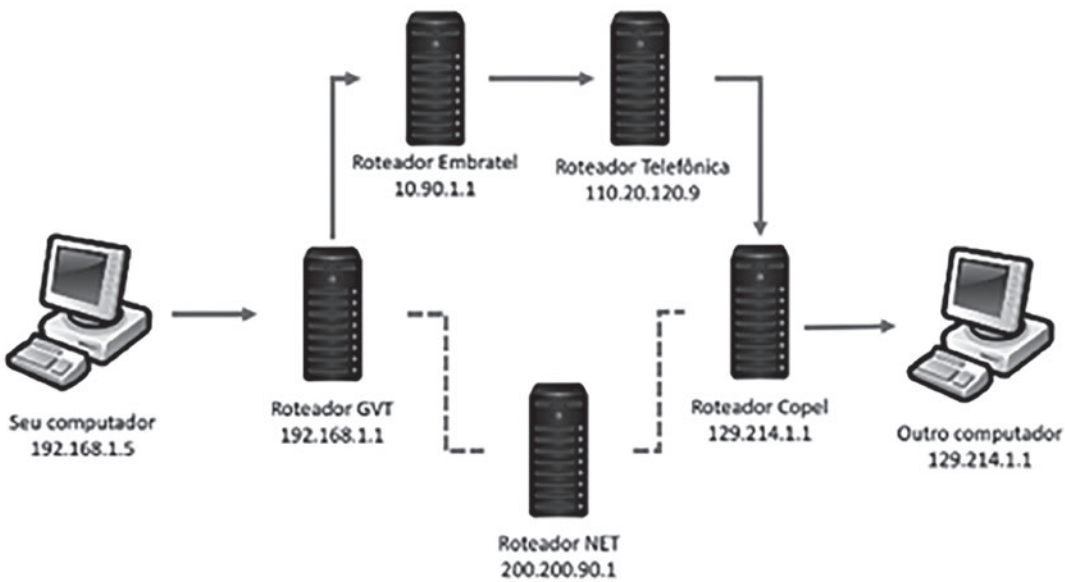
## 2.2.3 Entendendo a nuvem

Já entendemos por quais camadas lógicas os dados são submetidos para a comunicação ocorrer, mas o que exatamente é a nuvem no diagrama anterior (Figura 3)? Vejamos.

O que faz a comunicação efetivamente é um equipamento chamado *roteador* e conecta duas ou mais redes. Cada aparelho tem uma tabela e políticas de roteamento (TANEMBAUM; WETHERALL, 2011).

Quando uma mensagem chega, o roteador verifica se o destino é uma das redes diretamente ligadas a ele. Caso seja, ele encaminha a mensagem para essa rede. Caso contrário, ele encaminha para outro roteador, na esperança de que este conheça a máquina de destino. A escolha desse novo roteador é feita com base em várias diretivas: o quão próximo os dois endereços de rede são, a frequência que esse roteador consegue encaminhar mensagens ou, até mesmo, políticas de qualidade. O interessante é que essas regras podem ser recalculadas e, em duas comunicações diferentes, dois pacotes podem pegar caminhos completamente distintos. Então, nosso diagrama pode ser redesenhado assim:

**Figura 4 – Detalhamento da nuvem.**




Fonte: Elaborada pelo autor.

Na Figura 4, as linhas tracejadas representam uma rota não utilizada, passando pelo roteador da Net. Nada impede que um segundo e-mail tome essa rota. Como a internet é formada por centenas de milhares de roteadores ligados entre si, utilizamos esse desenho da nuvem para representar toda essa malha.

## 2.2.4 Firewall e NATs

Nem todos os computadores estão diretamente ligados à internet. Seu computador de casa estará ligado a um **provedor de internet (ISP)**. Este tem o endereço de internet válido.

Atrás dele, está uma rede com todos os computadores ligados ao provedor, com endereços de rede internos e não visíveis na internet.

Um dos equipamentos do provedor, chamado *NAT (Network Address Translator)* é responsável por essa tradução de endereços. Por isso, é quase impossível que alguém de fora da rede acesse sua máquina diretamente (TANEMBAUM; WETHERALL, 2011).

Você também já deve ter ouvido falar de firewall. Trata-se de um software que filtra quais pacotes de rede podem ou não trafegar. Ele analisa a comunicação e pode barrar conteúdo impróprio, bloquear ataques à rede, entre outros.

Vídeo

## 2.3 A internet em nossas vidas

O impacto da internet foi extremamente profundo, trouxe mudanças significativas tanto na área social, econômica, quanto colaborativa e política. Nesta seção, exploraremos parte desse impacto.

### 2.3.1 O impacto social

A internet possibilitou muita flexibilidade nas horas de trabalho e colaboração. Atualmente, é possível escrever documentos de maneira compartilhada e colaborativa em ferramentas como o Google Docs.

Podemos nos comunicar com pessoas em escritórios remotos por chats em ferramentas como o Skype. Para as companhias telefônicas, modelos de cobrança baseado em ligações a distância tornaram-se obsoletos, uma vez que a comunicação na internet independe desse fator.

Foi graças à internet que repositórios de software globais permitiram que desenvolvedores do mundo todo criassem programas de computador gratuitos, de forma livre, viabilizando uma indústria de software livre e, na maior parte das vezes, gratuito. Exemplos desses programas são o Linux, OpenOffice e o Firefox.

A internet também permitiu a criação de redes sociais. Nelas, pessoas podem trocar informações, fotos ou mesmo se divertir. São exemplos dessas redes fóruns técnicos, como o StackOverflow ou o GUJ, redes puramente sociais, como o Facebook e até redes para troca de currículos, como o LinkedIn.

Outro impacto significativo foi o fato de que a internet distribuiu largamente o acesso à informação. Embora estima-se que 51% do conteúdo da internet ainda esteja em língua inglesa (W3TECHS, 2017), não se pode negar o impacto que o motor de busca do Google ou a Wikipédia trouxeram para a educação. Estima-se que nada menos que 70% das pesquisas sobre qualquer tema se iniciem na Wikipédia (W3TECHS, 2017).

Na atualidade, a maioria das pessoas sabe que pode se informar sobre qualquer conteúdo utilizando a internet. A disponibilidade da informação em vídeos viabilizou até mesmo outros modelos educacionais, como a sala de aula invertida, na qual o professor sugere aos alunos que estudem o conteúdo em vídeos e textos na internet em casa, utilizando a aula somente para desenvolver trabalhos e atividades.

Outro exemplo de modelo educacional viabilizado é a Educação a Distância (EaD). O censo de Educação Superior no Brasil, realizado em 2013 (BRASIL, 2015), apontou que o número de matrículas em cursos a distância no Ensino Superior brasileiro ultrapassa a casa de um milhão. Esse Censo também apontou que a EaD já seria responsável por 15,8% dos cursos feitos no Brasil. Uma graduação a distância também pode custar até 60% menos, o que aumenta a democratização do ensino, mesmo que sua distribuição seja majoritariamente feita por empresas privadas.

## 2.3.2 Impacto econômico

Um dos impactos mais significativos foi o econômico. Em 2012, as vendas no e-commerce ultrapassaram a marca de 1 trilhão de dólares. Em 2013, a República Tcheca foi a economia em que o e-commerce teve a maior contribuição para as margens de lucro das empresas, com 24% das transações ocorrendo por esse canal (EUROPEAN COMMISSION, 2017). No Brasil, o crescimento do e-commerce ocorre em velocidade espantosa e ultrapassou a marca de 24%, em 2014, e 15%, em 2015 (E-COMMERCE..., 2017).

A internet viabilizou também que as vendas pudessem ocorrer on-line nos smartphones. O uso de telefones atualmente já representa 25% do total das vendas (E-COMMERCE..., 2017).

Vários setores foram remodelados pela tecnologia. Nos setores da música e dos vídeos, por exemplo, vendia-se um produto físico, como um CD ou DVD. Com a internet, os modelos de streaming alteraram a percepção de valor dos consumidores. Não se tratava mais de querer um produto físico, mas somente a experiência de assistir a um vídeo ou ouvir uma música quando quiser.

Os modelos de negócios mudaram de lojas físicas para serviços, abrindo espaço para empresas como a Netflix e o Spotify. Tudo isso acompanhado uma baixa significativa dos preços desses produtos.

Além disso, a internet agora facilita a distribuição desse conteúdo. Se antigamente um pequeno músico deveria pagar taxas exorbitantes para ter seu trabalho divulgado, agora encontra poucas barreiras em ferramentas como Spotify ou Apple Music, ou pode gratuitamente divulgar-se em um canal.

Transformação similar ocorreu no mercado de jogos e aplicativos. CDs e DVDs deram lugar a lojas como a App Store, Play Store ou Steam. Fabricantes de videogames embutem em seus produtos as próprias lojas on-line.

A compra do aplicativo foi substituída por modelos de licenciamento por usuário ou por tempo. Viabilizou-se a criação de licenças flutuantes, nas quais uma pessoa pode instalar o software em quantas máquinas quiser, mas usá-la apenas em uma máquina por vez. Além disso, aplicativos e games agora permitem que conteúdo adicional seja comprado à parte.

Da mesma forma, as lojas tornaram-se canais de distribuição fáceis para desenvolvedores. Qualquer programador, mesmo sozinho, pode facilmente disponibilizar seu software nessas lojas. Isso permitiu, por exemplo, que o brasileiro Michel Krieger e seu amigo Kevin

Systrom enriquecessem lançando – sem qualquer modelo de negócio ou maior pretensão – o aplicativo Instagram, que posteriormente foi vendido ao Facebook (WAZLAWICK, 2016).

Provavelmente, o impacto mais significativo seja o da economia colaborativa (PAULSON, 2016). A internet uniu pessoas interessadas em fazer negócios. Aplicativos como a Uber conectam um motorista particular a um passageiro; o Airbnb une alguém interessado em alugar uma casa a um inquilino; o Mercado Livre e a OLX unem pessoas interessadas em comprar e vender.

Nesse modelo, uma companhia age apenas como intermediária, e não como prestadora final do serviço. Ou seja, é fornecido por pessoas, o que tem deixado governos com incertezas quanto a esse modelo de negócios.

Essa mudança tem gerado impacto até em legislações, gerando discussões como: afinal, o motorista do Uber é um trabalhador autônomo ou um funcionário da empresa? A Uber é uma empresa de transportes ou uma empresa de software?

Outra forma de economia colaborativa interessante é o **crowdfunding** (PAULSON, 2016). Plataformas como o KickStarter e o Catarse permitem que pessoas invistam dinheiro em ideias que lhes agradam. Entretanto, como a internet permite uma abrangência enorme, muitos projetos chegam a receber milhões. Um exemplo disso é o filme *Star Trek: Axanar*, que não é oficial da franquia, mas uma produção de fãs, não licenciada pelos estúdios detentores dos direitos autorais (CBS/Paramount). O filme, porém, conseguiu mais de 1 milhão de dólares em plataformas de crowdfunding, o que chamou a atenção dos estúdios, iniciando uma guerra judicial entre a CBS/Paramout e os produtores do filme (HURST, 2016).

## 2.3.3 Internet na política e nos governos

A internet causou impactos significativos na política e nos governos. Em 2004, o presidenciável norte-americano Howard Dean conseguiu sucesso notável ao financiar sua campanha por meio de crowdfunding.

Para a política, o ativismo virtual é uma das principais transformações ocorridas na internet. Infelizmente, boa parte desse ativismo está na criação de *pós-verdades* – notícias falsas, parcialmente falsas ou fora de contexto com o intuito de gerar uma resposta mais emocional do que realmente informar.

Outro benefício, por exemplo, é a oportunidade encontrada pelos partidos menores para sua própria divulgação. Cabe lembrar que outros canais, como a imprensa, são altamente regulados e tanto o momento quanto a duração das campanhas devem seguir regras rígidas e definidas na constituição. Na internet, essas regras são mais flexíveis e dispõem de vários dos meios de divulgação, como a página do partido no Facebook.

Os governos também sofreram impacto significativo (WAZLAWICK, 2016). É possível acompanhar a reação da população a medidas tomadas, pesquisar propostas (como faz o senado federal) ou mesmo fazer marketing.

O surgimento de plataformas como o <colab.re> permitiu que pessoas divulgassem problemas da cidade diretamente para a prefeitura, ajudando de forma voluntária na fiscalização.

Finalmente, o próprio governo tornou-se muito mais fiscalizável. Por meio de portais de transparência, dados e gastos podem ser acompanhados. Isso permite que qualquer cidadão comum fiscalize o governo ou uma empresa pública. Um exemplo disso aconteceu no Paraná, onde uma estudante descobriu que bolsas da UFPR estavam sofrendo desvios e, após diversas pesquisas, pôde encaminhar os dados para auditoria, para verificar onde o dinheiro estava indo (BREMBATTI; MARCHIORI, 2017).

Esses são somente alguns impactos que a rede trouxe para o mundo. Porém, há algo de comum em todos os cenários: a internet trouxe dezenas de oportunidades. Por isso, é importante estar atento a elas.

## Ampliando seus conhecimentos

Como leitura complementar, indicamos esta notícia do portal EBC, sobre a regulamentação da internet no Brasil, o Marco Civil.

### Entenda o Marco Civil da Internet ponto a ponto

*(ENTENDA..., 2016)*

[...]

[A atual Lei n. 12.965, de 23 de abril de 2014, conhecida] popularmente como o *Marco Civil da Internet*, é uma espécie de "constituição" [...] definindo direitos e deveres de usuários e provedores da web no país. [...] Após quase três anos de tramitação na Câmara, o plenário da Casa aprovou o projeto. [...]

#### Conheça os principais pontos do Marco Civil

**Neutralidade na rede**

O princípio da neutralidade diz que a rede deve ser igual para todos, sem diferença quanto ao tipo de uso. Assim, ao comprar um plano de internet, o usuário paga somente pela velocidade contratada e não pelo tipo de página que vai acessar.

[...]

**Privacidade na web**

Além de criar um ponto de referência sobre a web no Brasil, o Marco prevê a inviolabilidade e sigilo de suas comunicações. O projeto de lei regula o

monitoramento, filtro, análise e fiscalização de conteúdo para garantir o direito à privacidade. Somente por meio de ordens judiciais para fins de investigação criminal será possível ter acesso a esses conteúdos.

[...]

**Logs ou registros de acessos**

Segundo o Marco Civil, os provedores de conexão são proibidos de guardar os registros de acesso a aplicações de internet. Ou seja, o seu rastro digital em sites, blogs, fóruns e redes sociais não ficará armazenado pela empresa que fornece o acesso. Mas, pelo artigo 15 do PL, toda empresa constituída juridicamente no Brasil (classificada como provedora de aplicação) deverá manter o registro desse traço por seis meses.

[...]

**Data centers fora do Brasil**

O relator do projeto retirou do texto a exigência de data centers no Brasil para armazenamento de dados. Um data center é uma central de computadores com grande capacidade de armazenamento e processamento de dados onde ficam, normalmente, os arquivos dos sites, e-mails e os logs de acesso. [...]

## Atividades

**1.** Associe o protocolo à sua respectiva camada:

( ) FTP (Transferência de arquivos)
( ) IP
( ) Protocolo de redes satélite
( ) Ethernet
( ) TCP
( ) Torrent

1. Camada de aplicação
2. Camada de transporte
3. Camada de rede
4. Camada física (hardware)

2. Para que serve o comando ping, comumente utilizado por administradores de rede e citado por fãs de games? Pesquise e responda.

3. Pesquise na internet o endereço IP dos seguintes sites: <google.com>, <nasa.gov>, <microsoft.com> e do <planalto.gov.br>. Esse exercício também pode ser resolvido com o comando ping, conforme pesquisado na atividade anterior.

## Referências


BRASIL. Ministério da Educação. Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira. **Censo da educação superior 2013**: resumo técnico. Brasília, DF, 2015. Disponível em: <http://download.inep.gov.br/download/superior/censo/2013/resumo_tecnico_censo_educacao_superior_2013.pdf>. Acesso em: 11 dez. 2017.

BREMBATTI, K.; MARCHIORI, R. Estudante detectou sozinha desvio milionário de bolsas da UFPR. **Gazeta do Povo**, 22 fev. 2017. Disponível em: <http://www.gazetadopovo.com.br/vida-e-cidadania/estudante-detectou-sozinha-desvio-milionario-de-bolsas-que-a-ufpr-nao-viu-52c7c52x896li4rb2qkrjeona>. Acesso em: 11 dez. 2017.

E-COMMERCE brasileiro deve crescer até 15% em 2017. **E-commerce Brasil**, 12 jan. 2017. Disponível em: <https://www.ecommercebrasil.com.br/noticias/e-commerce-brasileiro-crescer-15-2017/>. Acesso em: 11 dez. 2017.

ENTENDA o Marco Civil da Internet ponto a ponto. **EBC**, 22 abr. 2016. Disponível em: <http://www.ebc.com.br/tecnologia/2014/04/entenda-o-marco-civil-da-internet-ponto-a-ponto>. Acesso em: 11 dez. 2017.

EUROPEAN COMMISSION. **The Digital Economy and Society Index (DESI)**: Czech Republic, 2017. Disponível em: <http://ec.europa.eu/newsroom/document.cfm?doc_id=42999>. Acesso em: 11 dez. 2017.

HAFNER, K. **Where Wizards Stay Up Late**: The Origins Of The Internet. Nova York: Simon & Schuster, 1999.

HILTZIK, M. A. **Dealers of Lightining**: Xerox PARC and the dawn of the computer age. Nova York: Harper Collins, 2009.

HURST, S. Creators of Crowdfunding Success "Star Trek: Axanar" Unveils Trailers Midst Lawsuit (Video). **Crowdfund Insider**, 23 jun. 2016. Disponível em: <https://www.crowdfundinsider.com/2016/06/87198-creators-crowdfunding-success-star-trek-axanar-unveils-trailers-midst-lawsuit-video/>. Acesso em: 11 dez. 2017.

LEINER, B. M et al. **Brief History of Internet**: 1997. [S.l.]: Internet Society, 2017. Disponível em: <https://www.internetsociety.org/wp-content/uploads/2017/09/ISOC-History-of-the-Internet_1997.pdf>. Acesso em: 11 dez. 2017.

PACKET. **Dr. Lawrece Roberts**. About. Disponível em: <http://packet.cc/about.html>. Acesso em: 11 dez. 2017.

PAULSON, M. **Online Business from Scratch**. [S.l.]: CreateSpace Independent Publishing Platform, 2016.

SAWAYA, M. R. **Dicionário de informática e internet**. São Paulo: Nobel, 1999.

TANEMBAUM, A. S.; WETHERALL, D. **Redes de computadores**. 5. ed. São Paulo: Pearson, 2011.

W3TECHS. **Usage of content languages for websites**. Technologies. Disponível em: <https://w3techs.com/technologies/overview/content_language/all>. Acesso em: 11 dez. 2017.

WARD, B. **History of Internet**. [S.l.]: Amazon, 2015.

WAZLAWICK, R. **História da computação**. Rio de Janeiro: Elsevier, 2016.

## ☑ Resolução

**1.**

| | |
|---|---|
| ( 1 ) FTP (Transferência de arquivos) | 1. Camada de aplicação |
| ( 3 ) IP | 2. Camada de transporte |
| ( 4 ) Protocolo de redes satélite | 3. Camada de rede |
| ( 4 ) Ethernet | 4. Camada física (hardware) |
| ( 2 ) TCP | |
| ( 1 ) Torrent | |

**2.** O comando ping mede o tempo que um pacote de rede leva para chegar até o destino. É possível dar esse comando entrando no *prompt de comando* do computador e digitando ping e o endereço de rede desejado. Por exemplo: ping google.com.

**3.** <google.com>: 216.58.222.46, <nasa.gov>: 52.0.14.116, <microsoft.com>: 23.100.122.175, <planalto.gov.br>: 170.246.252.9.

# 3
# Mercado de informática

Com a evolução da internet, surgiram diferentes áreas de atuação para atender às demandas das empresas. Conforme a informática evolui, novos problemas aparecem. Por isso, é preciso profissionais capacitados que saibam resolvê-los.

Neste capítulo, vamos estudar as diferentes áreas do mercado da informática e entender um pouco a respeito da especialização de cada uma delas.

Vídeo

# 3.1 Desenvolvimento

O desenvolvimento de softwares é a área mais óbvia quando se pensa no mercado de informática. Por isso, começaremos descrevendo-a.

Na programação, existem diferentes profissões ou papéis que o profissional poderá atuar (VOCÊ..., 2016):

- Analista de sistemas – está preocupado em levantar os requisitos, entender a demanda dos clientes e comunicá-las ao time de desenvolvimento.
- Programador – codifica o software.
- Database administrator (DBA) – administra o banco de dados onde as informações são armazenadas.
- Equipe de testes e qualidade – testa e verifica se o sistema está de acordo com o planejado.

Entretanto, o mercado de programação divide-se em cinco grandes ramos. Programação web, programação para desktop, programação para dispositivos móveis, programação de jogos e área de controle e automação.

## 3.1.1 Programação web

Existem diversos tipos de programador web no mercado, mas todos os tipos são relacionados com o ambiente da internet. Os principais são os programadores de **front-end** e **back-end** (MARTINS, 2009).

O programador front-end é responsável por toda a parte visual da aplicação web. Trabalha muito em contato com a equipe de design, pois é dever dele ajustar todo o visual conforme planejado pelos designers (SCUDERO, 2017). As linguagens utilizadas são HTML, CSS e javascript, também é necessário conhecimento nas bibliotecas AngularJS, Node.js, jQuery, Laravel, Ionic, entre outras.

Já o programador back-end trabalha mais com a parte funcional do sistema, ou seja, com a programação do servidor web, onde é feito todo o processamento de dados. É preciso muito cuidado para atuar nessa área, visto que toda a informação da aplicação é tratada nessa parte do programa.

As linguagens de programação comumente usadas nesse campo são o Java, PHP, C#. As linguagens Python, Ruby e JavaScript também estão se destacando devido à facilidade na hora de programar e às tecnologias, como o servidor Node.js e o Rails.

## 3.1.2 Programação para desktop

A aplicação desktop é aquela que pode ser instalada e executada no computador. O objetivo dela é aumentar a capacidade de recursos disponibilizados pelo sistema operacional, proporcionando uma experiência mais completa para cada tipo de usuário.

Para aqueles que desejam trabalhar nessa área, é necessário ter conhecimento em sistemas operacionais de computadores **Windows**, **Linux** ou **Macintosh** e fluência em alguma das seguintes linguagens de programação: Pascal, C, C++, VB, C# e VB.NET.

Vale ressaltar que, com a grande expansão da internet, muitas aplicações estão migrando para plataformas on-line por proporcionar maior mobilidade e manutenção facilitada (COMPUTER SCIENCE ONLINE, 2017). Por conta disso, esse mercado diminuiu de tamanho no Brasil e no mundo. Atualmente, encontramos muitos sistemas legados, desenvolvidos em linguagens como Delphi e em versões antigas do Visual Basic.

## 3.1.3 Dispositivos móveis

Na atualidade, quase todas as pessoas possuem smartphone e estão o tempo todo conectadas a aplicativos. Devido a essa necessidade, foi preciso uma ramificação específica para o desenvolvimento de aplicativos (SCUDERO, 2017).

Bancos, entrega de comida, compras no exterior, redes sociais, locomoção e jogos são alguns dos tipos de programas existentes para celular. O programador mobile tem de se adaptar conforme a exigência da aplicação. As linguagens utilizadas são aquelas orientadas a objetos, pois facilitam na hora da organização lógica e reutilização de algoritmos.

Os sistemas operacionais móveis mais utilizados atualmente são o **Android** e o **iOS**. Para cada um deles é preferível um tipo de linguagem. No caso da plataforma Android, a melhor é Java (COMPUTER SCIENCE ONLINE, 2017). Entretanto, o Google anunciou que essa linguagem será substituída pela Kotlin, criada pela Jetbrains (SHAFIROV, 2017). Já no iOS a primeira opção é Swift ou, como segunda opção, o Objective-C.

## 3.1.4 Jogos eletrônicos

Com o constante crescimento do mercado de jogos no mundo, percebemos cada vez mais programadores dedicando integralmente seu tempo a isso. O sonho de muitos desenvolvedores é criar o próprio jogo.

Para isso é preciso de conhecimentos em matemática, física, inteligência artificial, lógica, além de muito conhecimento interdisciplinar (por exemplo, entender um mínimo de artes, música etc.).

Existem diversas plataformas possíveis para desenvolver jogos. A linguagem mais utilizada para a criação deles é o C++, embora essa tarefa seja possível com qualquer linguagem orientada a objetos.

Com o intuito de facilitar o desenvolvimento e aumentar a quantidade de jogos disponíveis no mercado, algumas empresas criaram motores – softwares com funcionalidades prontas que possibilitam a criação de uma maneira mais rápida e simplificada.

Atualmente, os motores mais utilizados pelas grandes produtoras de games são **Unity** e **Unreal**. Em ambos é possível desenvolver projetos 2D e 3D. Esses motores dispõem de editores gráficos, física, suporte a vários formatos de som, integração com editores de

modelos 3D, entre várias funcionalidades fundamentais para o desenvolvimento de qualquer jogo já implementadas.

Com o uso de um motor, programadores podem focar somente no desenvolvimento das regras do jogo.

## 3.1.5 Controle e automação

**Figura 1 – Robótica industrial.**

Fonte: Kinwun/iStockphoto.

Conhecida também por **robótica** ou **mecatrônica**, essa área está em destaque no mercado. Com o grande desenvolvimento das indústrias automobilísticas e petroquímicas, tornou-se necessário um tipo de mão de obra de menor custo e ágil produção. Assim, a robótica passou a ser fundamental para essas corporações: foi possível reduzir o custo de mão de obra, baratear o valor de mercado do produto e gerar empregos na área de manutenção e desenvolvimento.

Em 90% dos casos os robôs são estáticos, com movimentação apenas nas mãos e nas pernas e são utilizados dentro das indústrias para produção de artefatos, alguns como o Automated Guided Vehicle (AGV) e o Laser Guided Vehicle (LGV), que são robôs móveis geralmente utilizados para transporte de materiais dentro da área de produção (SOUZA; SANTO, 2009). Mas nem todos os robôs são estáticos e industriais, existem alguns que fazem tarefas domésticas; outros são utilizados em missões de risco, como desarmar uma bomba ou em algum tipo de catástrofe ou na área de medicina, em hospitais e no auxílio médico.

**Figura 2 – Robótica na medicina.**

Fonte: Gumpanat/iStockphoto.

Para contribuir nesse meio, é preciso muito conhecimento em matemática e física, além de bom raciocínio lógico. Conhecimentos em mecânica e eletrônica, programação e hardware são fundamentais. O profissional dessa área faz desde a criação de toda a máquina até a programação para torná-la inteligente.

As principais áreas de atuação para um engenheiro de automação são (PORTAL AUTOMAÇÃO INDUSTRIAL, 2017):

- automação industrial e robótica;
- automação comercial e domótica;
- projeto de equipamentos e sistemas de informática;
- projeto de equipamentos para biotecnologia.

No Brasil, o crescimento do campo é limitado, pois os equipamentos são caros e o material de estudo dificilmente é encontrado em português, mas com o grande avanço da tecnologia na medicina, estima-se que a robótica só tende a crescer no mercado de tecnologia (GUIA..., 2017).

## 3.2 Informática gerencial

Vídeo

A área de informática não é formada somente por programadores. Algumas funções são gerenciais: envolvem o contato com o cliente ou com a administração de uma infraestrutura.

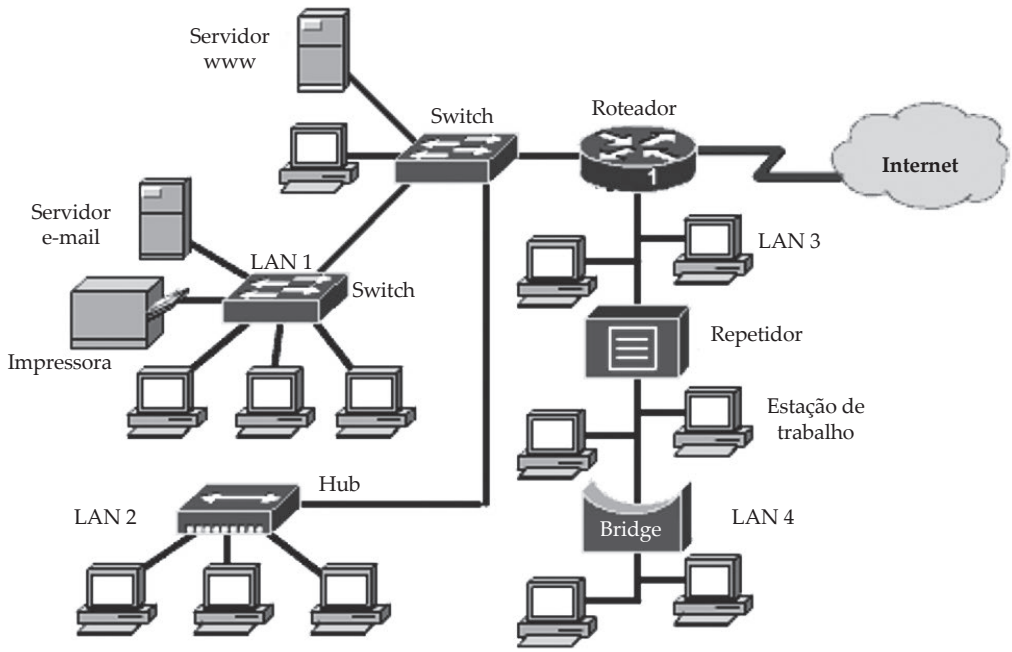
Áreas gerenciais exigem um profissional com grande responsabilidade. Não raro, são áreas muito bem-remuneradas (PORTAL GSTI, 2017). Nesta seção, estudaremos algumas delas.

## 3.2.1 Administrador de redes

O profissional da área de redes é necessário em qualquer grande empresa. Ele é responsável por garantir a funcionalidade e organização da rede, seja local ou externa. Isso exige um grande conhecimento técnico, pois deve saber identificar cabeamentos, equipamentos, softwares etc.

O profissional dessa área pode trabalhar desde analista até arquiteto de soluções (OLIVEIRA, 2017). Aqueles que optarem por essa especialização devem estar preparados para fazer avaliações e solucionar problemas de desempenho, desenvolver projetos de redes locais e distantes, gerenciar e detectar problemas de sistemas de comunicação de dados, segurança geral de redes, entre outros serviços exigidos.

**Figura 3 – Exemplo de rede empresarial.**




Fonte: Elaborada pelo autor.

O administrador de redes deve garantir o funcionamento dos equipamentos, estudar novas soluções para os problemas relacionados às redes no cotidiano empresarial, fazer relatórios de recursos necessários e criar rotinas para uma boa manutenção.

É necessário, para aqueles que desejam trabalhar na área, muita dedicação e estudo. Com a evolução constante da tecnologia, fica cada vez mais difícil para uma empresa sobreviver sem uma rede local funcionando. Qualquer problema relacionado a esse setor é de responsabilidade do administrador, portanto é preciso que o profissional em redes tenha muita dedicação e esteja preparado para plantões e atendimentos de emergência.

São conhecimentos úteis para o profissional:

- sistemas operacionais;
- infraestrutura e protocolos de rede;

- ambientes de programação e ferramentas;
- linguagens para plataforma web.

São necessárias as seguintes certificações:

- Microsoft – MCP+I, MCSE+I;
- Sun – Solaris;
- CCNA/CCNP, diferencial: CCIE.

O administrador será o responsável por acionar os técnicos certos, após identificar qual é o tipo do problema. Contudo é preciso muito cuidado, pois, em casos de grandes corporações, um problema de rede pode significar muito prejuízo. Além disso, é responsável por criar políticas de segurança e garantir que os usuários as cumpram. Isso inclui políticas para senhas, instalação de aplicativos, entre outros. Não raro, é ele também que gerencia o parque de máquinas instalado na empresa, conhecendo suas configurações, softwares, elaborando planos de manutenção etc.

## 3.2.2 Arquiteto de soluções

Uma das grandes dificuldades de se projetar sistemas corporativos é entender corretamente o problema do cliente, sua infraestrutura atual e propor uma solução que o atenda. O profissional responsável por isso é chamado de *arquiteto de soluções* ou *analista de negócios*.

São conhecimentos úteis para o profissional:

- ser autodidata;
- desenvolver mapeamento de soluções;
- facilidade em compartilhar ideias;
- pensar sistematicamente;

São necessárias as seguintes certificações:

- CCNA;
- CCDA;
- CCNP;
- CCDP;

Para trabalhar como arquiteto de soluções é preciso afinidade para encontrar estratégias e solucionar problemas (COSTA; CHAGAS, 2017). Raciocínio lógico, boa dicção e conhecimento na área de negócios são essenciais para quem deseja trabalhar nesse campo.

É preciso entender o problema por diferentes pontos de vista, para solucioná-lo de forma geral. Por isso, essas soluções devem contemplar a integração do implantado com os sistemas já existentes.

Muitas vezes, são necessários planos de migração de dados e acompanhamento da implantação do sistema para o cliente, planejando treinamentos e até realizando campanhas para reduzir a resistência ao novo sistema.

Para que se consiga atingir esse objetivo, o profissional precisará conversar com diversos funcionários da empresa, entender seus problemas e necessidades. Em seguida, ele deverá elaborar um plano para o cliente, verificando o que será desenvolvido, quem deverá ser treinado, qual será o cronograma do projeto, o custo etc. O convencimento da diretoria da empresa cliente se dará somente se o profissional, além de boa oratória, realizar demonstrações de cálculos de retorno sobre o investimento, mostrando a economia que trará o produto.

Finalmente, os profissionais dessa área trabalham muitas vezes ligados à área de vendas, auxiliando essas equipes a elaborar editais, quando o cliente desejado é o governo.

### 3.2.3 Auditor de sistemas

Para manter a rede de computadores funcionando dentro de uma empresa. É necessário um setor que se responsabilize por analisar todos os processos do sistema, para gerar relatórios de desempenho e verificar se tudo está de acordo com as leis e padrões éticos da companhia (QUAL..., 2016).

São conhecimentos úteis para o profissional:

- técnicas de análise de investimentos/riscos;
- modalidades de fraudes e outras irregularidades envolvendo dados;
- conceitos de administração;
- domínio de softwares populares (Office, sistema operacional – Windows e Linux etc.).

São necessárias as seguintes certificações:

- CISA (Certified Information System Auditors);
- CFE (Certified Fraud Examiner);
- COBIT e ITIL.

Com o intuito de proteger a informação e manter a eficácia do sistema, o profissional dessa área deve testar toda a estrutura computacional, objetivando encontrar falhas. Assim, é possível saber quais são os pontos fracos a serem melhorados e quais outros ajustes deverão ser feitos.

A parte mais importante de todo o processo é a segurança dos dados confidenciais. Por esse motivo, além de avaliar a parte técnica da rede, o auditor de sistemas tem o dever de examinar a confiabilidade das pessoas que a utilizam (QUAL..., 2017). Essa confidencialidade é de suma importância, por isso é preciso tomar cuidado na hora de compartilhar informações, porque somente pessoas de extrema confiança devem ter acesso a dados primordiais da empresa.

## 3.3 Pesquisas científicas

Vídeo

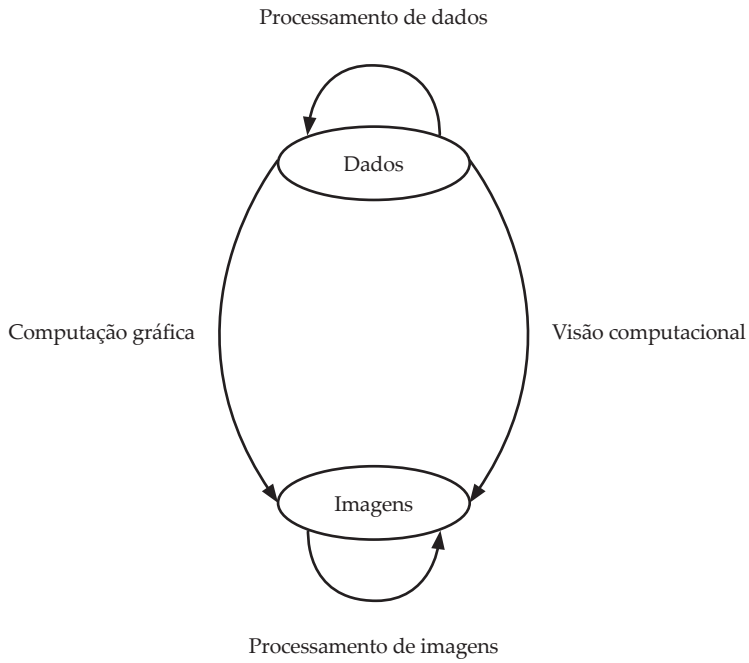
Além da indústria e da área gerencial, outro campo de destaque na área de informática é a pesquisa científica. Ela ocorre em universidades e centros de pesquisa e é por meio dela

que novas técnicas são desenvolvidas. Nesta seção, exploraremos algumas das áreas de pesquisa disponíveis no mercado.

## 3.3.1 Computação gráfica

Atendendo às necessidades de trabalhar com imagens na área computacional, surgiram pesquisas com enfoque em desenvolvimento de novas ferramentas para realizar esse tipo processamento de computação gráfica. O estudo é necessário devido à complexidade de manipular imagens no computador. É fundamental que o explorador dessa área tenha conhecimentos avançados em programação.

**Figura 4 – Demonstração visual do processo.**




Fonte: Elaborada pelo autor.

## 3.3.2 Processamento de imagens

Esse campo é o responsável por trabalhar com os dados de imagens e processá-las, realçando suas características importantes. Conhecimentos em matemática são fundamentais nessa área.

Para processar essas imagens, é preciso fazer um processo chamado *varredura*. Esse procedimento é feito por meio de matrizes, assim é possível alterar todos os pixels de uma imagem e, utilizando fórmulas matemáticas, mudar a cor, aumentando ou diminuindo o brilho, aplicando filtros etc.

O profissional conhecerá ferramentas úteis no processamento de imagens, como os histogramas, isto é, tabelas que indicam quantas vezes um pixel de determinada cor aparece na imagem e que auxiliam a criar diferentes efeitos.

## 3.3.3 Visão computacional

Muitas pessoas confundem essa área da computação gráfica com processamento de imagens, mas, apesar de trabalharem juntas para um objetivo em comum, elas têm funções diferentes em resultados.

Esse campo de estudo é responsável por extrair informações das imagens, ou seja, o objetivo é fazer com que elas reconheçam não só imagens, mas qualquer objeto do plano multidimensional. Essa tarefa envolverá não só separar os objetos presentes na imagem, mas também classificá-los (RUFINO, 2015).

Atualmente, a área de visão computacional se expandiu muito graças aos smartphones. A maioria das pessoas já possui uma câmera de vídeo nas mãos. Outra tecnologia que impulsionou a área é a dos veículos aéreos não tripulados (Vants), como os drones. Eles permitem acompanhamento de lavouras e filmagens feitas por computador. Mas as pesquisas na área também incluem o diagnóstico de doenças em imagens médicas, direção de carros sem motorista, criação de robôs mais inteligentes, sistemas de segurança capazes de rastrear automaticamente objetos e pessoas etc.

Linhas de pesquisa dessa área de acordo com a Universidade de São Paulo – USP (2017):

- Geometria computacional e processamento geométrico;
- Processamento de imagens e reconhecimento de padrões;
- Visualização e mineração visual de dados;
- Validação e avaliação de técnicas de visualização;
- Aplicações de visualização em análise de dados.

Linhas de pesquisa dessa área de acordo com a PUC-PR (2017):

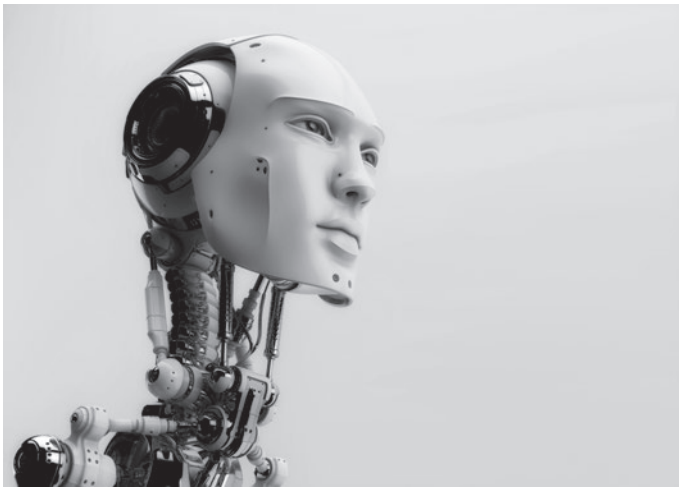
- Plataforma para criação e visualização de modelos 3D estereoscópicos em ambientes fulldome;
- Sistema imersivo para visualização de imagens holográficas em óculos RV;
- Sistema imersivo RV de teleconferência em ambientes cirúrgicos.

## 3.3.4 Inteligência artificial

Provavelmente um dos estudos mais cobiçados do mercado, a inteligência artificial (IA), trabalha com o desenvolvimento de soluções criadas para que uma máquina possa buscar respostas inteligentes no próprio sistema, agindo de forma independente.

As pesquisas que envolvem inteligência artificial são bastante complexas, matemática, física, lógica, programação são conhecimentos básicos para quem deseja adentrar nesse campo.

**Figura 5 – Humanoide.**



Fonte: Ociacia/iStockphoto.

Existem dois tipos de inteligência artificial (IA), a **IA forte** e a **IA fraca**. Elas têm pouca diferença tecnológica, mas causam um grande impacto ético (COELHO, 1994).

A IA forte objetiva estar além do sistema de soluções independente, ou seja, retratar a própria consciência humana, de forma que a máquina possa distinguir sensações e até mesmo o certo do errado. Definir se a autoconsciência pode ou não ser retratada por algoritmos lógicos tem sido pauta de discussões filosóficas desde o surgimento da área.

Em contrapartida, a IA fraca é um pouco menos ambiciosa, o objetivo é retratar o comportamento de raciocínio, sem a necessidade de consciência. Não é preciso que a máquina tenha inteligência própria, apenas que simule esse comportamento.

Linhas de pesquisa dessa área de acordo com a Universidade Federal do Rio Grande do Sul – UFRGS (2017):

- Aquisição automática de informação;
- Aprendizado de máquina;
- Bioinformática;
- Conhecimento, raciocínio e ontologias;
- Simulação de ambientes complexos;
- Robótica e sistemas autônomos e inteligentes.

Linhas de pesquisa de acordo com a Universidade de São Paulo – USP:

- Aprendizado de máquina;
- Descoberta de conhecimento e mineração de dados e textos;
- Aquisição de conhecimento;
- Sistemas baseados em conhecimento;
- Sistemas híbridos inteligentes.

## 3.3.5 Computação bioinspirada

A computação bioinspirada tem como base de estudo os padrões complexos encontrados na natureza e seus fenômenos, visando melhorar as máquinas atuais e aperfeiçoar sistemas já existentes (COELHO, 1994).

O estudo realizado nesse campo tem como objetivo criar formas de vida artificial, desenvolver dispositivos que simulam e descrevem fenômenos naturais, usar a genética ou outros recursos como novo modelo de computação etc.

Essa área de pesquisa divide-se em três ramos de estudo: computação inspirada na natureza, computação com mecanismos naturais e estudos sobre a natureza pela computação.

### 3.3.5.1 Computação inspirada na natureza

Essa ciência busca compreender os elementos da natureza e transformá-los em algoritmos para solucionar problemas e otimizar processos. Os estudos giram em torno de redes neurais e sistemas imunológicos artificiais, algoritmos evolutivos, entre outros.

Várias teorias são criadas com inspiração na natureza, as próprias leis físicas e alguns objetos, como aviões e submarinos, nada mais são do que retratos digitais da realidade. Essa evolução só é possível com muita análise e estudo do espaço natural.

### 3.3.5.2 Computação com mecanismos naturais

Estudo focado em criar sistemas de computadores genéticos ou orgânicos, que tenham bases naturais. Um exemplo são os computadores de DNA, que têm seus dados armazenados em cadeias genéticas, e os computadores quânticos, baseados na mecânica quântica. Não são binários, podem utilizar o 1 e 0 ao mesmo tempo.

Essa tecnologia é extremamente necessária e muito visada para o futuro da computação, uma vez que o poder de processamento não é infinito e precisamos dele para transmitir dados. É urgente uma solução para aumentar essa capacidade, caso contrário será criado um limite para a expansão da tecnologia.

### 3.3.5.3 Estudos sobre a natureza pela computação

Essa área busca, por meio do computador e de sistemas, simular os comportamentos naturais, tem como base de estudo a vida artificial e a geometria fractal.

O objetivo principal desse estudo é recriar a vida artificial, utilizando os conhecimentos observados no espaço natural, trazendo benefícios para a biologia, medicina, robótica e nanotecnologia. As figuras fractais são responsáveis por retratar computacionalmente os fenômenos da natureza, por isso são utilizadas nas pesquisas do ramo.

## ⊞ Ampliando seus conhecimentos

Uma das pautas frequentemente discutidas na área de informática é a ausência de regulamentação da profissão. Essa discussão levanta diversas questões como: deveriam

existir conselhos profissionais? Somente profissionais formados em informática deveriam atuar na área? É necessário um piso salarial? Os nomes e papéis das profissões devem ser regulamentados? É necessário criar métodos para que profissionais de informática possam ser responsabilizados penalmente por um software que não os atenda?

Com vistas a estudar um pouco desse tema, deixamos aqui a visão da Sociedade Brasileira de Computação, como descrito em seu texto *Efemérides da Regulamentação*.

## Profissão de informática

*(BIGONHA, 2016, p. 11-12)*

Um caminho reconhecidamente eficiente para se atingir competência profissional é o da diplomação em curso superior, ministrado por universidades de boa qualidade. O diploma de um bom curso superior, além de prover uma formação técnica especializada necessária para o exercício de uma determinada profissão, traz consigo uma preparação para a vida, com os conhecimentos necessários à mobilidade entre profissões, muito comum nos dias de hoje.

[...]

A Informática permeia de forma profunda e evidente quase todas as áreas do conhecimento humano. Para resolver problemas com o nível adequado de qualidade, além dos conhecimentos técnicos de informática, o profissional deve possuir competência nas áreas da aplicação específica, como Engenharia, Medicina, Administração, Direito, Arquitetura ou Música. Se, no início dos tempos, a multidisciplinaridade de formação profissional decorria naturalmente da inexistência de cursos superiores de informática no país, hoje é uma exigência para atender à demanda da sociedade por aplicações novas e cada vez mais sofisticadas.

E multidisciplinaridade somente se constrói sobre as férteis bases da liberdade de atuação profissional. De fato, a Informática muito se beneficiou da formação multidisciplinar oferecida pelos bons cursos superiores, os quais, durante anos, formaram engenheiros, matemáticos, administradores, físicos, advogados, apenas para citar alguns, para atuarem com competência, criatividade e engenho no desenvolvimento da Informática brasileira, cujas atividades profissionais tiveram início no Brasil na década de 1950, quando foram importados os primeiros computadores.

[...] O desenvolvimento e uso da tecnologia da informação não podem ficar restritos a uma classe de cidadãos.

É essencial para o país a participação de todos os profissionais liberais e técnicos no processo de desenvolvimento tecnológico. Não seria justo proibir profissionais de áreas como engenharia, administração, medicina, física e matemática, entre muitas outras, de aplicar a Informática na solução de problemas de sua respectiva área do conhecimento. Para assegurar a liberdade de exercício profissional, que é indispensável ao desenvolvimento da tecnologia brasileira em diversas áreas e à defesa dos interesses da sociedade brasileira, uma eventual regulamentação das profissões de informática deveria atender aos seguintes objetivos:

1. Defender a liberdade de exercício profissional, conforme estabelecido no Art. 5º, Inciso XIII, e no Art. 170, parágrafo único, da Constituição Federal;

2. garantir as condições de liberdade necessárias para o desenvolvimento tecnológico de diversas áreas de atuação profissional como Engenharia, Administração, Medicina, Biologia, Ciências Econômicas, Atuária, Química, Física e da própria Tecnologia da Informação, entre outras, que têm a Informática como uma atividade-meio;

3. garantir os meios para a atuação no mercado de trabalho de pessoal qualificado e de formação multidisciplinar, indispensável para o pleno desenvolvimento do país;

4. assegurar condições isonômicas de concorrência no mercado internacional, onde o exercício da profissão de informática é predominantemente livre;

5. defender a área de informática contra recorrentes invasões por parte de conselhos de profissão já estabelecidos no País, que insistem em definir como de sua exclusiva alçada atribuições consagradas dos profissionais de informática;

6. pacificar relações de conflitos recorrentes em editais de concurso público e de licitações, que impõem como requisito de participação o registro dos profissionais liberais da área de informática em conselhos de profissão;

7. preservar os interesses da Sociedade no uso de bens e serviços de informática.

Uma lei de regulamentação da profissão de informática com os objetivos acima permitiria convalidar uma situação de fato existente no Brasil e na

maioria dos países desenvolvidos, que é a plena liberdade do exercício profissional na área de informática.

Uma lei desse naipe proveria todas as garantias de liberdade e qualidade necessárias para impedir a criação de reserva de mercado e assim contribuir para o desenvolvimento nacional em todas as áreas.

[...]

## Atividades

1. Descreva a diferença entre os três campos descritos no capítulo: desenvolvimento, gerencial e pesquisa científica.

2. Diferencie o programador de front-end do programador de back-end em desenvolvimento web.

3. Diferencie os conceitos de IA forte e IA fraca.

4. Procure cursos presenciais e a distância em programas de pós-graduação (especialização e mestrado) na área de informática e informe-se sobre quais são as linhas de pesquisa deles.

## Referências

BIGONHA, R. S. **Efemérides da regulamentação**. Porto Alegre: SBC, 2016. Disponível em: <http://www.sbc.org.br/images/flippingbook/efemerides/efemerides-digital-2016.09.24.pdf>. Acesso em: 14 dez. 2017.

COELHO, H. **Inteligência Artificial em 25 Lições**. Lisboa: Fundação Calouste Gulbenkian, 1994.

COMPUTER SCIENCE ONLINE. **Understanding Information Technology**: a guide to IT career opportunities. Disponível em: <http://www.computerscienceonline.org/information-technology>. Acesso em: 13 dez. 2017.

COSTA, M.; CHAGAS, J. O Papel do Arquiteto de Solução. **Portal InfoQ**,. Disponível em: <https://www.infoq.com/br/articles/papel-arquiteto-solucao>. Acesso em: 4 ago. 2017.

FONSECA, A. F. Cinco profissões que o profissional de redes pode seguir. **Unipe**, 3 jan. 2017. Disponível em: <http://blog.unipe.br/graduacao/5-profissoes-que-o-profissional-de-redes-pode-seguir>. Acesso em: 15 dez. 2017.

GUIA de profissões. Engenharia Mecatrônica. **Último segundo**, 2017. Disponível em: <http://ultimosegundo.ig.com.br/educacao/guia-de-profissoes/engenharia-mecatronica/4ee0b1eefb3b72f05700002c.html#mercado>. Acesso em: 14 dez. 2017.

HENRIQUE, F. Profissão TI: auditor de sistemas. **Domínio TI**. 2011. Disponível em: <https://dominioti.wordpress.com/2011/03/26/profissao-ti-auditor-de-sistemas/>. Acesso em: 14 dez. 2017.

MARTINS, E. Profissão: Programador Web. **Tecmundo**, 2009. Disponível em: <https://www.tecmundo.com.br/2862-profissao-programador-web.htm>. Acesso em: 13 dez. 2017.

PORTAL AUTOMAÇÃO INDUSTRIAL. Disponível em: <http://www.automacaoindustrial.info/>. Acesso em: 15 dez. 2017.

PORTAL GSTI. Pesquisa Salarial Profissionais de Informática. Disponível em: <https://www.portalgsti.com.br/2017/01/pesquisa-salarial-profissionais-de-ti-2017.html>. Acesso em: 4 ago. 2017.

PUC-PR – Pontifícia Universidade Católica do Paraná. Programa de Pós-Graduação em Informática. **Visão computacional e reconhecimento de padrões**. Disponível em: <https://www.ppgia.pucpr.br/pt/?q=node/114>. Acesso em: 14 dez. 2017.

QUAL a importância da auditoria de sistemas de informação? **Zillion**, 2016. Disponível em: <https://www.zillion.com.br/site/dicas-de-ti/qual-a-importancia-da-auditoria-de-sistemas-de-informacao>. Acesso em: 14 dez. 2017.

RUFINO, R. Visão Computacional, Java e BoofCV sem JNI. **Portal InfoQ**, 2015. Disponível em: <https://www.infoq.com/br/articles/visao-computacional-boofcv>. Acesso em: 4 ago. 2017.

SANTOS JÚNIOR., Francisco S. Profissional em Redes de Computadores**. PET News**, 2010. Disponível em: <http://www.dsc.ufcg.edu.br/~pet/jornal/setembro2010/materias/carreira.html>. Acesso em: 14 dez. 2017.

SCUDERO, E. A trajetória de um desenvolvedor mobile: tudo o que você precisa saber! **Portal BeCode**, jan. 2017. Disponível em: <https://becode.com.br/trajetoria-de-um-desenvolvedor-mobile/>. Acesso em: 13 dez. 2017.

SCUDERO, E. Web Designer × Desenvolvedor Web: Entenda a diferença! **Portal BeCode**, 2016. Disponível em: <https://becode.com.br/web-designer-x-desenvolvedor-web-entenda-a-diferenca/>. Acesso em: 13 dez. 2017.

SHAFIROV, M. Kotlin on Android: now official. **Kotlin Blog**, 2017. Disponível em: <https://blog.jetbrains.com/kotlin/2017/05/kotlin-on-android-now-official>. Acesso em: 15 dez. 2017.

SOUZA, F. J. A. M; SANTO, A. E. **Automação industrial e robótica**. Covilhã, 2009. Notas de aula. Disponível em: http://webx.ubi.pt/~felippe/texts3/autom_ind_cap1.pdf>. Acesso em: 15 dez. 2017.

STACK OVERFLOW BRASIL. **Visão Computacional**. Disponível em: <https://pt.stackoverflow.com/tags/vis%C3%A3o-computacional/info>. Acesso em: 15 dez. 2017.

UFRGS – UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL. Instituto de Informática de UFRGS. **Inteligência Artificial**. Grupos de Pesquisa. Disponível em: <http://www.inf.ufrgs.br/site/pesquisa/grupos-de-pesquisa/inteligencia-artificial>. Acesso em: 20 dez. 2017.

USP – UNIVERSIDADE DE SÃO PAULO. Instituto de Ciências Matemáticas e de Computação. **Visualização, imagens e computação gráfica**. Disponível em: <http://conteudo.icmc.usp.br/Portal/Pesquisa/pesquisaDinamico.php?id_laboratorio=69>. Acesso em: 14 dez. 2017.

VOCÊ conhece bem os principais profissionais da área de TI? **GAEA Consulting**, 2016. Disponível em: <https://gaea.com.br/voce-conhece-bem-as-principais-areas-de-ti/>. Acesso em: 14 dez. 2017.

## ☑ Resolução

**1.** Desenvolvimento: trata-se da produção de software para venda. Profissionais dessa área trabalharão com programação na web, dispositivos móveis, desktop, games ou automação.

Gerencial: é o campo da informática que lida com cliente. Profissionais dessa área precisam ter boa comunicação, saber tomar decisões e atuarão mais próximo de funções administrativas.

Pesquisa científica: é a área acadêmica. Profissionais da área pesquisarão inovações tecnológicas em centros de pesquisa e universidades, muitos deles se tornando professores em faculdades.

**2.** Programador front-end: cuida do desenvolvimento da interface gráfica, a parte visível do website. O programador de backend cuida do desenvolvimento da parte funcional do site, implementando as regras do negócio, lidando com banco de dados.

**3.** IA forte: tenta resolver problemas da maneira mais eficiente e otimizada possível. IA fraca: Procura simular o comportamento humano incluindo: emoções, ou diálogos.

**4.** Resposta pessoal.

# 4
# Organização física do computador – Hardware

Um computador é feito de diversos componentes interconectados, cada um com uma tarefa distinta. Neste capítulo, iremos entender o funcionamento desses componentes e como eles se relacionam.

Vídeo

# 4.1 Tipos de computadores

A primeira coisa que temos de entender sobre computadores é que eles não se resumem ao equipamento que temos em cima de nossas mesas. Há diversos tipos diferentes de dispositivos. Vamos iniciar nosso estudo sobre o hardware classificando-os.

## 4.1.1 Supercomputadores

Supercomputadores (Figura 1) são dispositivos grandes, criados por empresas com o intuito de realizar enormes volumes de processamento. O desempenho desses computadores é medido ao se calcular quantas operações de ponto flutuante (números com vírgula) ele pode realizar em um único segundo.

**Figura 1 – Supercomputador IBM Blue Gene/P no Laboratório Nacional de Argonne nos Estados Unidos.**

Fonte: Wikimedia Commons.

Para se ter uma ideia da potência desses gigantes, o Deep Blue, computador que venceu o campeão de xadrez Garry Casparov, operava na velocidade de 11,38 giga flops (IBM, 2017).

Supercomputadores são usados em diversos ramos da ciência, incluindo previsão do tempo, simulações aéreas, balística, simulações físicas (astrofísica, mecânica quântica) e simulações biológicas. Cientistas e estudantes podem alugar o tempo de um computador desse tipo para suas próprias pesquisas acadêmicas. Historicamente, muitos supercomputadores também foram utilizados no ramo da criptografia para quebrar senhas e fazer espionagem.

## 4.1.2 Computadores pessoais

Computadores pessoais são aqueles que estamos acostumados a trabalhar em nosso dia a dia. Entre seus tipos básicos, podemos destacar:

- Computadores desktop – criados para serem colocados sobre uma mesa. Destacam-se principalmente pela presença de um gabinete retangular, ligado ao monitor de vídeo e aos demais periféricos (teclado, mouse, impressora etc.). Uma das grandes vantagens desses modelos é que são configurados para diversos fins, como servir de estação de trabalho ou jogos. Para isso, basta comprar itens de hardware separadamente. Por requererem menos miniaturização, são mais baratos do que notebooks de potência equivalente, além disso, os componentes de hardware podem ser facilmente trocados e substituídos, aumentando a vida útil do equipamento.
- Computadores monobloco – são um subtipo de computador desktop que combinam monitor e CPU em uma única peça. Um exemplo famoso desse tipo de computador para uso pessoal é o iMac, da Apple. Alguns computadores monobloco também possuem monitor touchscreen e são popularmente utilizados em shoppings centers ou em lojas.
- Laptops/notebooks – são computadores pessoais pequenos, projetados para que sejam portáteis. Suas dimensões variam. Podem ser computadores de 15 polegadas e largos, com o intuito de substituir o desktop, podem ser finos e leves, para necessidades comerciais gerais, ou podem ser muito pequenos, com tamanhos de até 8 polegadas, para serem muito leves e portáteis.

  Todo hardware adicional, como memória, teclado ou placa de vídeo, costuma vir integrado ao equipamento. Alguns notebooks incluem até touchscreen, com o intuito de substituir o mouse.

  O design geralmente lembra o de uma concha de marisco, pois é possível abri-lo. A maior parte desses equipamentos também possui bateria e pode operar sem estar conectado a uma fonte de energia.
- Minicomputadores – são dispositivos pequenos e portáteis, mas geralmente precisam ser ligados a um monitor e a periféricos, como teclado e mouse, para serem operados. Geralmente, são dispositivos baratos e, além do uso doméstico, são comumente usados em lojas e escritórios para criar painéis informativos ou como pequenos servidores.

## 4.1.3 Dispositivos móveis

Tratam-se de dispositivos de tamanho bastante reduzido, projetados para serem levados a todo lugar. Entre eles, destacam-se:

- Tablets – dispositivos finos, retangulares. Possuem tela LCD, touchscreen, câmera frontal e traseira e sensores, como giroscópio e acelerômetro (APPLE, 2017). Seu uso doméstico é em entreterimento (leitura de livros, acesso à internet, filmes e jogos).

Comercialmente, é muito usado em vendas, substituindo catálogos e permitindo ordens de pedidos. Geralmente, têm grande autonomia de bateria e tela luminosa.

- E-readers – similares a tablets, porém com função específica para leitura de livros. Atualmente, o mais famoso desse tipo é o Kindle, da Amazon. Essa tecnologia utiliza e-paper (papel eletrônico), trazendo conforto para a leitura. Consomem pouquíssima bateria e podem ficar semanas sem recarregar, caso recursos como a retroiluminação e a internet não sejam ligados (VISIONNECT, 2017). Diferentemente dos tablets, e-readers não possuem telas coloridas, nem são rápidos o suficiente para a visualização de vídeos.
- Pocket computers – são computadores feitos para caber dentro do bolso ou na palma da mão. Na atualidade, a maioria dos pocket computers são smartphones. No passado, foram populares o iPod e os Palmtops. Uma das grandes características dos pocket computers é a presença de diversos tipos de sensores, tais como acelerômetros, GPS e giroscópios – que viabilizam uma gama incrível de aplicações.
- iWatches – computadores ainda menores. Tratam-se de relógios de pulso com a capacidade de receber notificações. Assim como os smartphones, os iWatches também dispõem de sensores, como giroscópio e acelerômetro e monitores de batimento cardíaco.

### 4.1.4 Outros dispositivos

Há uma série de dispositivos eletrônicos ainda menores, criados para permitir a automação industrial e comercial. Pode parecer futurista, mas os computadores estão presentes em carros (injeção eletrônica) ou até na cozinha (muitos fornos de micro-ondas são microprocessados).

Existem também computadores do tamanho de um cartão de crédito, como o Raspberry Pi e kits de desenvolvimento, como o Arduíno. Ambos permitem criar uma série de pequenas automações, ou mesmo dar inteligência a pequenos robôs ou controlar o voo de drones.

Vídeo

## 4.2 Componentes

Um computador é formado por diversos componentes menores, cada um realizando um papel diferente. Existem dois tipos básicos de componentes (STALLINGS, 2006): os periféricos, que são ligados ao computador; e os principais, que fazem parte do computador em si.

### 4.2.1 Periféricos

Periféricos são componentes que auxiliam no uso do computador. Eles podem ser divididos em **dispositivos de entrada**, que permitem colocar dados no computador, tais como

teclado, mouse, unidades de DVD e scanner; e **dispositivos de saída**, que retiram informação do computador, tais como monitor de vídeo, alto-falantes ou fones de ouvido e impressora.

### 4.2.2 Componentes principais

Entre os componentes principais, podemos citar:

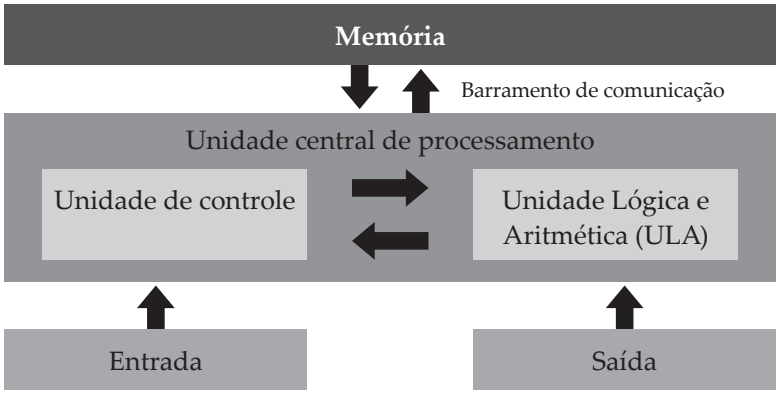
- A placa-mãe – é o principal do circuito. Interliga todos os demais componentes, diretamente ou via cabo. É nela que serão ligados o processador e a memória e é onde estará o firmware que inicia o computador.
- O processador – também conhecido como Unidade Central de Processamento (CPU), realiza as instruções dos programas. Nele, estão contidos milhares de transistores. Sua operação gera muito calor, por isso, geralmente é conectado a dissipadores de temperatura e recebe ventilação de pequenos coolers. Podemos associar o processador ao "cérebro" do computador.
- Fonte de alimentação – converte uma fonte de corrente alternada (geralmente 110 V ou 220 V) para as unidades de corrente contínua utilizadas pelos componentes do computador. Geralmente necessita de uma potência em torno de 40% maior do que a soma de todos os dispositivos internos somados, para que possa resistir a picos de energia.
- Memória – refere-se à memória de armazenamento volátil, ou seja, a que possui dados que serão apagados quando o computador for desligado. Geralmente, dividida em placas de circuito compridas, chamadas de *pentes de memória*. Ela é utilizada para armazenar os programas e seus dados.
- Disco rígido (HD) – são dispositivos com grande capacidade de armazenamento de dados. Também são considerados dispositivos de memória, porém não voláteis e com velocidade consideravelmente menor. Os discos rígidos geralmente utilizam discos magnéticos. Um modelo que vem se popularizando são os discos de estado sólido (SSD), muito velozes, silenciosos e menos volumosos, porém mais caros.
- Placa de vídeo (GPU) – realiza a tarefa de processar a saída gráfica da CPU e enviar para o monitor de vídeo. Possui processador e memórias próprias e é responsável por realizar desenhos em tempo real, sendo essencial para jogos, modelagem 3D, vídeo ou arquitetura.
- Gabinete – é a carcaça, geralmente metálica, que envolve todo computador. Pode ser compacto ou bastante grande (torre). Pode ser vertical ou horizontal. Como é a parte visível do computador, pode-se encontrar modelos bastante personalizados.

Vídeo

## 4.3 Arquitetura

Como vimos no Capítulo 1, os computadores foram organizados de modo que dados e programas fiquem armazenados nas mesmas unidades de memória. Essa arquitetura, conhecida como **Arquitetura de Von Neumann**, persiste até a atualidade (BACKUS, 2017). O diagrama a seguir (Figura 2) descreve os componentes presentes nessa arquitetura:

**Figura 2 – Arquitetura de Von Neumann.**




Fonte: Elaborada pelo autor.

Nas seções a seguir, mostraremos cada uma dessas partes.

## 4.3.1 Memória

A memória de um computador é formada por um conjunto de **registradores**, ou seja, espaços de memória em que os dados podem ser guardados. A cada um desses registradores é associado um **endereço**. Este nada mais é do que uma numeração sequencial.

Portanto é a quantidade de bits usados no endereçamento que vai determinar a capacidade máxima de memória de um computador. Com 32 bits, por exemplo, o computador será capaz de endereçar $2^{32}$ endereços, ou seja, pouco mais de 4 milhões de bytes, equivalente a 4 GB de memória.

Os dados são transferidos entre a memória e o computador. O tamanho máximo desse bloco de dados é chamado de **palavra**. Esse valor é importante, pois tem impacto significativo na construção do computador: uma palavra maior na memória significa utilizar mais trilhas de circuito para transportar esse dado até o processador. Significa também ter um processador que seja capaz de processar mais bits em uma única instrução.

A memória é responsável por armazenar tanto os dados de um programa quanto as instruções a serem executadas (o programa em si). É essa característica que torna a máquina de Von Neumann tão útil, afinal, se programas podem ser tratados como dados, podemos escrever programas que geram programas. São esses os compiladores das linguagens de programação.

## 4.3.2 Unidade central de processamento (UCP)

Trata-se do processador do computador. Também é conhecida por seu termo em inglês, *Central Processing Unit (*CPU). Sua principal tarefa é executar os programas dentro da máquina. Para isso, conta com dois componentes, a Unidade de Controle (UC) e a Unidade Lógica e Aritmética (ULA) (STALLINGS, 2006).

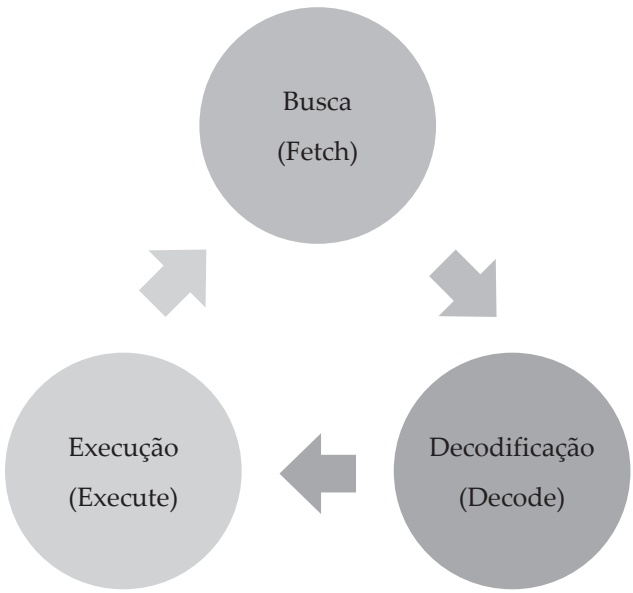
## 4.3.3 Unidade de controle

Para o computador, uma **instrução** nada mais é do que um valor numérico que instrui o processador a realizar uma operação. Esse valor é convencionado pelos fabricantes do hardware do processador. Por exemplo, se tivéssemos uma empresa que produzisse computadores, poderíamos dizer que a soma seria representada pelo número 1, a subtração pelo 2, e que esses comandos somariam com os dois próximos números na memória. Portanto um programa poderia ser escrito como **01** 10 20 **02** 60 10. Esse programa faria o cálculo de 10+20 e 60-10.

É a unidade de controle que executa os programas no computador. Ela faz isso controlando um **ponteiro de execução** (STALLINGS, 2006), ou seja, conhecendo qual endereço deve ser executado.

A unidade de controle então realiza três operações (Figura 3) (STALLINGS, 2006):

**Figura 3 – Ciclo busca, decodificação e execução.**




Fonte: Elaborada pelo autor.

- Busca da instrução – no caso do nosso programa, ao iniciar, ele leria a instrução 01.
- Decodificação da instrução – o computador descobriria que 01 é a soma. Como essa instrução necessita de dados, ele também faz a leitura deles: os números 10 e 20.
- Execução da instrução – a unidade de controle aciona os circuitos que irão realizar as instruções. No caso das operações matemáticas e lógicas, esse circuito é a Unidade Lógica e Aritmética (ULA).

Ao fim desses três passos, o computador move o ponteiro de execução para a próxima instrução da sequência e repete o processo. Todos os programas que temos em nossos computadores nada mais são do que milhões de instruções desse tipo.

## 4.3.4 Unidade lógica e aritmética

Realiza as operações matemáticas propriamente ditas. Possui pelo menos um registrador para indicar a operação a ser realizada e registradores para armazenar o valor dos operandos e para armazenar o resultado. Realiza no mínimo as seguintes operações (BACKUS, 2017):

- Aritmética básica – soma e subtração de dois números. Incremento e decremento de 1 em um único número. Complemento de dois, ou seja, subtrair de 0 o número.
- Operações lógicas – e, ou, ou exclusivo, não.
- Movimentações de bits (shift) – shift para esquerda ou direita, preservando ou não o sinal. Rotação de bits.

As ULAs também podem conter operações mais avançadas, tais como cálculo de raiz quadrada ou do resto da divisão. Entretanto, operações mais complexas podem exigir circuitos mais elaborados e isso impacta no custo. São usados três tipos de estratégia (STALLINGS, 2006):

- Cálculo em uma única operação – monta-se um circuito grande e complexo que consegue realizar a operação imediatamente. É veloz, mas é a alternativa mais cara.
- Cálculo iterativo – monta-se um circuito mais simples que, com vários passos, realiza a operação. Por exemplo, uma multiplicação pode ser resolvida como um conjunto de somas (3*5=5+5+5).
- Pipeline – utiliza-se um circuito mais simples, mas com várias ULAs paralelas, formando uma espécie de linha de montagem. Assim, quando a primeira operação é realizada, outra operação já pode ser introduzida na sequência. Trata-se de uma alternativa intermediária.

## 4.3.5 Barramento

Barramentos são as trilhas de circuito que transportam informação entre os vários componentes dentro do computador.

Um dos problemas do barramento, entretanto, reside no fato de a velocidade de tráfego desses dados geralmente ser menor do que a velocidade na qual o processador é capaz de trabalhar. Isso porque tanto a memória quanto os barramentos não podem transportar mais de um dado de uma só vez. Isso gera um gargalo, conhecido como *gargalo de Von Neumann* (BACKUS, 2017).

Em supercomputadores, esses gargalos muitas vezes são solucionados com o uso de múltiplos barramentos, para permitir esse paralelismo.

### Atividades

**1.** Descreva o papel dos dois principais componentes da arquitetura de uma CPU.

**2.** Associe os termos da coluna da esquerda com as definições à direita:

| | |
|---|---|
| A. CPU | ( ) Realiza as operações matemáticas e lógicas no computador. |
| B. Barramento | ( ) Trilhas de circuito por onde a informação trafega entre os componentes do computador |
| C. Palavra | ( ) Processador destinado à criação de gráficos, essencial para jogos. |
| D. Memória | ( ) Dispositivos que exibem informações do computador, por exemplo, monitor, impressora e fone de ouvido. |
| E. GPU | ( ) Indica qual será o próximo comando a ser dado para o processador. |
| F. Dispositivo de saída | ( ) Quantidade de informação que a memória é capaz de armazenar em um único bloco de dados. |
| G. Ponteiro de execução | ( ) Processador do computador, formado pela unidade lógica e aritmética (ULA) e a unidade de controle (UC). |
| H. ULA | ( ) Armazena as informações e os programas. |

**3.** Aponte se o dispositivo indicado é de entrada ou saída:

a. Óculos de realidade virtual: _____.

b. Joystick: _____.

c. Microfone: _____.

d. Vibracall: _____.

e. Botões de volume do celular: _____.

**4.** Diferencie **memória volátil** de **não volátil** e cite um dispositivo de cada.

**5.** Diferencie as etapas de busca, identificação e execução da unidade de controle.

## Referências

APPLE. **Core motion**: process accelerometer, gyroscope, pedometer, and environment-related events. Documentation. Disponível em: <https://developer.apple.com/reference/coremotion>. Acesso em: 23 nov. 2017.

BACKUS, J. W. Can programming be liberated from the von Neumann Style? A functional style and its algebra of programs. **ACM Communications**, v. 21, n. 8, ago. 1978. Disponível em: <http://dl.acm.org/citation.cfm?doid=359576.359579>. Acesso em: 23 nov. 2017.

IBM. **Deep Blue**. Overview. Disponível em: <http://www-03.ibm.com/ibm/history/ibm100/us/en/icons/deepblue/>. Acesso em: 30 nov. 2017.

STALLINGS, W. **Computer organization & architecture**: designing for performance. 7. ed. Upper Saddle River: Pearson Prentice Hall, 2006.

VELLOSO, F. C. **Informática**: conceitos básicos. Rio de Janeiro: Ed. Campus, 1994.

PRIMOZIC, U. Battery lifetime of electronic paper signs. **Visionect**, 26 maio 2015. Disponível em: <https://www.visionect.com/blog/battery-lifetime-of-electronic-paper-signs>. Acesso em: 23 nov. 2017.

## ☑ Resolução

**1.** Unidade de controle: executa os programas do computador por meio do ciclo de busca, identificação e execução de instruções.

Unidade lógica e artimética: realiza cálculos e operações lógicas.

**2.** Ordem correta: H; B; E; F; G; A; C; D.

**3.** a) Saída; b) Entrada; c) Entrada; d) Saída; e) Entrada.

**4.** Memória volátil: o acesso às informações é muito veloz. Assim que o computador desliga, os dados são apagados. Exemplos: memória RAM e memória cache.

Memória não volátil: as informações permanecem gravadas mesmo com o computador desligado. Exemplos: disco rígido e disco SSD.

**5.** Busca: localiza a próxima instrução a ser executada.

Identificação: decodifica o significado dessa instrução e carrega os dados necessários para sua execução.

Execução: realiza a operação descrita e guarda os resultados.

# 5
# Organização lógica do computador - O sistema operacional

No capítulo anterior, vimos que o computador representa programas e dados de maneira numérica. Neste capítulo, iremos explorar um pouco mais esse conceito, entendendo como os vários 0s e 1s da eletrônica digital se tornam letras, números ou, até mesmo, imagens.

Vídeo

# 5.1 Bits e bytes

Os circuitos de um computador trabalham representando a informação de presença ou ausência de sinal. Por isso, sua unidade mais básica, chamada de *bit*, representa apenas dois valores 0 ou 1. O conjunto de 8 bits é chamado de **byte**.

Como 0 e 1 representam apenas dois estados, as unidades maiores também são baseadas em potências de 2. Um **kilobyte** são $2^{10}$ bytes, ou 1024 bytes. Um **megabyte** equivale a $2^{20}$ ($2^{10} \times 2^{10}$) bytes, ou 1.048.576 bytes, ou seja, aproximadamente 1 milhão de bytes. É mais fácil recordar essas unidades como potências de 10, mesmo que essa informação não seja tão precisa.

A Tabela 1 mostra os valores exatos de cada unidade:

**Tabela 1 – Valores de cada unidade.**

| Unidade | Sigla | Múltiplo | Tamanho (bytes) |
|---|---|---|---|
| Byte | B | $2^0$ | 1 |
| Kilobyte | kB | $2^{10}$ | 1024 |
| Megabyte | mB | $2^{20}$ | 1.048.576 |
| Gigabyte | gB | $2^{30}$ | 1.073.741.824 |
| Terabyte | tB | $2^{40}$ | 1.099.511.627.776 |
| Petabyte | pB | $2^{50}$ | 1.125.899.906.842.624 |

Fonte: Elaborada pelo autor.

Alguns sistemas, como a rede, utilizam as referências em bits. As siglas são similares, porém utilizando **b** minúsculo, por exemplo: 1 Kb. É importante tomar cuidado pois estamos falando em uma unidade oito vezes menor. Uma conexão de rede, por exemplo, de 500 mb/s (megabits/segundo) equivale a uma conexão de 62,5 mB/s.

Podemos associar valores de bits à informação. Poderíamos dizer que 0 significa "ligado" e 1 significa "desligado". Ou então, poderíamos dizer que 0 significa "alto" e 1 significa "baixo". Observe que utilizando vários bits, poderíamos representar mais estados, por exemplo, poderíamos representar quatro cores utilizando 2 bits:

**Quadro 1 – Representação das cores.**

| Dígito | Cor |
|---|---|
| 00 | Preto |
| 01 | Vermelho |
| 10 | Verde |
| 11 | Marrom |

Fonte: Elaborado pelo autor.

As cores, na verdade, eram utilizadas em uma das palhetas de uma das primeiras telas coloridas de computador, conhecida como *monitor CGA*, capaz de reproduzir só quatro cores simultaneamente. (IBM, 2017).

Observe que se utilizarmos 1 bit a mais, poderemos representar oito informações diferentes (000, 001, 010, 011, 100, 101, 110, 111). Com mais 1 bit, 16. Ou seja, com b bits representaremos $2^b$ valores diferentes.

Não importa o que iremos representar. O importante é entendermos que uma sequência de bits sempre será utilizada e a informação será exibida de acordo com a **interpretação dessa sequência**. Essa interpretação é chamada de *tipo de dado* e alguns tipos de dados comuns são números sem sinal, números com sinal, datas ou letras.

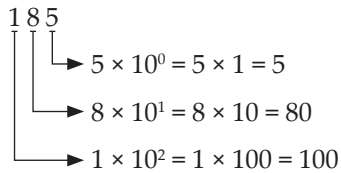
Vídeo

## 5.2 Bases numéricas

Quando queremos escrever números, escrevemos dígitos de 0 até 9. A combinação desses dígitos nos dá qualquer número que desejamos representar. Por exemplo, o valor 185 representa cento e oitenta e cinco. Esse sistema é chamado de *decimal*, pois possuímos **dez** símbolos para representar cada dígito: 0, 1, 2, 3, 4, 5, 6, 7, 8 e 9.

Como temos 10 símbolos, os números são agrupados em potências de 10, que são determinadas pela sua casa decimal, por exemplo:

**Figura 1 – Número 185 na base decimal.**



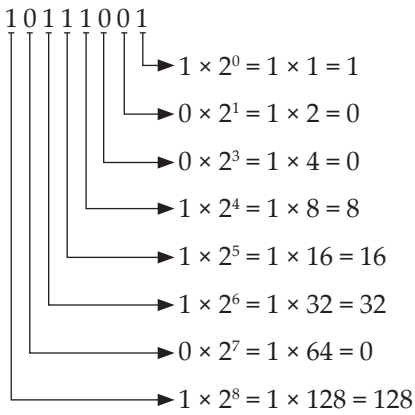

Fonte: Elaborada pelo autor.

O valor 185 é obtido pela soma de 5 + 80 + 100.

### 5.2.1 Base binária

Como o computador representa a informação utilizando apenas dois dos dígitos 0 e 1, ele não utiliza a base decimal, mas a base **binária**. Ele precisará, portanto, de um conjunto muito maior de bits para representar o mesmo o valor 185, observe:

**Figura 2 – 185 em binário (10111001).**




Fonte: Elaborada pelo autor.

Observe que a soma de 128 + 32 + 16 + 8 + 1 também é 185. Portanto o número 10111001 representa 185 nessa base. Foram necessários 8 dígitos (1 byte inteiro) para representar esse valor.

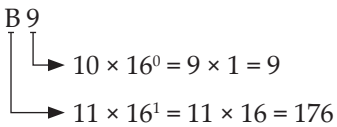
Como vimos, 8 bits podem representar 256 valores diferentes ($2^8$). Entretanto, iniciando do 0, poderíamos representar qualquer número de 0 até 255. Com 2 bytes, poderíamos representar qualquer número entre 0 e 65535. Com 4 bytes (32 bits), qualquer número entre 0 e 4.194.303.

## 5.2.2 Base hexadecimal

Os primeiros cientistas da computação precisavam de uma forma mais sucinta para representar os valores numéricos utilizados pelo computador. A forma encontrada foi recorrer a uma base com 16 símbolos: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D e F. Com a letra A valendo 10, B valendo 11, C, valendo 2 e assim sucessivamente, até o F, valendo 15.

Vejamos como o número 185 fica representado nessa base:

**Figura 3 – 185 em hexadecimal (B9).**




Fonte: Elaborada pelo autor.

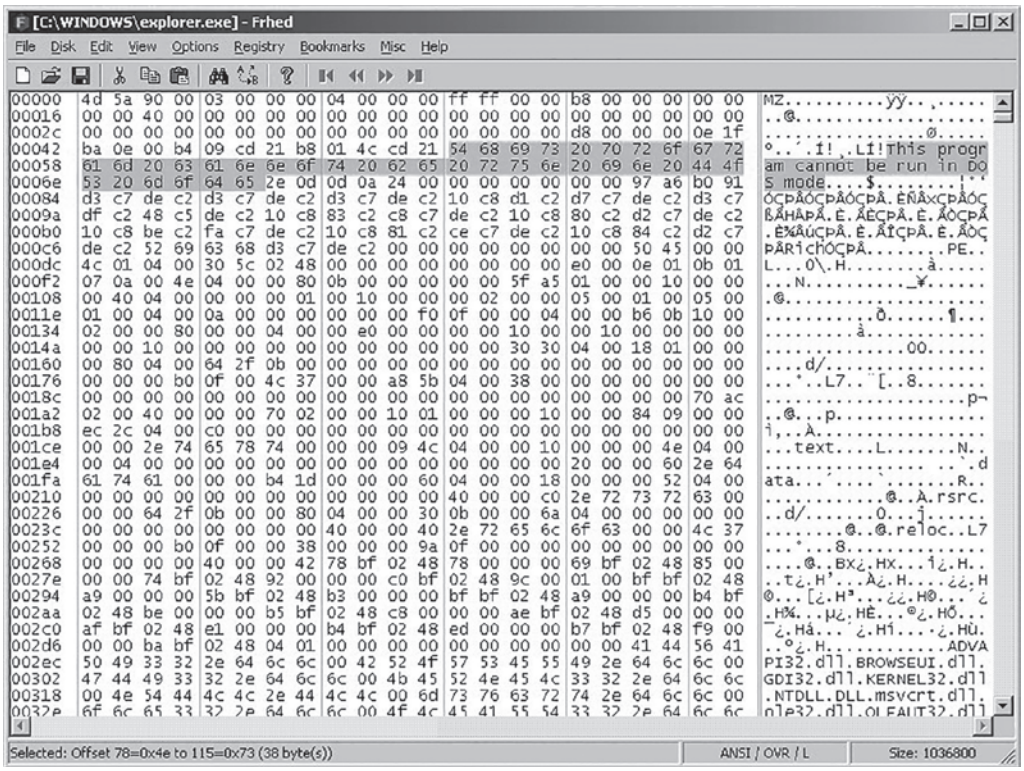
Observe que a conversão para binário nessa base é muito simples. Lembre-se que a letra B representa o valor 11, que em binário é 1011. O número 9 em binário é representado por 1001. B9 é a junção dos dois: 10111001. Dessa forma, os cientistas da computação poderiam decorar apenas os 16 valores binários correspondentes à sequência de 0 até F, e poderiam entender números muito grandes como F0BA00. Ou, mesmo que esquecessem, poderiam apenas fazer a conversão dígito por dígito.

Programadores indicam que um número está na base hexadecimal pelo prefixo 0x. Por exemplo, ao dizer 0x150, eles estão dizendo que o número é 150 na base hexadecimal, ou seja, equivalente ao valor 336 na base decimal.

$$1 \times 16^2 + 5 \times 16^1 + 0 \times 16^0 = 256 + 80 + 0 = 336$$

A imagem a seguir mostra um programa qualquer, para Windows, exibido byte a byte:

**Figura 4 – Arquivo explorer.exe exibido byte a byte.**




Fonte: Raihan Kibria.

## 5.2.3 Conversões entre bases

Já vimos que para converter qualquer base para a base decimal, utilizamos a fórmula Dígito × Base$^{casa-1}$ em que:

- **Dígito** – é um dígito numérico presente no número.
- **Base** – é a base numérica utilizada: 2 para binário, 10 para decimal, 16 para hexadecimal.
- **Casa** – é onde o dígito está. Ela é contada da direita para a esquerda. No número 185, 5 está na primeira casa, 8 na segunda casa e 1 na terceira casa.

O quadro a seguir, mostra algumas conversões numéricas como exemplo:

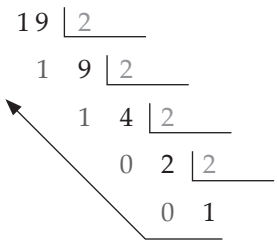
**Tabela 2 – Conversões numéricas.**

| Base | Número | Conversão | Valor |
|---|---|---|---|
| Binária | 1101 | $1 \times 2^0 + 1 \times 02^1 + 1 \times 2^2 + 1 \times 2^3$<br>$1 + 0 + 4 + 8$ | 13 |
| Binária | 110100 | $0 \times 2^0 + 0 \times 02^1 + 1 \times 2^2 + 0 \times 2^3 + 1 \times 2^4 + 1 \times 2^5$<br>$0 + 0 + 4 + 0 + 16 + 32$ | 52 |
| Hexadecimal | 5EB3 | $3 \times 16^0 + 11 \times 16^1 + 14 \times 16^2 + 5 \times 16^3$<br>$3 \times 1 + 11 \times 16 + 14 \times 256 + 5 \times 4096$<br>$3 + 176 + 3\,584 + 20\,480$ | 24 243 |

Fonte: Elaborada pelo autor.

Há duas formas de converter da base decimal para qualquer outra base numérica. A primeira é por meio das divisões sucessivas (VELLOSO, 1994). Para isso, dividimos o número várias vezes pela base desejada, até que não seja mais possível dividir. O resultado mais todos os restos pegos na ordem inversa será a base convertida. Por exemplo, observe a conversão do número 19 em binário:

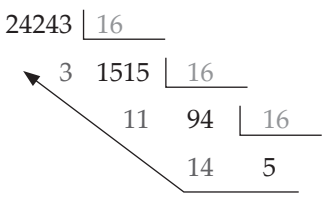
**Figura 5 – Conversão do número 19 para binário.**




Fonte: Elaborada pelo autor.

O resultado é o número 10011. Vejamos como exemplo o número 24243 como seria a representação do número decimal 24243 em hexadecimal:

**Figura 6 – Conversão do número 24243 para hexadecimal.**




Fonte: Elaborada pelo autor.

Em hexadecimal, 14 é representado pelo dígito E 11 por B, portanto, teremos o valor 5EB3.

Outra forma de converter números decimais para binários é por meio de subtrações sucessivas. Ela funciona apenas para números simples. Para utilizá-la, escrevemos as potências de 2 a cada casa numérica, em ordem decrescente, em uma tabela, iniciando pelo valor anterior ao imediatamente superior ao número que queremos converter. Depois, continuamos com os seguintes passos (VELLOSO, 1994):

1. Se o número que quisermos converter for maior ou igual à potência de 2 daquela linha da tabela, incluímos o dígito 1 no resultado, naquela linha.
2. Subtraímos o valor da potência daquela linha do número que estamos convertendo.
3. Repetimos o processo até que o número que estamos convertendo seja 0.

Vamos exemplificar o processo novamente convertendo o número 19. As potências de 2 até o número superior ao 19 são: 1, 2, 4, 8, 16, 32. Portanto montamos a tabela com as potências a partir do 16, já que ele é o imediatamente anterior à primeira potência superior a 19, ou seja, 32. Em seguida, realizamos o cálculo:

**Tabela 3 – Potências.**

| Número conversão | Potência | Dígito | Subtração |
|---|---|---|---|
| 19 | 16 | **1** (19 > 16) | 19 – 16 = 3 |
| 3 | 8 | **0** (3 < 8) | 3 – 0 = 3 |
| 3 | 4 | **0** (3 < 4) | 3 – 0 = 3 |
| 3 | 2 | **1** (3 > 2) | 3 – 2 = 1 |
| 1 | 1 | **1** (1 = 1) | 1 – 1 = 0 |

Fonte: Elaborada pelo autor.

O resultado é o valor binário 10011, indicado pelos dígitos lidos de cima para baixo. Essa forma pode não ser prática para números grandes, mas é bastante fácil de converter de cabeça. Especialmente útil para valores dentro do espaço de um byte.

## 5.3 Tipos de dados

Vídeo

Nesta seção, iremos explicar algumas representações comuns do computador. São os números inteiros positivos e negativos, os números de decimais e as letras. Também veremos um tipo de dado mais complexo: as imagens.

### 5.3.1 Números inteiros

O computador trabalha com dois tipos de números inteiros: com e sem sinal. Para os números sem sinal, o seu valor é simplesmente a representação binária do número decimal.

Para números com sinal, utiliza-se a seguinte regra:

- Se o número for **positivo**, o **primeiro bit** possuirá o valor **0**. Os demais indicarão o número em binário.
- Se o número for **negativo**, o **primeiro bit** possuirá o valor **1**. Os demais indicarão o **complemento de dois** do número em binário.

O complemento de 2 é obtido invertendo todos os bits do número e somando-se um ao resultado. Por exemplo, para representar o número -6 em 8 bits:

1. Escrevemos o número 6 positivo em 8 bits: 0000010.
2. Invertemos o bit de sinal: 10000101
3. Invertemos os demais bits: 11111010
4. Somamos 1 ao resultado: 11111011

A vantagem do complemento de 2 é que somente existirá um único zero. A desvantagem é que esse sistema representa um número negativo a mais do que positivo, sendo assimétrico.

Em linguagens de programação, geralmente utilizamos 1, 2, 4 ou 8 bytes para representar tipos numéricos (IEEE, 2017). Portanto, os valores máximos e mínimos que podemos representar variam (Tabela 4):

**Tabela 4 – Representação de bytes.**

| Tipo | Bytes | Com sinal | Sem sinal |
|---|---|---|---|
| Byte | 1 | -128 até 127 | 0 até 255 |
| Short | 2 | -32768 até 32.768 | 0 até 65.535 |
| Int | 4 | -2097152 até 2097151 | 0 até 4.194.303 |
| Long | 8 | -8.796.093.022.208 até 8.796.093.022.207 | 0 até 17.592.186.044.415 |

Fonte: Elaborada pelo autor.

## 5.3.2 Números decimais

Existem duas formas diferentes de representarmos números decimais. Por meio de **ponto flutuante** ou de **ponto fixo**. Vamos começar entendendo a primeira forma, pois ela é mais simples.

Vamos supor que queremos representar um número decimal, utilizando 10 dígitos. Esse número terá uma vírgula. Com um número de ponto fixo, arbitraríamos uma casa para a vírgula , por exemplo, a quarta. Assim, o maior número possível de ser representado seria 999999,9999, e o menor sem contar o zero seria 000000,0001. Essa é a estratégia de ponto fixo, uma vez que a posição da vírgula foi definida por nós e não muda.

Uma outra abordagem, é considerar que a primeira casa contém a posição da vírgula, contada da esquerda para direita. Portanto um número como 2000000001 representaria o valor 0,01 e um número como 4000000001 o valor 0,0001.

Dessa forma, o maior número representado sem a vírgula seria dado por 0999999999 e o menor número sem o zero seria 9000000001 = 0,000000001. Com essa representação, a casa decimal varia, ou seja, "flutua" por entre as casas. Por isso, a chamamos de ponto flutuante.

Os computadores representam números de ponto flutuante utilizando com base a notação científica. Um número em notação científica é representado por uma **mantissa**, uma

**base** e um **expoente**. Por exemplo, o número 1235 é representado por $0{,}1235 \times 10^4$, em que 0,1235 é a mantissa, 10 é a base e 4 é o expoente.

Existem três tipos de dados para números de ponto flutuante nas linguagens de programação, chamados de *float*, *double* e *long double*, cada um usando quantidades de bits diferentes para a mantissa e o expoente e 1 bit para representar o sinal (IEEE, 2017):

**Tabela 5 – Ponto flutuante.**

| Tipo | Mantissa | Expoente | Desvio do expoente |
|---|---|---|---|
| Float | 23 | 8 | 127 |
| Double | 52 | 11 | 1023 |
| Long double | 113 | 15 | 16383 |

Fonte: Elaborada pelo autor.

Como no caso dos números inteiros, o bit de sinal é o primeiro. É utilizado o valor 0 para números positivos e 1 para negativos.

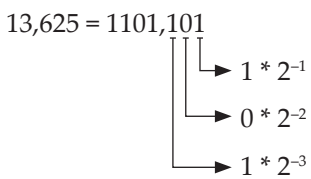
Em seguida, os próximos bits representam o expoente. Para defini-lo, soma-se o desvio (Tabela 5) ao seu valor e codifica-se o resultado em binário.

Por exemplo, para um expoente como 12 em um float, somaríamos 12 + 127 = 139 e utilizaríamos 139 em binário: 10001011. Para um expoente negativo, como -100, faríamos –100 + 127 = 7 e representaríamos esse valor em binário 00000111. Observe que, com essa estratégia, não utilizamos complemento de 2 para o expoente.

Por fim, podemos representar a mantissa. Ela é calculada sempre considerando um único dígito de casa decimal. Portanto, mesmo que forneçamos ao computador um número como 1234,56 ele converterá para $1{,}23456 \times 10^3$. A geração de uma fração em binário segue a mesma regra que em decimal, mas como agora a posição das casas está à direita da vírgula, elas assumem números negativos.

Por exemplo, veja como seria convertida a fração do número 13,625:

**Figura 7 – Fração.**




Fonte: Elaborada pelo autor.

Já que $2^{-1} = 0{,}5$ e $2^{-3} = 0{,}125$.

O número decimal 0,625 em binário seria representado como 101. Observe que o primeiro dígito é **sempre** 1. Na notação de ponto flutuante, ele não precisa ser representado. Assim, o número 0,625 em float seria representado da seguinte forma:

- **Dígito de sinal**: 0 (positivo),
- **Mantissa**: 1100. Como o primeiro 1 não precisa ser representado 100. São 23 bits, portanto 00000000000000000000100.
- **Expoente**: –3. Somado a 127 dá 130. Em binário, com 8 bits, é: 10000011.

O número final é: 0 10000011 00000000000000000000100.

## 5.3.3 Texto

Os primeiros sistemas utilizavam uma tabela para converter textos em caracteres. Essa tabela foi convencionada pela American Standard Code for Information Interchange e recebeu o nome de *tabela ASCII*.

**Tabela 6 – Tabela ASCII.**

| Decimal | Binário | Hex | Referência |
|---|---|---|---|
| 0 | 00000000 | 00 | Null - NUL |
| 1 | 00000001 | 01 | Start of Heading - SOH |
| 2 | 00000010 | 02 | Start of Text - STX |
| 3 | 00000011 | 03 | End of Text - ETX |
| 4 | 00000100 | 04 | End of Transmission - EOT |
| 5 | 00000101 | 05 | Enquiry - ENQ |
| 6 | 00000110 | 06 | Acknowledge - ACK |
| 7 | 00000111 | 07 | Bell, rings terminal bell - BEL |
| 8 | 00001000 | 08 | BackSpace - BS |
| 9 | 00001001 | 09 | Horizontal Tab - HT |
| 10 | 00001010 | 0A | Line Feed - LF |
| 11 | 00001011 | 0B | Vertical Tab - VT |
| 12 | 00001100 | 0C | Form Feed - FF |
| 13 | 00001101 | 0D | Enter - CR |
| 14 | 00001110 | 0E | Shift-Out - SO |
| 15 | 00001111 | 0F | Shift-In - SI |
| 16 | 00010000 | 10 | Data Link Escape - DLE |
| 17 | 00010001 | 11 | Device Control 1 - D1 |
| 18 | 00010010 | 12 | Device Control 2 - D2 |
| 19 | 00010011 | 13 | Device Control 3 - D3 |
| 20 | 00010100 | 14 | Device Control 4 - D4 |
| 21 | 00010101 | 15 | Negative Acknowledge - NAK |
| 22 | 00010110 | 16 | Synchronous idle - SYN |

| Decimal | Binário | Hex | Referência |
|---|---|---|---|
| 23 | 00010111 | 17 | End Transmission Block - ETB |
| 24 | 00011000 | 18 | Cancel line - CAN |
| 25 | 00011001 | 19 | End of Medium - EM |
| 26 | 00011010 | 1A | Substitute - SUB |
| 27 | 00011011 | 1B | Escape - ESC |
| 28 | 00011100 | 1C | File Separator - FS |
| 29 | 00011101 | 1D | Group Separator - GS |
| 30 | 00011110 | 1E | Record Separator - RS |
| 31 | 00011111 | 1F | Unit Separator - US |
| 32 | 00100000 | 20 | Space - SPC |
| 33 | 00100001 | 21 | ! |
| 34 | 00100010 | 22 | “ |
| 35 | 00100011 | 23 | # |
| 36 | 00100100 | 24 | $ |
| 37 | 00100101 | 25 | % |
| 38 | 00100110 | 26 | & |
| 39 | 00100111 | 27 | ‘ |
| 40 | 00101000 | 28 | ( |
| 41 | 00101001 | 29 | ) |
| 42 | 00101010 | 2A | * |
| 43 | 00101011 | 2B | + |
| 44 | 00101100 | 2C | , |
| 45 | 00101101 | 2D | - |
| 46 | 00101110 | 2E | . |
| 47 | 00101111 | 2F | / |
| 48 | 00110000 | 30 | 0 |
| 49 | 00110001 | 31 | 1 |
| 50 | 00110010 | 32 | 2 |
| 51 | 00110011 | 33 | 3 |
| 52 | 00110100 | 34 | 4 |
| 53 | 00110101 | 35 | 5 |
| 54 | 00110110 | 36 | 6 |
| 55 | 00110111 | 37 | 7 |
| 56 | 00111000 | 38 | 8 |

| Decimal | Binário | Hex | Referência |
|---|---|---|---|
| 57 | 00111001 | 39 | 9 |
| 58 | 00111010 | 3A | : |
| 59 | 00111011 | 3B | ; |
| 60 | 00111100 | 3C | < |
| 61 | 00111101 | 3D | = |
| 62 | 00111110 | 3E | > |
| 63 | 00111111 | 3F | ? |
| 64 | 01000000 | 40 | @ |
| 65 | 01000001 | 41 | A |
| 66 | 01000010 | 42 | B |
| 67 | 01000011 | 43 | C |
| 68 | 01000100 | 44 | D |
| 69 | 01000101 | 45 | E |
| 70 | 01000110 | 46 | F |
| 71 | 01000111 | 47 | G |
| 72 | 01001000 | 48 | H |
| 73 | 01001001 | 49 | I |
| 74 | 01001010 | 4A | J |
| 75 | 01001011 | 4B | K |
| 76 | 01001100 | 4C | L |
| 77 | 01001101 | 4D | M |
| 78 | 01001110 | 4E | N |
| 79 | 01001111 | 4F | O |
| 80 | 01010000 | 50 | P |
| 81 | 01010001 | 51 | Q |
| 82 | 01010010 | 52 | R |
| 83 | 01010011 | 53 | S |
| 84 | 01010100 | 54 | T |
| 85 | 01010101 | 55 | U |
| 86 | 01010110 | 56 | V |
| 87 | 01010111 | 57 | W |
| 88 | 01011000 | 58 | X |
| 89 | 01011001 | 59 | Y |
| 90 | 01011010 | 5A | Z |

| Decimal | Binário | Hex | Referência |
|---|---|---|---|
| 91 | 01011011 | 5B | [ |
| 92 | 01011100 | 5C | \ |
| 93 | 01011101 | 5D | ] |
| 94 | 01011110 | 5E | ^ |
| 95 | 01011111 | 5F | _ |
| 96 | 01100000 | 60 | ` |
| 97 | 01100001 | 61 | a |
| 98 | 01100010 | 62 | b |
| 99 | 01100011 | 63 | c |
| 100 | 01100100 | 64 | d |
| 101 | 01100101 | 65 | e |
| 102 | 01100110 | 66 | f |
| 103 | 01100111 | 67 | g |
| 104 | 01101000 | 68 | h |
| 105 | 01101001 | 69 | i |
| 106 | 01101010 | 6A | j |
| 107 | 01101011 | 6B | k |
| 108 | 01101100 | 6C | l |
| 109 | 01101101 | 6D | m |
| 110 | 01101110 | 6E | n |
| 111 | 01101111 | 6F | o |
| 112 | 01110000 | 70 | p |
| 113 | 01110001 | 71 | q |
| 114 | 01110010 | 72 | r |
| 115 | 01110011 | 73 | s |
| 116 | 01110100 | 74 | t |
| 117 | 01110101 | 75 | u |
| 118 | 01110110 | 76 | v |
| 119 | 01110111 | 77 | w |
| 120 | 01111000 | 78 | x |
| 121 | 01111001 | 79 | y |
| 122 | 01111010 | 7A | z |
| 123 | 01111011 | 7B | { |
| 124 | 01111100 | 7C | \| |

| Decimal | Binário | Hex | Referência |
|---|---|---|---|
| 125 | 01111101 | 7D | } |
| 126 | 01111110 | 7E | ~ |
| 127 | 01111111 | 7F | Delete |
| 128 | 10000000 | 80 | Ç |
| 129 | 10000001 | 81 | ü |
| 130 | 10000010 | 82 | é |
| 131 | 10000011 | 83 | â |
| 132 | 10000100 | 84 | ä |
| 133 | 10000101 | 85 | à |
| 134 | 10000110 | 86 | å |
| 135 | 10000111 | 87 | ç |
| 136 | 10001000 | 88 | ê |
| 137 | 10001001 | 89 | ë |
| 138 | 10001010 | 8A | è |
| 139 | 10001011 | 8B | ï |
| 140 | 10001100 | 8C | î |
| 141 | 10001101 | 8D | ì |
| 142 | 10001110 | 8E | Ä |
| 143 | 10001111 | 8F | Å |
| 144 | 10010000 | 90 | É |
| 145 | 10010001 | 91 | æ |
| 146 | 10010010 | 92 | Æ |
| 147 | 10010011 | 93 | ô |
| 148 | 10010100 | 94 | ö |
| 149 | 10010101 | 95 | ò |
| 150 | 10010110 | 96 | û |
| 151 | 10010111 | 97 | ù |
| 152 | 10011000 | 98 | ÿ |
| 153 | 10011001 | 99 | Ö |
| 154 | 10011010 | 9A | Ü |
| 155 | 10011011 | 9B | ø |
| 156 | 10011100 | 9C | £ |
| 157 | 10011101 | 9D | Ø |
| 158 | 10011110 | 9E | × |

| Decimal | Binário | Hex | Referência |
|---|---|---|---|
| 159 | 10011111 | 9F | ƒ |
| 160 | 10100000 | A0 | á |
| 161 | 10100001 | A1 | ù |
| 162 | 10100010 | A2 | ó |
| 163 | 10100011 | A3 | ú |
| 164 | 10100100 | A4 | ñ |
| 165 | 10100101 | A5 | Ñ |
| 166 | 10100110 | A6 | ª |
| 167 | 10100111 | A7 | º |
| 168 | 10101000 | A8 | ¿ |
| 169 | 10101001 | A9 | ® |
| 170 | 10101010 | AA | ¬ |
| 171 | 10101011 | AB | ½ |
| 172 | 10101100 | AC | ¼ |
| 173 | 10101101 | AD | ¡ |
| 174 | 10101110 | AE | « |
| 175 | 10101111 | AF | » |
| 176 | 10110000 | B0 | ░ |
| 177 | 10110001 | B1 | ▒ |
| 178 | 10110010 | B2 | ▓ |
| 179 | 10110011 | B3 | │ |
| 180 | 10110100 | B4 | ┤ |
| 181 | 10110101 | B5 | Á |
| 182 | 10110110 | B6 | Â |
| 183 | 10110111 | B7 | À |
| 184 | 10111000 | B8 | © |
| 185 | 10111001 | B9 | ╣ |
| 186 | 10111010 | BA | ║ |
| 187 | 10111011 | BB | ╗ |
| 188 | 10111100 | BC | ╝ |
| 189 | 10111101 | BD | ¢ |
| 190 | 10111110 | BE | ¥ |
| 191 | 10111111 | BF | ┐ |
| 192 | 11000000 | C0 | └ |

| Decimal | Binário | Hex | Referência |
|---|---|---|---|
| 193 | 11000001 | C1 | ┴ |
| 194 | 11000010 | C2 | ┬ |
| 195 | 11000011 | C3 | ├ |
| 196 | 11000100 | C4 | ─ |
| 197 | 11000101 | C5 | ┼ |
| 198 | 11000110 | C6 | ã |
| 199 | 11000111 | C7 | Ã |
| 200 | 11001000 | C8 | ╚ |
| 201 | 11001001 | C9 | ╔ |
| 202 | 11001010 | CA | ╩ |
| 203 | 11001011 | CB | ╦ |
| 204 | 11001100 | CC | ╠ |
| 205 | 11001101 | CD | ═ |
| 206 | 11001110 | CE | ╬ |
| 207 | 11001111 | CF | ¤ |
| 208 | 11010000 | D0 | ð |
| 209 | 11010001 | D1 | Ð |
| 210 | 11010010 | D2 | Ê |
| 211 | 11010011 | D3 | Ë |
| 212 | 11010100 | D4 | È |
| 213 | 11010101 | D5 | ı |
| 214 | 11010110 | D6 | Í |
| 215 | 11010111 | D7 | Î |
| 216 | 11011000 | D8 | Ï |
| 217 | 11011001 | D9 | ┘ |
| 218 | 11011010 | DA | ┌ |
| 219 | 11011011 | DB | █ |
| 220 | 11011100 | DC | ▄ |
| 221 | 11011101 | DD | ¦ |
| 222 | 11011110 | DE | Ì |
| 223 | 11011111 | DF | ▀ |
| 224 | 11100000 | E0 | Ó |
| 225 | 11100001 | E1 | ß |
| 226 | 11100010 | E2 | Ô |

| Decimal | Binário | Hex | Referência |
|---|---|---|---|
| 227 | 11100011 | E3 | Ò |
| 228 | 11100100 | E4 | õ |
| 229 | 11100101 | E5 | Õ |
| 230 | 11100110 | E6 | µ |
| 231 | 11100111 | E7 | þ |
| 232 | 11101000 | E8 | Þ |
| 233 | 11101001 | E9 | Ú |
| 234 | 11101010 | EA | Û |
| 235 | 11101011 | EB | Ù |
| 236 | 11101100 | EC | ý |
| 237 | 11101101 | ED | Ý |
| 238 | 11101110 | EE | ¯ |
| 239 | 11101111 | EF | ´ |
| 240 | 11110000 | F0 | |
| 241 | 11110001 | F1 | ± |
| 242 | 11110010 | F2 | ‗ |
| 243 | 11110011 | F3 | ¾ |
| 244 | 11110100 | F4 | ¶ |
| 245 | 11110101 | F5 | § |
| 246 | 11110110 | F6 | ÷ |
| 247 | 11110111 | F7 | ¸ |
| 248 | 11111000 | F8 | ° |
| 249 | 11111001 | F9 | ¨ |
| 250 | 11111010 | FA | · |
| 251 | 11111011 | FB | ¹ |
| 252 | 11111100 | FC | ³ |
| 253 | 11111101 | FD | ² |
| 254 | 11111110 | FE | ■ |
| 255 | 11111111 | FF | |

Fonte: ASCII, 2017.

De acordo com essa tabela, para representar a letra “A” maiúscula bastaria utilizar o número 65 binário 01000001. Observe que na Figura 7 existem alguns caracteres especiais para representar quebras de linhas, bordas de tabelas ou letras acentuadas.

O problema ocorreu quando os computadores começaram a ser vendidos internacionalmente. Percebeu-se que esse conjunto de letras era muitíssimo pequeno, já que outros idiomas incluíam acentuação, ou caracteres diferentes, como os vários kanjis japoneses.

Posteriormente, criou-se um padrão ainda maior de caracteres chamado *Unicode*, utilizando-se 2 bytes por letra. Com 2 bytes, a tabela passou a incluir 65535 símbolos e pode incluir caracteres japoneses, árabes, indígenas, vários símbolos úteis (como os emojis) e até símbolos matemáticos. Os primeiros 255 caracteres da tabela correspondem aos mesmos da tabela ASCII.

A tabela completa pode ser consultada on-line. Disponível em: <https://unicode-table.com/pt/>. Acesso em: 15 dez. 2017.

## 5.3.4 Imagens

As câmeras comuns, como as de celulares, captam o campo da luz visível de maneira similar aos nossos olhos, isto é, por meio de sensores fotossensíveis, que agem como receptores dos comprimentos de onda de verde, vermelho e azul (RGB). A informação de cada sensor é chamada de *canal de cor*.

A intensidade da luz que atinge o sensor é armazenada pelo computador na forma de um número, geralmente 0 para a ausência de luz e 255 para o brilho máximo que a câmera é capaz de capturar.

Em imagens em tons de cinza, há apenas um único sensor, que irá armazenar os valores na forma de uma tabela de números. Essa tabela apresentará um número para cada pixel na imagem gerada pela câmera.

Já em câmeras coloridas, a informação será armazenada em três tabelas diferentes, uma para cada canal.

Algumas imagens ainda têm um quarto canal, chamado de alfa. Esse canal possui um número que indica como deve-se misturar a cor com alguma outra imagem já desenhada. É usado para dar a sensação de transparência. Cada canal de cor geralmente é representado por um byte.

A informação é disposta na memória no padrão ARGB ou RGB. Ou seja, os bytes dos canais alfa (se houver) são colocados na sequência vermelho, verde e azul, respectivamente. Em seguida, passa-se para o próximo pixel. Por isso, não é incomum ver cores representadas na internet em hexadecimal. Por exemplo, o verde pode ser descrito como #00FF00.

Porém, essa informação consome muito espaço. Para se ter uma ideia, uma imagem de 1920 × 1080 pixels, contendo os canais R, G e B, ocupa:

$$1920 \times 1080 \times 3 = 6.220.800 = 6MB$$

Agora, lembre-se que um vídeo grava 24 imagens em um único segundo, ou seja, equivale a 149 MB ou 8 GB em um minuto! Obviamente, seria impraticável armazenar toda essa informação em seu estado bruto.

Por isso, os cientistas da computação inventaram diferentes **formatos de imagem**, que otimizam o armazenamento dessas informações por meio de diferentes estratégias de **compressão**.

Os formatos de imagem definem:

- quais canais de cor serão armazenados, com qual densidade e em qual ordem;
- que tipo de compressão será utilizada;
- metadados relacionados à imagem (data, coordenadas GPS etc.);
- se o armazenamento é de uma imagem isolada ou de vídeo.

O princípio básico por trás da compressão está em eliminar informação redundante de uma imagem ou otimizar o uso do seu armazenamento. As estratégias comuns para isso são:

- **Uso de índices de cores** – com três bytes, um para canal, os canais vermelho (R), verde (G) e azul (B) podem representar 16 milhões de cores diferentes. Porém, ao olhar para uma única imagem, logo percebemos que muitas cores não estão presentes. Uma das formas de otimizar o armazenamento da imagem é criar uma tabela contendo somente a informação das cores efetivamente presentes na imagem. Em seguida, em vez de armazenarmos a informação da cor, bastaria armazenarmos em que linha dessa tabela a cor real poderia ser encontrada (índice). Essa estratégia é chamada de imagem indexada. É a estratégia usada nos formatos GIF e PNG.
- **Compressão** – é possível também otimizar o armazenamento de imagens evitando duplicar informações iguais que estejam dispostas lado a lado. Por exemplo, uma imagem escura terá vários pixels pretos, um ao lado do outro. Um formato de arquivo com compressão poderia guardar apenas a informação de quantos pixels são, e, em seguida, sua cor, poupando muito espaço. Programas mais inteligentes poderiam até encontrar e comprimir padrões de repetição de cor. Esse tipo de compressão é similar ao que fazemos em arquivos, quando utilizamos software como o *WinZip*. Compressão é utilizada nos formatos JPG, TIFF e no formato de vídeo MP4.

Novamente, ao observamos uma imagem, podemos reparar que boa parte dos tons de uma mesma cor não são distinguíveis para os seres humanos. Por exemplo, você é capaz de diferenciar a olho nu o verde de RGB 200, 1, 0 do verde 200, 0, 0?

Com base nessa informação, alguns formatos de imagem simplificam a informação antes de salvá-la, equalizando tons similares. Embora o resultado seja muitas vezes imperceptível aos olhos, essa estratégia melhora muito os algoritmos de compressão. Se estivermos usando uma imagem indexada, poderemos ter uma tabela de cores menor e, portanto, usar menos bits para representar seus índices. Se estivermos usando compressão, teríamos um número maior de padrões. Porém essa estratégia efetivamente **destrói parte da informação**. Por isso,

esse tipo de estratégia é chamada *de compressão com perdas (Lossy)*. Quando não a utilizamos, estamos nos referindo à *compressão sem perdas* (*Lossless*).

O formato JPG utiliza algoritmos de compressão com perdas. A agressividade para a equalização de cores pode ser regulada; em níveis mais altos, ela pode se tornar visível.

Algoritmos de compressão de vídeo funcionam de maneira similar, mas também podem identificar padrões entre dois quadros de cena. Isso é melhor, já que quadros com a câmera estacionária, ou com pouco movimento, terão vários elementos em comum. Entretanto, ao levarmos em consideração o tamanho dos vídeos, vemos que a maior parte utiliza estratégias de compressão com perdas.

## Atividades

**1.** Utilizando o conhecimento adquirido em bases numéricas, explique porque um sistema 32 bits não é capaz de endereçar mais do que 4 GB em memória.

**2.** Converta os seguintes números binários para decimal e hexadecimal.

a. 101. Decimal: _____ Hexadecimal: _____

b. 1010100. Decimal: _____ Hexadecimal: _____

c. 11111111. Decimal: _____ Hexadecimal: _____

**3.** Converta os números decimais para binário e hexadecimal:

a. 127: Binário: _____ Hexadecimal: _____

b. 2842: Binário: _____ Hexadecimal: _____

c. 3856: Binário: _____ Hexadecimal: _____

**4.** Converta os números hexadecimais em binário e decimal:

a. 0xABC: Binário: _____ Decimal: _____

b. 0xF7: Binário: _____ Decimal: _____

c. 0xAA: Binário: _____ Decimal: _____

**5.** Como podemos saber se um número binário positivo é par ou ímpar?

**6.** Quanto tempo levará a transmissão de um arquivo de 4MB numa rede de 2Mb por segundo?

## Referências

ASCII table. Disponível em: <http://www.theasciicode.com.ar>. Acesso em: 15 dez. 2017.

IBM. **Color Graphics/Monitor Adapter (CGA) Reference**. Disponível em: <http://www.minuszero degrees.net/oa/OA%20-%20IBM%20Color%20Graphics%20Monitor%20Adapter%20(CGA).pdf>. Acesso em: 5 ago. 2017.

IEEE. **IEEE Standard for Floating Pointing Arithmetic**. Disponível em: <http://ieeexplore.ieee.org/document/4610935/>. Acesso em: 15 dez. 2017.

ISO/IEC. Especificação dos números inteiros em C: ISO/IEC 9899 – The C Language Standard. **Committee Draft**, 7 set. 2007. Disponível em: <http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1256.pdf>. Acesso em: 15 dez. 2017.

JOSHI, P; ESCRIVÁ, D; GODOY, V. **Open CV by example**. [S.l.]: Packt Publishing, 2016.

VELLOSO, F. C. **Informática**: conceitos básicos. Rio de Janeiro: Ed. Campus, 1994.

## Resolução

**1.** Um sistema de 32 bits terá seu endereçamento de também com 32 bits. Como cada endereço é um número sequencial, e representa um byte na memória, a maior quantidade de bytes que pode ser representada com um número de 32 bits é $2^{32}$, ou seja, 4.194.304 (4 GB).

**2.** a: 3 e 3 b: 84 e 54 c: 255 e FF.

**3.** a: 1111111 e 7F b: 101100011010 e B1A c: 111100010000 e F10.

**4.** a: 101010111100 e 2748 b: 11110111 e 247 c: 10101010 e 170.

**5.** Pelo seu último dígito. Se for 0 o número será par, se for 1, ímpar.

**6.** 4 megabytes equivalem a 4x8 = 32 megabits. Considerando uma taxa de transmissão de 2 megabits por segundo o tempo será 32/2 = 16 segundos.

# 6
# Bits, bytes e hexadecimal - A informação no computador

Sabemos que todo computador é formado por hardware e software. O hardware é formado pelo processador, memórias, periféricos (teclado, mouse etc.) e demais circuitos eletrônicos. Já os softwares fornecem funcionalidades, como planilhas eletrônicas, jogos digitais e até mesmo o editor de textos.

Entre eles, porém, há um conjunto de software especial, chamado *sistema operacional*. Neste capítulo, veremos quais são os objetivos desse sistema e porque ele é tão importante para o computador.

Vídeo

# 6.1 Breve histórico

Nos anos 1940, os computadores eram capazes de executar apenas um único programa. Ele rodava sozinho e tinha total controle sobre o sistema. Desde a carga do programa na memória até o controle dos dispositivos de saída, todos eram programados pelo desenvolvedor.

Os programadores dessa época começaram a criar rotinas de código que cuidavam da parte básica. Nos anos 1950, os computadores já dispunham de uma série de bibliotecas de sistema, ou seja, pequenas rotinas prontas, para facilitar a programação das aplicações. Além disso, os computadores possuíam um programa chamado *monitor do sistema*, capaz de carregar e descarregar aplicações da memória. Ainda assim, o programador tinha de escrever muito código relacionado ao hardware e poderia, se quisesse, ignorar completamente as rotinas pré-escritas e escrever as próprias (MAZIERO, 2017).

O primeiro sistema operacional, chamado de *GM-NAA I/O* nasceu no fim da década de 1950, produzido pela divisão de pesquisa da General Motors para o IBM 704. Sua principal função era a carga de programas na memória e a gerência dos dispositivos de entrada e saída, mas ainda era muito focado apenas no hardware que iria atender, que era completamente diferente de computador para computador.

As coisas começaram a mudar quando a IBM, principal fabricante de mainframes[1] da época, decidiu criar um sistema operacional que atendesse a todos os seus novos computadores. Esse sistema, conhecido como *OS/360,* demandou diversos esforços da empresa (BROOKS, 1995). Porém, devido às diferenças enormes entre as várias famílias de computadores, o sistema acabou por ter três versões bem diferentes. Além disso, à época, surgiu outro sistema operacional também importante, chamado de *DOS/360*.

O OS/360 foi lançado em 1964. Ele incluía conceitos de outros sistemas lançados na época, como (TANEMBAUM, 2016):

- Multiprogramação – a capacidade de interromper um programa em execução quando ele esperava pelo acesso a periféricos, como disco ou fita. Esse conceito foi introduzido pelo sistema britânico Leo III, em 1961.
- Spooling – criação de uma fila de tarefas para dispositivos lentos, como fitas magnéticas ou impressoras.
- Time-sharing – permitia que vários programas dividissem o tempo do processador e rodassem simultaneamente. Embora o primeiro sistema a introduzir esse conceito, chamado *CTSS*, tenha sido criado em 1962 por Fernando Corbató, o recurso só foi incluído em 1971, no OS/360.

Em 1965, outro sistema operacional, chamado *Multics*, surgiu. Ele foi uma parceria entre o Massachusetts Institute of Technology (MIT), a General Eletric (GE) e o Bell Labs. Esse sistema influenciou vários sucessores modernos, inclusive os sistemas Unix, Linux e Windows. Ele incluiu o conceito de **memória virtual** – que fazia com que programas não conhecessem os endereços reais de memória nos quais trabalhavam – e o conceito de **vinculação dinâmica**

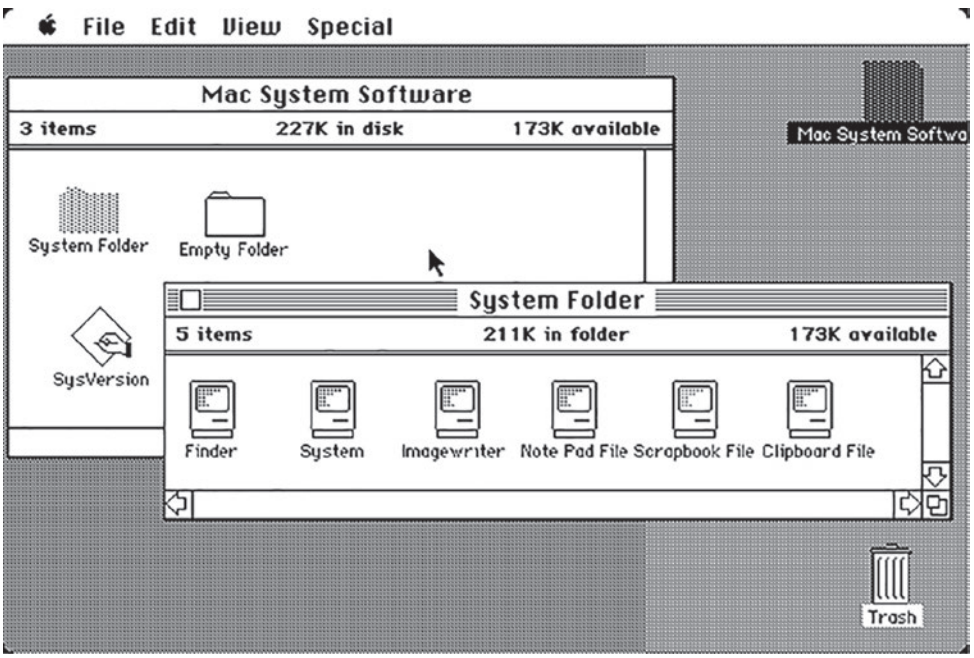
1 Computadores de grande porte destinados a grandes volumes de processamento de informações.

(dinamic linking), que permitia aos programas em execução carregar bibliotecas de software adicionais e as executar. Além disso, foi um dos primeiros sistemas operacionais a incluir medidas de segurança, como impedir que áreas inválidas da memória fossem acessadas e criar um **sistema de arquivos hierárquico**, baseado em pastas (TANEMBAUM, 2016).

A partir do Multics, vários outros sistemas operacionais foram criados. A seguir, um resumo:

- 1969 – o Bell Labs lançou o sistema Unix, criado por Dennies Richie e Ken Thompson.
- 1981 – a Microsoft lançou o MS-DOS. O sistema foi desenvolvido pela empresa Seattle Computers, em 1980. A grande inovação do MS-DOS, entretanto, estava no fato de que Bill Gates foi um dos primeiros a vislumbrar o potencial de vender o sistema operacional como uma parte separada do hardware que o acompanhava.
- 1984 – a Apple lançou o Macintosh System Software 1.0 (Figura 1), o primeiro sistema operacional com interface gráfica incorporada. Um sucesso de vendas.

**Figura 1 – Macintosh System Software 1.0.**




Fonte: Wikimedia Commons.

- 1985 – a Microsoft lançou o Windows 1.0, também com interface gráfica.
- 1987 – o professor Tanembaum criou o sistema Minix (TANEMBAUM, 2016). O sistema foi elaborado com base em Unix para fins didáticos. Com ele, seus alunos poderiam substituir partes do sistema, como a forma em que o HD trabalha, e verificar as diferenças.
- 1991 – o estudante holandês Linus Torvalds lançou o sistema operacional Linux, de código aberto. O sistema rapidamente ganha várias distribuições e se espalha pelo mundo.

- Em 1996, a Palm lançou o Palm OS (Figura 2), um dos primeiros sistemas operacionais para dispositivos móveis.

**Figura 2 – Palm OS, exibido no emulador (POSE).**


Fonte: Wikimedia Commons.

- 1997 – foi lançado o Symbian, usado pelos smartphones Nokia, Ericsson e Motorola.
- 2007 – a Apple lançou o sistema operacional iOS, para seus smartphones e, posteriormente, seus tablets.
- 2008 – o Google lançou o sistema operacional Android, baseado no Núcleo Linux. A proposta do Android era ser uma plataforma mais aberta que o iOS e disponível para qualquer fabricante de hardware.

Vídeo

## 6.2 Funcionamento

Para entender um sistema operacional, devemos compreender conceitos-chave que norteiam seu funcionamento. Esse entendimento é crítico para qualquer profissional de TI, já que é sobre ele que todos os programas desenvolvidos serão executados.

### 6.2.1 Objetivos de um sistema operacional

Um sistema operacional tem dois objetivos principais (MAZIERO, 2017), o primeiro é o da **abstração de recursos**. Considere, por exemplo, a tarefa de acessar uma foto em seu computador. Ela pode estar no disco rígido, no pendrive ou em um CD.

Cada um desses dispositivos exigiria uma sequência de hardware completamente diferente para ser acessado. Cabe ao sistema operacional conhecer essa sequência e fornecer aos programadores uma interface mais simples para acessá-los, como os comandos open, read,

write e close. A abstração também traz como vantagem o fato de que software e hardware podem evoluir de maneira independente.

O segundo objetivo de um sistema operacional é o da **gerência de recursos**. Um computador precisa executar diversas tarefas ao mesmo tempo, por exemplo, rodar o antivírus, baixar um arquivo da internet, tudo isso enquanto o usuário escreve um texto no Word e ouve música no Spotify. Muitas dessas tarefas farão uso concorrente do hardware, por isso, conflitos podem ocorrer quando dois ou mais aplicativos tentam usar o mesmo recurso ao mesmo tempo.

É a gerência de recursos que permite ao computador executar um conjunto maior de tarefas simultâneas do que o número de processadores que sua máquina possui. O sistema reveza o tempo do processador entre as várias tarefas, permitindo que cada uma rode um pouco, dando a sensação de paralelismo – técnica conhecida como *time sharing (*TANEMBAUM, 2016).

É também a gerência de recursos que impede o conflito entre recursos que não possam ser compartilhados, como a impressora e/ou os autofalantes. No caso da impressora, o sistema pode até gerenciar uma fila de impressão. No caso do áudio, o sistema consegue parar um som de maneira suave para executar uma notificação e retomar o áudio que estava tocando.

Os objetivos de um sistema operacional são cumpridos por meio de um conjunto de **políticas** e **mecanismos**. As políticas definem as decisões de projeto relativas àquele sistema, por exemplo, quanto tempo um programa deve rodar no processador antes de outro tomar seu lugar. E os mecanismos representam o software criado para que as políticas sejam possíveis.

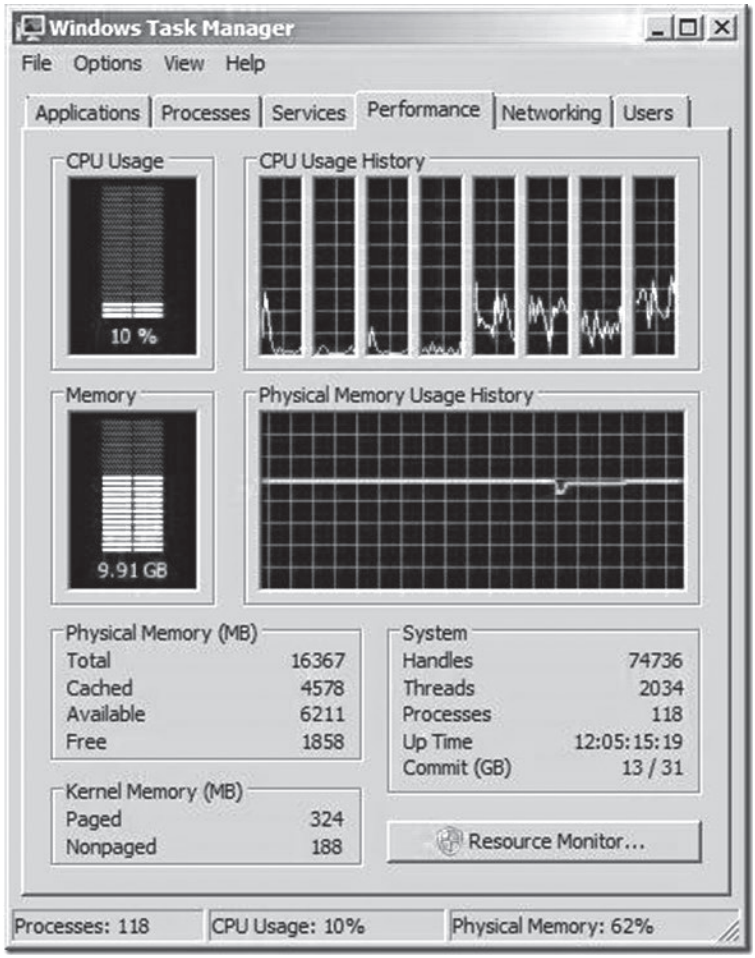
## 6.2.2 Recursos gerenciados

Como vimos, a gerência de recursos é um aspecto importante de qualquer sistema operacional. Mas você pode estar se perguntando: quais recursos são esses? O que o sistema gerencia?

A seguir, listamos os recursos principais:

- Processadores – o sistema precisa distribuir o tempo de processamento dos processadores da máquina de maneira justa. Isso impede que um programa trave o sistema inteiro ou que outro demore muito tempo para executar. Além disso, ele precisa aproveitar oportunidades para processamento, como trocar o programa em execução, caso ele esteja esperando recursos (como a leitura do disco).

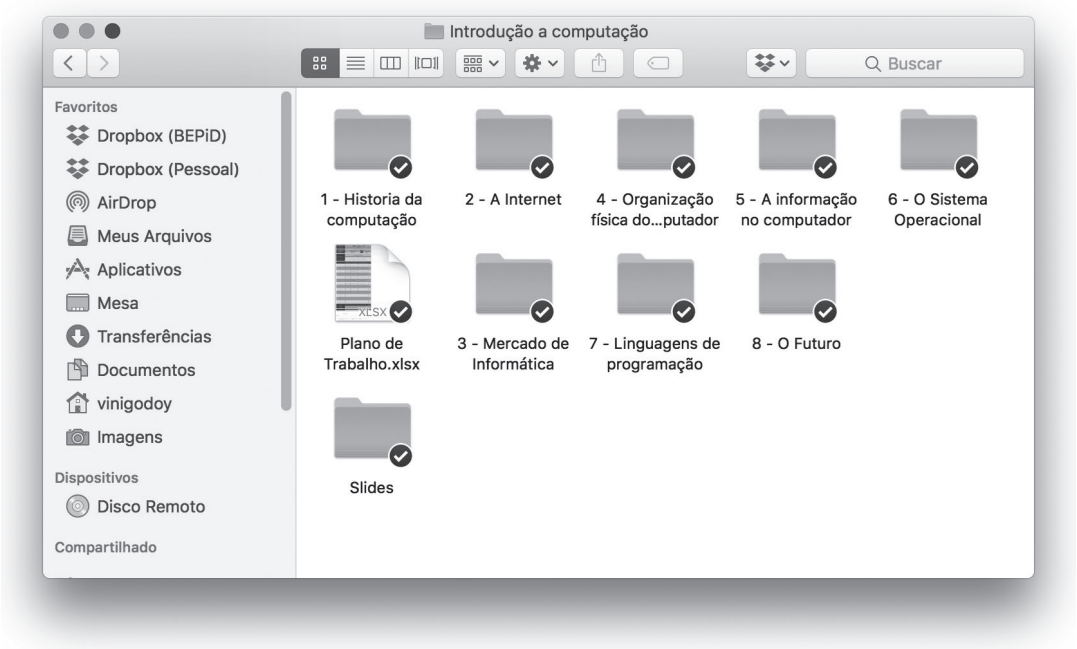
**Figura 3 – Gerenciador de tarefas exibindo o uso de processadores.**




Fonte: Microsoft Windows.

- Memória – criar uma área de memória específica para cada programa, de modo que um não interfira em outro, é tarefa do sistema. Isso impede, por exemplo, que um programa leia dados de outro de maneira maliciosa. Além disso, as áreas de memória devem ser gerenciadas de acordo com sua velocidade, com o sistema priorizando memórias rápidas para aplicações em execução e guardando dados de aplicações em segundo plano (minimizadas, por exemplo) em memórias lentas, como o disco rígido.
- Arquivos – cria o conceito de arquivos, ou seja, um conjunto de dados bem estruturado. E de diretórios (pastas), contendo arquivos dentro.

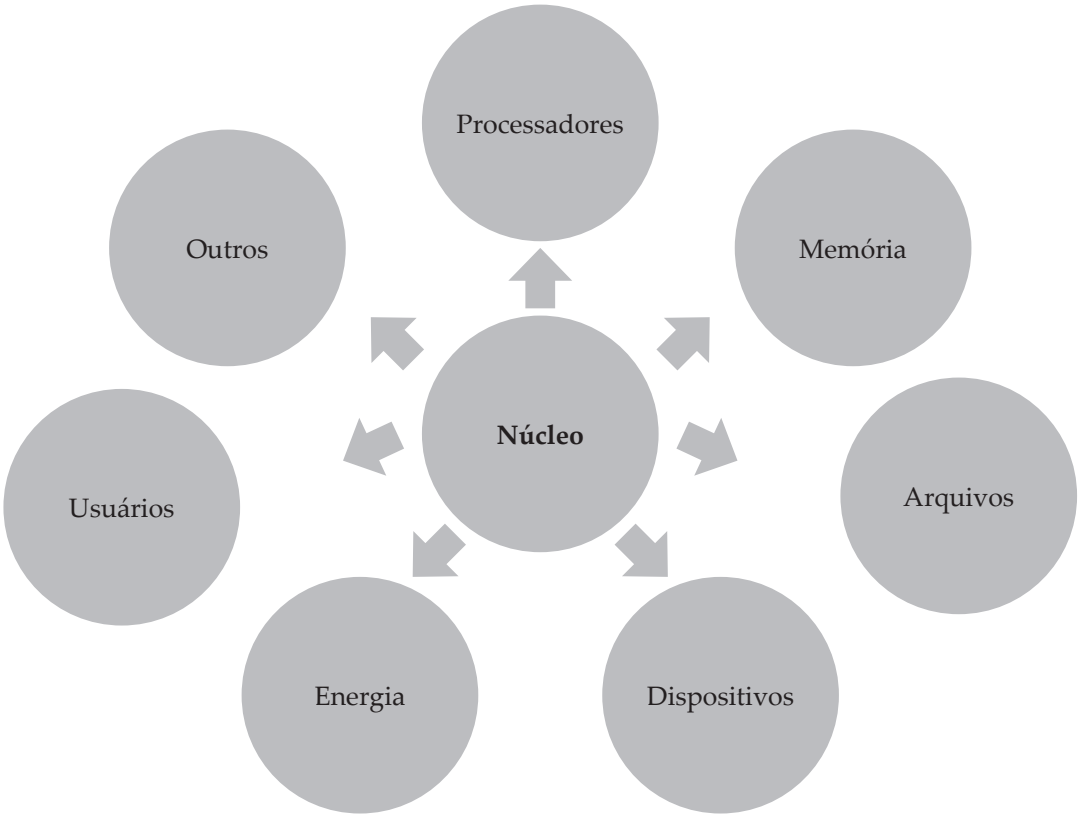
**Figura 4 – Arquivos e diretórios vistos no Finder do MacOS.**




Fonte: Elaborada pelo autor.

- Dispositivos – o computador possui muitos hardwares: vários tipos de disco rígido, leitores de CD, teclados, monitores etc., que vêm de diferentes fabricantes. A gestão dos dispositivos envolve definir para o programador de maneira clara como um determinado tipo de dispositivo funciona, independentemente do fabricante ou do hardware específico. As especificidades desses dispositivos estão em programas chamados de *drivers*, que são implementados pelo fabricante do hardware. Por exemplo, um tipo de dispositivo definido pode ser uma unidade de armazenamento, porém pouco importa para o usuário se ela é um disco rígido ou um pendrive.
- Usuários – o sistema operacional precisa definir quais usuários existem e garantir que recursos, como arquivos, sejam acessados apenas pelas pessoas ou softwares certos. Isso é especialmente relevante em sistemas de rede, que permitem o acesso remoto. Um bom sistema permite a criação de políticas de privacidade, a definição de usuários para diferentes papéis (uso, manutenção, administração) e até mesmo atribuições de permissão em condições específicas.
- Energia – é cada vez mais relevante a gestão de energia, sobretudo em notebooks, celulares ou tablets. O sistema deve ser capaz de monitorar a bateria, diminuir a luminosidade se uma fonte de energia estiver desconectada e, até mesmo, diminuir a velocidade do computador para prolongar a vida de bateria, se necessário. Além disso, o sistema deve ser capaz de guardar os dados em local seguro, para permitir o desligamento ou a hibernação, caso necessário.

**Figura 5 – Recursos gerenciados pelo sistema operacional.**




Fonte: Elaborada pelo autor.

Obviamente, sistemas operacionais modernos gerenciam uma série de outros recursos importantes, como redes (quais estão disponíveis, tráfego), áudio etc.

Vídeo

## 6.3 Arquiteturas

Como vimos, um sistema operacional é formado por diversas partes diferentes. Assim como um carro, os engenheiros do sistema podem montar essas partes de diferentes formas, priorizando flexibilidade, velocidade ou segurança. Nesta seção, estudaremos algumas dessas organizações possíveis.

### 6.3.1 Sistemas monolíticos

O tipo mais simples de arquitetura de sistema é a monolítica. Nela, todos os componentes do sistema têm total acesso a todos os recursos e à memória, ou seja, executam em modo núcleo.

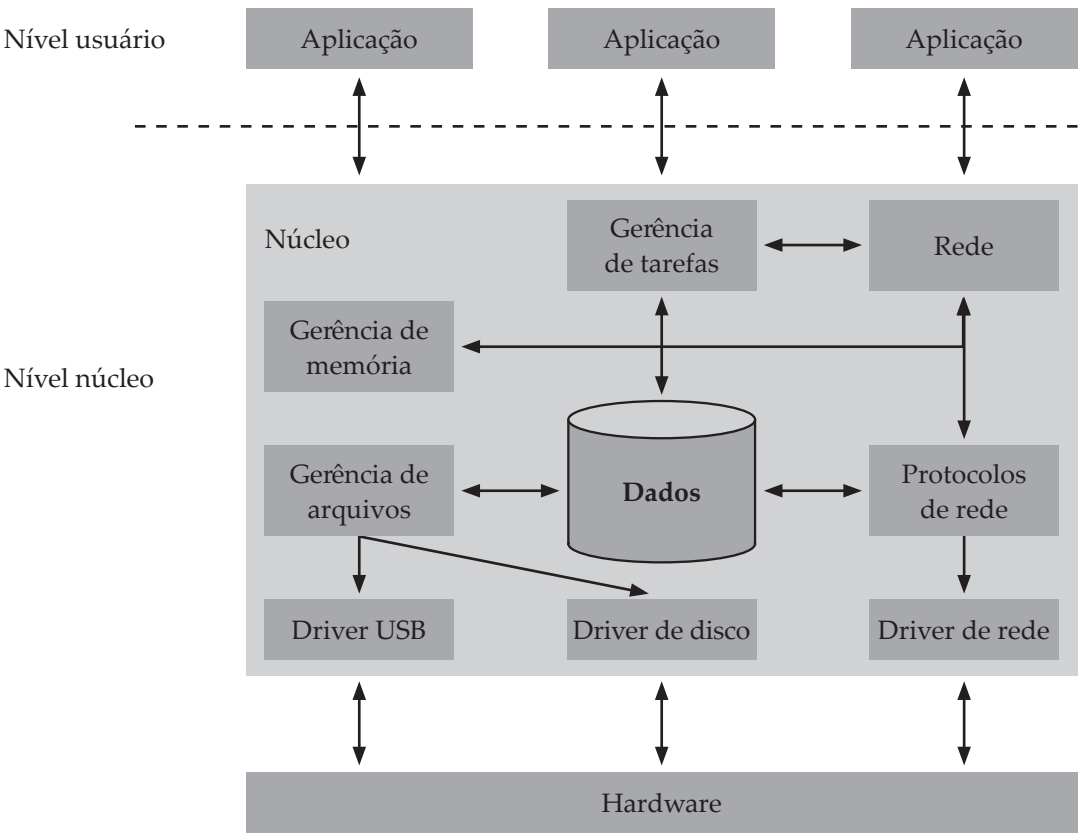
Essa arquitetura tem como grande vantagem sua simplicidade de construção, seu desempenho e o tamanho do binário. O sistema não perde tempo verificando políticas, permissões ou qualquer outro tipo de barreira. Essa construção simplificada também torna o tamanho final do sistema operacional pequeno. Entretanto essa arquitetura também carrega algumas desvantagens bastante graves.

Embora sua facilidade inicial de construção seja uma vantagem, é um sistema cuja manutenção é traumática, afinal os componentes ficam fortemente acoplados entre si. Isso dificulta a inclusão de novos componentes no sistema ou mesmo a adição de funcionalidades novas em componentes existentes. Além disso, a falha de um componente pode rapidamente levar o sistema ao colapso – levando a travamentos e reinicializações.

Esse tipo de arquitetura foi muito empregada nos primeiros sistemas operacionais, mas se manteve ativo até em sistemas como o UNIX e o MS-DOS. O núcleo do sistema Linux também é monolítico, mas essa arquitetura está sendo substituída para a de micronúcleo desde a versão 2.0.

Há nos chats da Usenet (DIBONA, 1999), inclusive, uma discussão famosa entre Linus Torvalds – criador do Linux – e Andrew Tanembaum – um dos criadores do Minix –, em que Linus defendia essa arquitetura enquanto Tanembaum o criticava por não optar pela arquitetura micronúcleo.

**Figura 6 – Arquitetura monolítica: alto acoplamento.**




Fonte: Elaborada pelo autor.

## 6.3.2 Sistemas micronúcleo (microkernel)

A arquitetura micronúcleo (ou *microkernel*, do inglês) consiste em manter no núcleo do sistema somente o código de baixo nível para acesso ao hardware e às abstrações essenciais para o funcionamento do sistema (MAZIERO, 2017).

Nessa arquitetura, por exemplo, o código de acesso à memória seria mantido no núcleo, enquanto o software que gerencia e mantém políticas de uso de memória (como a memória virtual) estaria de fora.

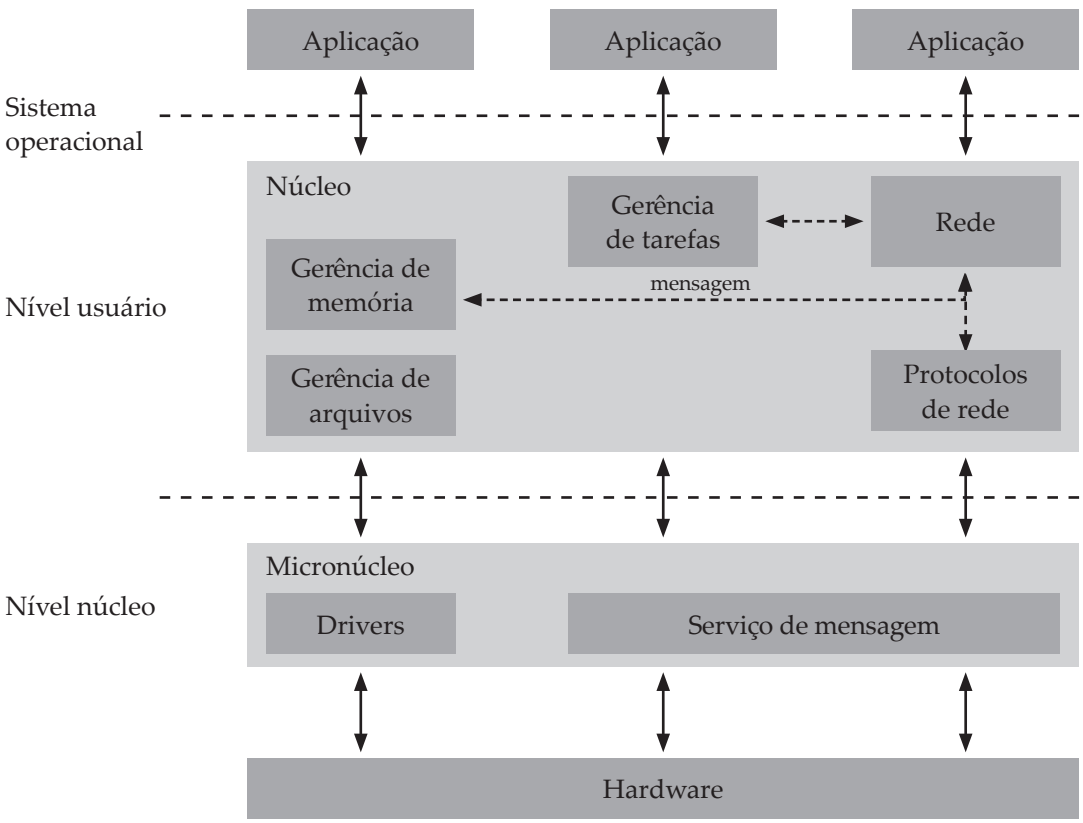
O micronúcleo também implementa um mecanismo de mensagens, pelos quais os processos comunicam-se entre si. Essa comunicação indireta garante que mecanismos de proteção de hardware sejam implementados e, ao mesmo tempo, torna o acoplamento entre os componentes menor. Para que dois processos troquem informações, um deverá solicitar serviços do outro. O processo requisitante torna-se "cliente" do processo que provê o serviço, considerado o "servidor", por isso, essa abordagem às vezes é conhecida como cliente/servidor.

A principal vantagem dos sistemas micronúcleo é a robustez. Como o núcleo é bastante essencial e enxuto, raramente trava ou entra em colapso. Além disso, caso um dos componentes do sistema apresente mau funcionamento, raramente comprometerá os demais, já que não os acessa diretamente – diferentemente do que ocorre em um sistema monolítico.

Outra vantagem é a flexibilidade. Como os componentes só conhecem as mensagens que trocam entre si, e não o seu funcionamento interno, é possível fazer atualizações das partes com facilidade, ou adicionar novas partes.

Do ponto de vista teórico, essa abordagem é considerada superior ao sistema monolítico (TANEMBAUM, 2016). E, na prática, ela foi adotada por praticamente todos os sistemas operacionais modernos, inclusive o Windows 10, MacOS e Unix.

**Figura 7 – Arquitetura micronúcleo.**




Fonte: Elaborada pelo autor.

### 6.3.3 Sistemas em camadas

Uma abordagem diferente seria estruturar o sistema em camadas cada vez mais abstratas. Assim, na camada inferior, estaria o acesso ao hardware. Nos sistemas que a implementaram, essa camada ficou conhecida como *Hardware Abstraction Layer* (HAL).

Acima dela, estaria uma camada de administração do hardware, que poderia implementar serviços como o conceito de pastas no sistema de arquivos, a memória virtual etc.

O sistema incluiria camadas cada vez mais altas, implementando outras funções, como políticas de segurança e até a interface gráfica do sistema.

Entretanto o empilhamento de camadas faz com que uma requisição de software tenha de ser processada camada por camada. E isso compromete a velocidade geral do sistema. Além disso, nem sempre é claro em que camada um componente deveria estar, ou mesmo, quais camadas deveriam existir. Isso é especialmente problemático, uma vez que recursos são interdependentes e, muitas vezes, um componente de uma camada inferior poderia necessitar de um serviço ou parte de um serviço de algo que está em uma camada superior.

Esses inconvenientes fizeram que essa abordagem, como arquitetura principal, fosse abandonada. Entretanto é comum que a estratégia seja adotada parcialmente em componentes de sistemas micronúcleo ou monolíticos. Por exemplo, no sistema de arquivos, é comum que exista uma camada para acesso ao hardware, abstraindo os dispositivos e outra que implemente o conceito de arquivos e divisão de blocos e uma superior, dividindo os diretórios e criando a estrutura hierárquica.

Essa abordagem foi utilizada no Windows NT, IBM OS/2 e Multics.

## Atividades

**1.** Explique o que os seguintes termos significam:

- Memória virtual.
- Time sharing.

**2.** Diferencie política e mecanismos e dê um exemplo de cada.

**3.** Cite as diferenças entre as arquiteturas micronúcleo e monolítica.

## Referências

BROOKS, F. **The mythical man-month**. Boston: Addison-Wesley, 1995.

DIBONA, C. **Open sources**: voices from the open source revolution. Sebastopol: O'Reilly, 1999.

MAZIERO, C. A. **Sistemas operacionais**: conceitos e mecanismos. Curitiba: UFPR, 2017. Disponível em: <http://wiki.inf.ufpr.br/maziero/lib/exe/fetch.php?media=so:so-livro.pdf>. Acesso em: 23 nov. 2017.

TANEMBAUM, A. S. **Sistemas operacionais modernos**. 4. ed. São Paulo: Pearson, 2016.

## ☑ Resolução

**1.**

- Memória virtual: programas não conhecem os endereços reais de memória, e sim, endereços fornecidos pelo sistema. Isso permite que o sistema remaneje conteúdo na memória sem que as aplicações percebam.
- Time-sharing: consiste em dividir o tempo do processador entre diversas aplicações. Assim, um único processador pode dar a ilusão de que processa várias tarefas ao mesmo tempo.

**2.** Políticas: são decisões tomadas no sistema operacional, tal como a quantidade de memória que será reservada para o sistema ou quanto tempo um processo durará no processador até que seja substituído pelos mecanismos de *time-sharing*. Um mecanismo é o programa que realmente faz as políticas acontecerem, como o sistema de gerência de processos.

**3.** Na arquitetura monolítica, todos os componentes têm forte acoplamento entre si e comunicam-se diretamente, tornando o sistema muito acoplado. Isso permite muita performance, mas pouca flexibilidade e dificuldade de manutenção. A arquitetura de micronúcleo mantém no núcleo apenas as atividades essenciais. Os recursos do sistema comunicam-se entre si por mensagens, ficando bastante desacoplados. Isso permite flexibilidade e facilidade de manutenção.

# 7
# Linguagens de programação

Os computadores seriam pouco úteis sem os softwares que os compõem. Eles são criados por programadores, que utilizam de uma forma de texto estruturada para escrevê-los. A gramática desse texto é chamada de *linguagem de programação*. Neste capítulo, entenderemos o que são as linguagens, seus diferentes tipos e como elas funcionam em nosso computador.

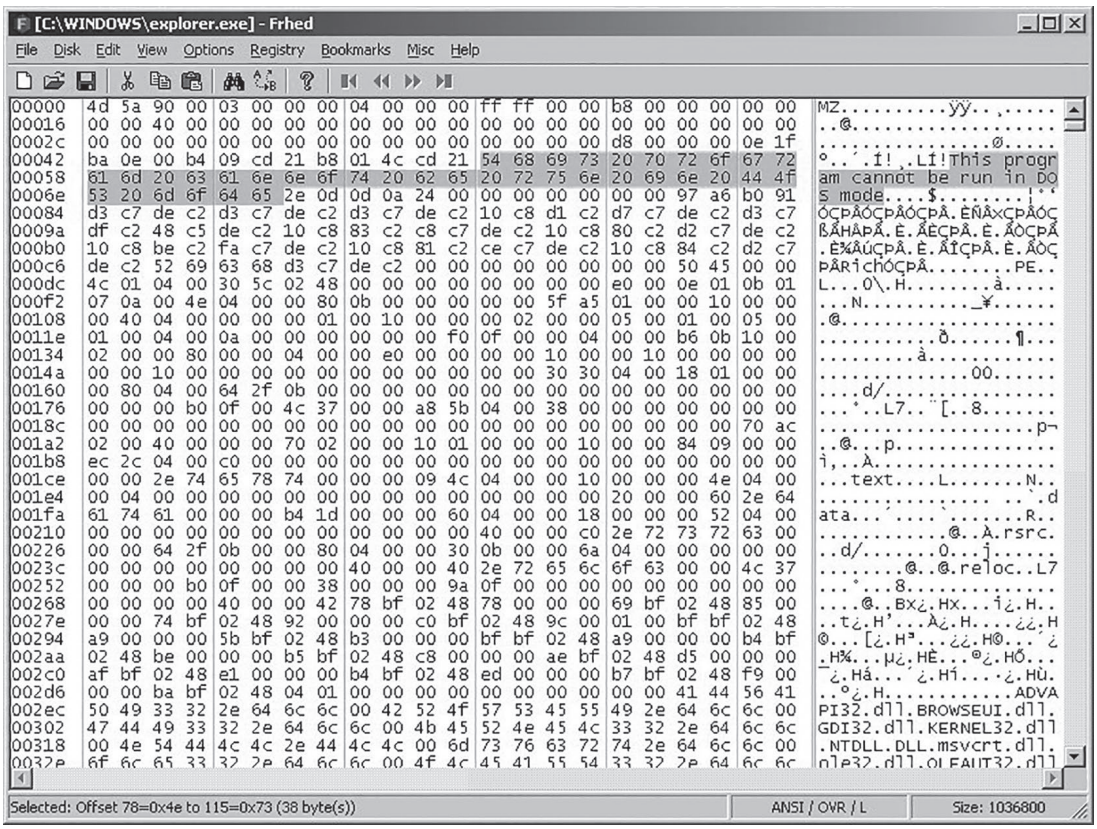
Vídeo

# 7.1 Linguagens

Conforme visto no Capítulo 4, a informação no computador é simplesmente um conjunto de números em binário. O computador executa instruções sequencialmente e existe um conjunto fixo delas, que são numéricas.

Essas instruções são chamadas de *linguagem de máquina* e variam de processador para processador ou de sistema operacional para sistema operacional. O código de máquina, entretanto, é pouco legível, extenso e muito difícil de entender. A Figura 1 a seguir mostra o trecho de um programa nessa linguagem.

**Figura 1 – Início do programa do Windows Explorer em linguagem de máquina.**




Fonte: Elaborada pelo autor.

O código de máquina é o conjunto de instruções hexadecimais na parte central. No lado direito, está a interpretação desse código, no caso de ser texto, no qual podemos ver destacada a mensagem de erro, caso o programa seja executado fora do Windows.

Imagine como seria complexo se seres humanos tivessem de lidar com esses números. Com o tempo, programadores criaram uma versão simplificada desse código. O que fizeram foi criar um texto que substituía a instrução do computador por um comando de três letras, seguidas dos parâmetros separados por vírgula.

Criaram então um programa que faz a tradução desse código, chamado de *montador* (*assembly*). E a linguagem estruturada, criada para representar a linguagem de máquina, foi conhecida como *assembler*.

A seguir, um exemplo de código, que imprime o texto *Hello World* (TUTORIALS-POINT, 2017):

**Figura 2 – Hello World em Assembly.**

```
section	.text
global _start ;must be declared for linker (ld)
_start:	;tells linker entry point
mov	edx,len ;message length
mov	ecx,msg ;message to write
mov	ebx,1 ;file descriptor (stdout)
mov	eax,4 ;system call number (sys_write)
int	0x80 ;call kernel
mov	eax,1 ;system call number (sys_exit)
int	0x80 ;call kernel
section	.data
msg db 'Hello, world!', 0xa ;string to be printed
len equ $ – msg ;length of the string
```

Fonte: Elaborada pelo autor.

Observe que os comandos ainda são muito específicos e pouco intuitivos. Há muitas operações de movimentações de dados (MOV), colocando valores em locais especiais da memória para serem interpretados de acordo com o que estiver no manual do processador.

Todos os textos ao lado direito, após os ponto e vírgulas, não fazem parte da linguagem de programação, são simplesmente comentários que serão descartados pelo montador, mas úteis para que um programador entenda o que está ocorrendo. Por serem tão próximos do hardware, chamamos esse tipo de *linguagem de baixo nível*.

Programas em assembly, embora muito rápidos e enxutos, mesmo que melhores do que escritos em linguagem de máquina, ainda eram difíceis de escrever, sujeito a erros e muito caros para se manter. Um erro poderia levar dias para ser descoberto. Caso o programador fosse desligado da empresa, seu substituto levaria vários meses para conseguir entender o código e atualizá-lo. Além disso, o programa é feito para um ambiente específico. Caso o sistema operacional ou o processador mudem, as instruções também alterariam e manutenções seriam necessárias.

Por volta dos anos 1940, programadores começaram o desenvolvimento de formas mais sofisticadas de escrever software. A primeira linguagem de programação popular para microcomputadores surgiu em 1954 e se chamava *Formula Translation (*Fortran) (IBM, 2017). Na Figura 3 há um exemplo de programa que escreve o mesmo texto (Hello World) escrito nessa linguagem:

**Figura 3 – Hello World em Fortran.**

```
program helloworld
print *, "Hello world!"
end program helloworld
```

Fonte: Elaborada pelo autor.

O Fortran já possuía comandos mais legíveis, estruturas que deixavam clara a repetição, notação matemática para fórmulas e conceitos importantes, como o de guardar valores em variáveis que fossem inteligíveis para seres humanos. Trata-se de uma **linguagem de alto nível**. Como seu código é muito distante da linguagem de máquina, um programa teve de ser escrito para fazer a tradução. Esse programa é chamado de *compilador*.

Além de fornecer uma sintaxe mais legível, era possível utilizar o mesmo programa em sistemas operacionais ou processadores diferentes, bastava que se escrevesse um compilador adequado. Melhor do que isso, compiladores poderiam ser programados para utilizar recursos avançados, como processadores matemáticos, e detectar alguns tipos de erro no código. O código final talvez não ficasse tão otimizado quanto o seu equivalente em assembly, mas os ganhos de flexibilidade e manutenção eram significativos.

Com base nisso, diversas outras linguagens de programação surgiram e, na atualidade, existem centenas de linguagens disponíveis.

## 7.1.1 Programas de computador

Em uma linguagem de computador moderna, um programa nada mais é do que um **algoritmo**, ou seja, uma sequência finita de instruções que definem como um problema pode ser resolvido.

Algoritmos não existem apenas no mundo da computação. Na cozinha, as receitas são algoritmos que descrevem como as comidas podem ser preparadas. Por exemplo:

Pão de queijo low carb

Ingredientes:

- 1 ovo
- 200 g de muçarela ralada
- 200 g de parmesão ralado
- Sal a gosto

Modo de fazer:

Misture os ingredientes. Bata-os até adquirirem consistência homogênea. Adicione sal a gosto. Coloque-os em uma forma de cupcake e leve-os ao forno preaquecido em temperatura média por 15 minutos.

Observe que essa receita é seguida de cima para baixo, instrução a instrução. Computadores fazem o mesmo.

Entretanto, a receita difere das linguagens de programação em alguns pontos importantes:

- Ambiguidade – em receitas, utilizamos muitos termos ambíguos, como temperatura média, sal a gosto e consistência homogênea. A interpretação disso varia de acordo com o cozinheiro que está seguindo a receita. Linguagens de programação têm pouquíssima ou nenhuma ambiguidade desse tipo.
- Estrutura – linguagens de programação têm uma gramática mais simples e estruturada, com um conjunto fixo de comandos e formas claras de representação.

Tudo isso ocorre porque, diferentemente de um ser humano, o computador executará os comandos cegamente, executando até mesmo erros óbvios até mesmo travando o programa, caso você o comande a executar o mesmo grupo de instruções para sempre.

Vídeo

## 7.2 Do texto ao software

Existem duas formas de se transformar texto em software. Uma delas é o processo de compilação, mencionado na seção anterior. A outra, é o processo de interpretação. Nesta seção, entenderemos melhor como esse processo funciona e a diferença entre eles, além de conhecer o funcionamento de linguagens que usam uma abordagem híbrida.

### 7.2.1 Compilação

```
print "Meu nome é " + nome
```

Para que o computador transforme esse comando em código assembly, o compilador realiza três tipos de análise sobre o texto (AHO; SETHI; ULLMAN, 1995):

1. Análise léxica – primeiramente, é necessário separar cada elemento que compõe esse comando. No caso citado, os elementos são o comando **print**, os delimitadores as **aspas** que indicam onde é um texto, operadores o sinal de **+** que combina dois textos, e uma palavra **nome**, indicando uma variável.
2. Análise sintática – em seguida, o compilador precisa identificar se esses comandos existem na gramática da linguagem. Seria **print** um código válido? E como ele deve ser usado, é realmente na forma print <texto>?
3. Análise semântica – por fim, o compilador precisa localizar o significado de símbolos criados pelo programador. Ao criar esses símbolos, o compilador associará essas variáveis com valores de memória, testará se o programador está utilizando a variável corretamente (por exemplo, não tentando colocar números em uma variável que só guarda texto) e se as operações sobre aquela variável fazem sentido (por exemplo, a expressão x/3 faz sentido se x for um número, mas não faz se x for um texto).

Após isso, o compilador será capaz de transformar o texto do programa em código binário. Para isso, ele irá mapear os comandos em estruturas predefinidas. Além disso, vários outros passos opcionais podem ser feitos, como otimizações de código, para

torná-lo menor ou mais rápido, para o uso de instruções específicas de um determinado processador, identificação de código inútil ou inatingível etc.

Na prática, porém, um programa de computador não é formado por um único arquivo, mas por dezenas deles. Como o processo de compilação não é rápido, seria inviável que todos os arquivos fossem sempre compilados, já que raramente o programador mexe neles de uma só vez (AHO; SETHI; ULLMAN, 1995).

A solução encontrada para isso foi compilar cada arquivo separadamente e utilizar um outro programa para unir o resultado depois, chamado de *linker* (ligador). Este irá gerar o executável final.

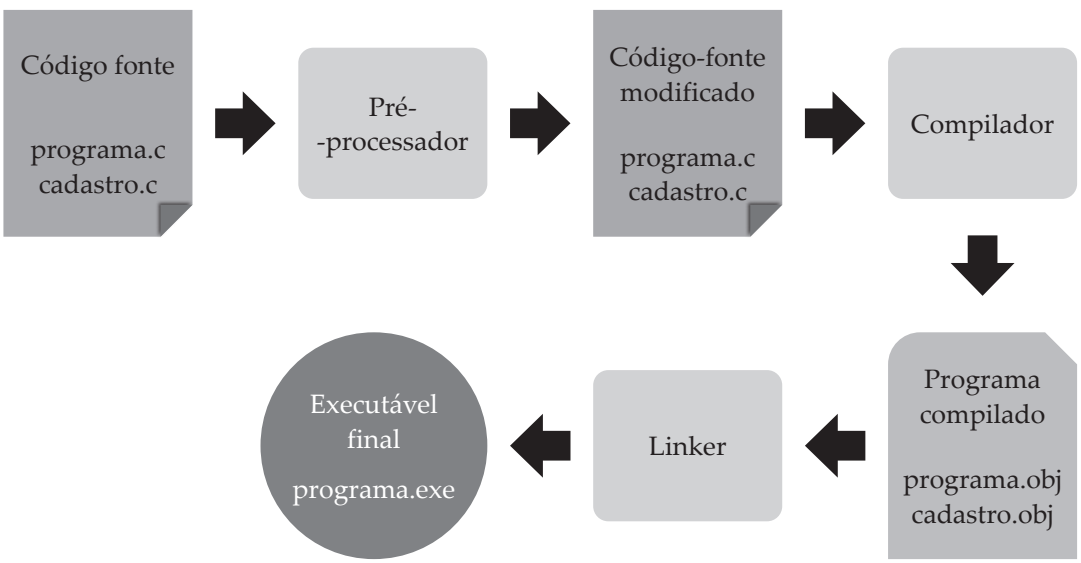
Algumas linguagens também possuem um passo anterior à compilação, conhecido como *pré-processamento* (STROUSTRUP, 2014). Esse tipo de linguagem permite que o programador defina uma série de comandos para substituição de texto (macros), que serão executados antes de a compilação iniciar. Por exemplo, em C, um dos comandos existentes na linguagem é o comando **while**, que faz repetições. Se o programador, porém, escrever uma diretriz como:

```
#define enquanto while
```

Poderá utilizar a palavra *enquanto* em seu código, pois ela será substituída pela palavra *while*, e o programa compilará. Comandos de pré-processamento também podem unir arquivos ou especificar trechos de código a serem incluídos, caso o compilador esteja rodando em uma determinada plataforma.

São exemplos de linguagens compiladas a linguagem C, C++ e o Delphi.

**Figura 4 – Compilação na linguagem C.**




Fonte: Elaborada pelo autor.

## 7.2.2 Interpretador

Uma abordagem alternativa à compilação é o processo de interpretação. Se fizermos analogia com uma língua estrangeira, o compilador agiria como um tradutor e o interpretador seria um intérprete, por exemplo.

Em linguagens interpretadas, cada comando é traduzido na hora e para seu respectivo código de máquina (AHO; SETHI; ULLMAN, 1995). A vantagem desse tipo de linguagem é que você pode acompanhar passo a passo o que o programa está fazendo, inspecionar valores de variáveis durante o processo e verificar se seu código está certo.

Outra vantagem é que não será necessário passar por um lento e demorado processo de compilação, o que é especialmente relevante para o caso de você querer que um programa rode em ambientes muito diferentes, basta que cada um tenha seu interpretador adequado.

Porém, há algumas desvantagens no processo: o programa rodará mais lentamente, já que a interpretação é uma tarefa dispendiosa e terá de ser feita todas as vezes. Além disso, um erro no programa poderá manifestar-se somente durante sua execução. Finalmente, o interpretador precisa do código-fonte para executar, que deverá ser distribuído – o que permite a um concorrente estudá-lo, copiá-lo ou modificá-lo.

Um exemplo de linguagem interpretada é a JavaScript, que é interpretada e executada pelos navegadores web.

## 7.2.3 Linguagens híbridas e máquinas virtuais

Com a necessidade cada vez mais crescente de um mesmo código rodar em várias plataformas e rodar de maneira rápida, surgiram linguagens as híbridas que tentam combinar esses aspectos.

Essas linguagens instalam na plataforma na qual irão rodar uma **máquina virtual**. Esta é uma espécie de interpretador, porém de um código muito mais próximo de uma linguagem de baixo nível, conhecida como *bytecode* (TUTORIALS-POINT, 2017).

Os programadores não escrevem esse código diretamente. Eles utilizam uma linguagem de alto nível e então a submetem a um compilador, que gera o bytecode.

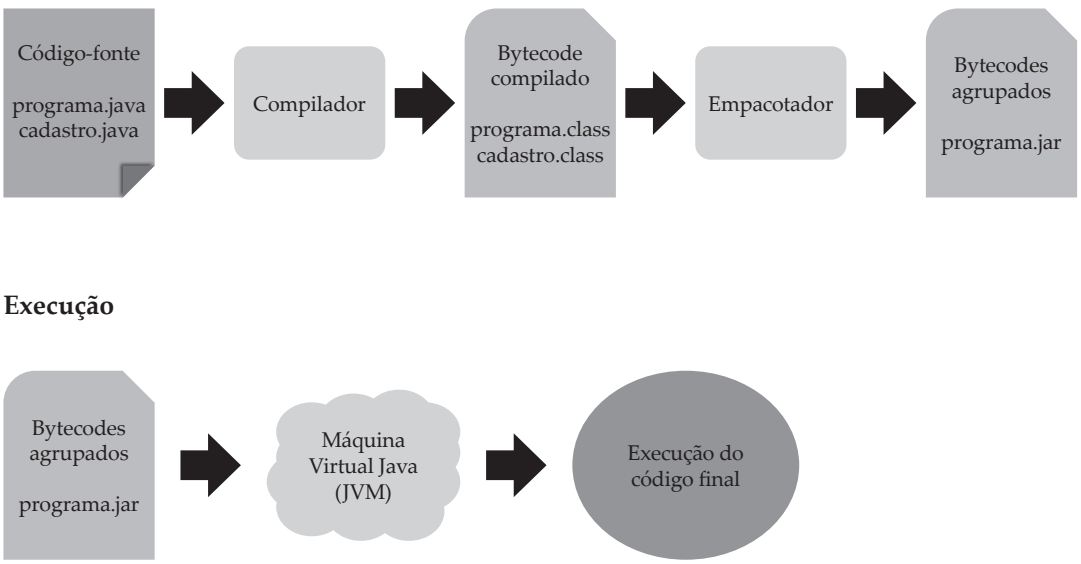
Assim, nessa abordagem, a parte mais demorada da compilação também ocorre previamente. Entretanto a execução final desse código já otimizado ainda é feita de maneira interpretada, pela máquina virtual instalada. O mesmo bytecode pode rodar em várias plataformas, desde que a máquina virtual correspondente esteja disponível (TUTORIALS-POINT, 2017).

A linguagem que popularizou esse conceito foi o Java. Outras linguagens também o utilizam, como o C# ou VB.Net. Algumas linguagens permitem agrupar os bytecodes em um único arquivo por um programa conhecido como *empacotador*. Não se trata de um linker, já que a tarefa de união é geralmente feita pela própria máquina virtual.

Máquinas virtuais modernas possuem habilidades bastante poderosas, como a capacidade de analisar trechos do bytecode mais utilizados e compila-los de fato (compilação just in time). Além disso, por conhecerem a plataforma na qual o código está realmente sendo

executado, as máquinas virtuais sempre podem habilitar instruções específicas de modo a aproveitar ao máximo seus recursos. Elas também podem garantir um ambiente de execução seguro, proibindo que programas maliciosos executem instruções que prejudiquem o computador onde executam (ORACLE, 1999).

**Figura 5 – Compilação e execução em Java.**




Fonte: Elaborada pelo autor.

A desvantagem dessa abordagem é que o usuário do programa precisa instalar a máquina virtual. Esse pode ser um passo misterioso, já que ele não entende exatamente o que ela é.

Outra desvantagem é que, geralmente, bytecodes são facilmente reversíveis em seu código-fonte original. Finalmente, embora sejam consideravelmente mais rápidas do que as linguagens puramente interpretadas, elas ainda são mais lentas do que as totalmente compiladas, o que pode ser um problema para aplicações que exijam muito processamento, como jogos.

Vídeo

## 7.3 Paradigmas

Embora existam centenas de linguagens de programação, boa parte delas têm estrutura similar, isso porque obedecem ao mesmo **paradigma de programação**.

Os paradigmas representam os conceitos utilizados pelos criadores da linguagem para a solução do problema. A mudança de um paradigma representará uma forma totalmente diferente de pensar no mesmo problema (FARIAS, 2013).

Será relativamente fácil para um programador alternar entre duas linguagens que usem o mesmo paradigma: geralmente, a tarefa se resume em aprender pequenas diferenças sintáticas e um ou outro recurso interessante que a linguagem implemente. Porém uma mudança

de paradigma é geralmente traumática. Envolve descobrir novas maneiras de abordar o problema e estruturá-lo.

## 7.3.1 Programação imperativa

Na programação imperativa, também conhecida como *programação procedural*, o programador diz ao computador uma série de comandos. Ou seja, ele escreve diretamente algoritmos, indicando ao computador o que fazer, passo a passo.

Nessa abordagem, o programador trabalha com dados e instruirá ao computador exatamente como eles devem ser trabalhados.

Vamos utilizar como exemplo o cálculo de fatorial, que é dado pelo produto dele com seus inferiores até o 1. Por exemplo, o fatorial de 5 é 5*4*3*2*1 = 120.

```
4. número ← 5
5. fatorial ← 1
6. REPITA ATÉ QUE número = 1
7. fatorial = fatorial * número
8. número = número – 1
9. FIM REPITA
```

Vejamos o que esse programa faz passo a passo: as linhas 1 e 2 guardam os valores 5 e 1 nas variáveis **número** e **fatorial**, respectivamente.

A linha 3 instrui o computador a repetir os comandos da linha 4 e 5 (delimitados pela instrução **FIM REPITA** na linha 6) até que a variável **número** passe a ter valor 1.

Na linha 4, a variável **fatorial** receberá o valor dela mesma multiplicado por **número**. Como na primeira vez que o programa executa o valor dela é 1 e o da variável **número** é 5, a linha equivale a:

```
fatorial = 1 * 5
```

O valor 5 será, portanto, armazenado na variável **fatorial**.

Na linha seguinte, a variável **número** terá seu valor atual reduzido de 1, ou seja, passará a ser 4. O programa atinge o comando **FIM REPITA** e volta até a linha 3, na qual a condição de parada é novamente testada. Como **número** ainda é diferente de 1, a linha 4 será executada novamente.

Observe que agora o valor na variável **fatorial** é 5 (fruto da execução anterior) e da variável **número** é 4. Portanto, a linha 4 agora rodará o comando:

```
fatorial = 5 * 4
```

E a variável **fatorial** passará a guardar na memória o valor 20. Novamente a linha 5 executa, e a variável **número** passará a valer 3.

O programa prosseguirá a execução novamente na linha 3. E continuará repetindo, já que **número** ainda tem não tem valor 1.

E, novamente, a linha 4 rodará, agora fazendo:

**fatorial = 20 * 3**

E atribuindo 60 ao **fatorial**. Seguindo a mesma lógica, vemos que o programa ainda rodará mais uma vez. Fazendo 60 * 2, e o fatorial de 5 (120) será calculado.

Observe que nessa abordagem o programador foi obrigado a descrever para o computador passo a passo o que fazer para calcular um fatorial.

O paradigma imperativo ainda é um dos mais comuns e mesmo outros paradigmas, como o orientado a objetos, ainda fazem uso de instruções imperativas.

Provavelmente, a linguagem imperativa de maior relevância no mercado atual é a linguagem C, presente no mercado de firmware e eletrônicos no geral.

## 7.3.2 Programação lógica

Nesse paradigma, o programador descreve apenas associações lógicas e então pede para o computador avaliá-las.

Vamos dar um exemplo. O mesmo programa de fatorial pode ser feito assim:

Se N for 1, então o fatorial de N é 1
Senão, o fatorial de N será dado por N * fatorial(N-1)

Observe que essas afirmações podem ser dadas em qualquer ordem.

O programador então pergunta ao computador: qual é o fatorial de quando N = 5? O computador tenta resolver as afirmativas lógicas.

N será dado por **N * fatorial(N–1)**. Como N é 5, o computador tenta avaliar **5 * Fatorial(5-1)**, ou seja **5 * fatorial(4)**. Agora, o problema se repete. Qual é o fatorial de 4? Ele busca novamente a alternativa lógica que responde à pergunta. E a resposta é: **4 * fatorial(4-1)**. O processo se repete, com o computador descobrindo que:

**O fatorial de 3 é 3 * fatorial(2)**
**O fatorial de 2 é 2 * fatorial(1)**

Então, ele para, já que agora ele sabe que o fatorial de 1 é 1 (já que se N for 1, o fatorial é 1). Por isso, pode resolver toda a expressão, pois agora sabe que:

**2 * fatorial(1) = 2**
**3 * fatorial(2) = 3 *2 = 6**
**4 * fatorial(3) = 4 * 6 = 24**
**5 * fatorial(4) = 5 * 24 = 120**

A linguagem PROLOG é um exemplo de linguagem que implementa esse paradigma. Ela é muito utilizada em pesquisas com inteligência artificial.

## 7.3.3 Programação funcional

No paradigma funcional, o programador descreve uma série de funções matemáticas. A composição das funções descreve como o problema deve ser resolvido.

Por exemplo, para o cálculo de uma média, o programa seria:

```
DIVIDIR(SOMA(números), QUANTIDADE(números)
```

Em algum momento, há um pouco de programação imperativa (por exemplo, para explicar como se faz a soma de um conjunto de números). Embora isso ocorra, rapidamente o programador passa a ter um conjunto robusto de funções e dá ênfase à resolução de problemas, em vez de pensar nos comandos e dados.

As linguagens LISP e Haskel implementam esse paradigma. Várias linguagens modernas têm acrescentado suporte parcial a esse paradigma: é o caso do C++, C#, Swift e até do Java.

## 7.3.4 Programação orientada a objetos

Nesse paradigma, em vez de pensar nos comandos e funções, o programador deverá pensar nos objetos que compõem o problema. Cada objeto possuirá um conjunto de **atributos**, que descrevem suas características, e um conjunto de **métodos,** que descrevem suas ações. Se o programador estiver construindo um cadastro de usuários para um website, por exemplo, os atributos do objeto usuário seriam nome, e-mail e avatar. Enquanto suas ações poderiam autenticar-se no sistema, excluir o seu cadastro etc. (BOOCH, 2007).

O programador então modela vários tipos de relacionamento entre objetos, sejam eles hierárquicos (do tipo "é um", chamados de relacionamento de herança), em que um objeto é um tipo mais específico de outro, sejam relacionamentos de uso (BOOCH, 2007).

Por exemplo, em um sistema de empresa, um objeto do tipo **gerente** poderia ser um tipo de **funcionário** (já que um **gerente** é um **funcionário**), portanto terá os mesmos atributos e métodos de qualquer funcionário de uma empresa (código, nome, salário etc.). Um **funcionário**, porém, tem uma mesa na empresa.

O paradigma orientado a objetos é atualmente o principal do mercado. As linguagens Java, C++ e C# são exemplos de linguagens orientadas a objetos.

## 7.3.5 Programação declarativa

Por fim, a programação declarativa é aquela que indica ao computador valores de um conjunto de dados. Ela normalmente é um modelo de programação simples, muito usado como forma de especificar o que fazer.

Um exemplo de linguagem declarativa é a de marcação HTML. Ela simplesmente explica ao computador como o texto de uma página está organizado: onde estão os títulos, o que é

texto, o que é link e para onde ele leva etc. O HTML utiliza junto de si folhas de estilo (CSS), que indicam como o HTML deve ser formatado.

A seguir, um exemplo de código HTML.

```
<html>
<head>
<title>Introdução à computação</title>
</head>
<body>
 Exemplo de HTML!
</body>
```

Nesse exemplo, vemos uma seção de cabeçalho (head), indicando que o título da página (que aparecerá na barra do navegador) é *Introdução à computação*. No corpo da página (indicado por BODY), está escrito *Exemplo de HTML!*

Por ser muito restrito e geralmente não dar muita liberdade para expressar qualquer tipo de problema, é muito comum que se exclua o paradigma declarativo ao se falar em linguagens de programação de maneira geral. São exemplos de linguagens desse paradigma o HTML, CSS e o JSON.

## Ampliando seus conhecimentos

No texto a seguir, observe essa reflexão feita por Dan Crow, em seu artigo no *The Guardian*.

### O software é a linguagem do nosso mundo

*(CROW, 2017)*[1]

O software está se tornando uma camada crítica de nossas vidas. É a linguagem do nosso mundo. No futuro, não saber a linguagem dos computadores vai ser tão desafiador quanto ser analfabeto ou não conhecer números hoje em dia.

[...]

O pensamento computacional ensina [...] como lidar com problemas grandes subdividindo-os em uma sequência de problemas menores e mais gerenciáveis. Ele permite que você lide com problemas complexos de maneiras eficientes e que funcionam em larga escala. Ele envolve criar modelos do mundo real com um nível de abstração adequado e focar nos aspectos mais relevantes. Ele ajuda a obter soluções específicas a partir das gerais.

1 Tradução do autor.

A aplicação dessa abordagem se estica além de escrever software. Campos tão distintos quanto a engenharia mecânica, mecânica de fluídos, física, biologia, arqueologia e música estão aplicando a abordagem computacional. Nos negócios, estamos começando a entender que mercados geralmente seguem regras que podem ser identificadas utilizando análise computacional.

O pensamento computacional é uma habilidade que todos deveriam aprender. Mesmo se você nunca se tornar um engenheiro de software profissional, você vai se beneficiar de ser capaz de pensar desse jeito. Ele vai auxiliar você a entender e dominar todos os tipos de tecnologia e resolver problemas em praticamente qualquer disciplina.

[...]

## Atividades

**1.** Associe o termo à esquerda com a definição à direita.

| | |
|---|---|
| A. Interpretador | 1. Modela problemas indicando passo a passo como o computador deve processar os dados. |
| B. Compilador | 2. Representação textual da linguagem de máquina. |
| C. Máquina virtual | 3. Uma sequência finita de instruções que define como um problema pode ser resolvido. |
| D. Paradigma imperativo | 4. Traduz e executa os comandos de um programa imediatamente. |
| E. Paradigma orientado a objetos | 5. Interpreta bytecodes permitindo que eles rodem em diferentes plataformas. |
| F. Algoritmo | 6. Traduz um programa integralmente em linguagem de máquina. |
| G. Assembly | 7. Modela o problema na forma de objetos com atributos e métodos. |

**2.** Cite quais são os três tipos de análises básicas feitas pelo compilador.

**3.** Escreva um algoritmo que explique como o pneu de um carro pode ser trocado.

## Referências

AHO, A; SETHI, R; ULLMAN, J. D. **Compiladores**: princípios, técnicas e ferramentas. Tradução de Daniel A. Pinto. São Paulo: Ed. LTC. 1. edição, 1995.

BOOCH, G. **Objected oriented analysis and design**: with applications. Londres: Addison-Wesley, 2007.

CROW, D. Why every child should learn to code. **The Guardian**, 7 fev. 2014. Disponível em: <https://www.theguardian.com/technology/2014/feb/07/year-of-code-dan-crow-songkick>. Acesso em: 24 nov. 2017.

FARIAS, G.; MEDEIROS, E. S. **Introdução à computação**. Brasília, DF: Universidade Aberta do Brasil, 2013.

IBM. **Fortran**: the pioneering programming language. Overview. Disponível em: <http://www-03.ibm.com/ibm/history/ibm100/us/en/icons/fortran/>. Acesso em: 24 nov. 2017.

ORACLE. **Java SE Documentation**. Introduction. Disponível em: <https://docs.oracle.com/javase/7/docs/technotes/guides/security/spec/security-spec.doc1.html>. Acesso em: 24 nov. 2017.

ORACLE. **Java virtual machine specification**. Charpter 1. Disponível em: <https://docs.oracle.com/javase/specs/jvms/se7/html/jvms-1.html>. Acesso em: 24 nov. 2017.

STROUSTRUP, B. **Principles and Practices Using C++**. Londres: Addison-Wesley, 2014.

TUTORIALS-POINT. **Assembly Variables**. Disponível em: <https://www.tutorialspoint.com/assembly_programming/assembly_variables.htm>. Acesso em: 24 nov. 2017.

WING, J. M. **Research notebook**: computational thinking – what and why. Disponível em: <http://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why>. Acesso em: 24 nov. 2017.

## ☑ Resolução

**1.** A-4, B-6, C-5, D-1, E-7, F-3, G-2

**2.**

a. Léxica: Separa os elementos que compõem o texto do programa.

b. Sintática: Testa se esses elementos existem e se respeitam a gramática da linguagem.

c. Semântica: Identifica o valor associado a variáveis e seus tipos de dado, criando seu endereçamento de memória e garantindo que seu significado seja respeitado.

**3.** Essa seria uma solução possível, mas é normal que algoritmos sejam ligeiramente diferentes entre si:

```
Abra o porta-malas do carro
Retire o macaco, a chave de roda e o pneu step
Localize o pneu que precisa ser substituído
Para cada parafuso do pneu
    Engate a chave de roda
    Gire no sentido anti-horário até o parafuso soltar
    Repita até retirar todos os parafusos
Puxe o pneu
Coloque o step
```

```
Para cada parafuso do pneu
  Engate o parafuso
  Gire no sentido horário até o parafuso estar firme
  Repita até retirar todos os parafusos
Guarde os equipamentos no porta-malas
```

# 8
# O futuro da informática

Depois de aprender sobre a história da informática e o funcionamento de um computador, percebemos o quão vasto é o campo da tecnologia.

A tecnologia já tem promessas para o futuro e, com essa evolução, podem aparecer problemas de maior complexidade, que serão frequentes na informática.

O que esperar para o futuro? Como se preparar para as novas tecnologias? Para finalizar, neste capítulo, vamos abordar as tecnologias pouco exploradas e quais são as perspectivas do amanhã na tecnologia.

Vídeo

# 8.1 O ensino

Pode parecer estranho tratarmos sobre educação básica em um livro de informática, mas, depois desse tempo de descobertas tecnológicas, nada mais natural do que isso impactar globalmente em uma sociedade. A tecnologia tem uma enorme importância na evolução do ensino. Pode-se dizer até mesmo que a era digital transformou a forma de pensar das pessoas e isso trouxe mudanças significativas no formato do aprendizado (MERANUS, 2012).

Desde a criação do computador até o avanço da internet, todo esse progresso foi fundamental para que na atualidade se possa repensar os métodos atuais de educação. Com o crescimento das áreas no mercado e a necessidade de melhorar a capacitação dos profissionais, necessita-se com urgência de novos métodos de ensino e uma reciclagem de modelo nas instituições.

## 8.1.1 School of One

Criada em 2009 por Joel Rush e Chris Rush, a **School of One** é um programa que visa proporcionar uma aprendizagem combinada. A ideia é que o aluno tenha a orientação do professor, mas que aprenda de acordo com seu ritmo, pois o foco passa a ser o estudante, não a metodologia. Cada aluno pode aprender conforme suas habilidades e possibilitar, por exemplo, que um aluno esteja no 5° ano em Matemática, mas no 9° em Biologia.

Essa nova maneira de aprender traz inovações ao ensino, que ainda segue padrões muito antigos e rígidos. É humanamente impossível para um professor atender a todas as necessidades pessoais de aprendizado de cada aluno em uma sala com 30 ou mais estudantes. Para que isso seja possível, foi preciso unir as habilidades do professor com a tecnologia e criar um sistema único que atendesse a necessidade de todos de maneira eficaz.

A School of One parte do princípio de que todos são diferentes e que cada um tem sua forma e seu tempo para evoluir (NEW YORK..., 2015). Alguns podem demorar mais tempo que outros para entender um problema matemático ou talvez algumas pessoas simplesmente não consigam entender alguns conceitos de biologia, mas isso não deve impedir que ela avance no que tem facilidade e gosta e, com isso, podemos aproveitar o melhor de cada um, sem deixar que fiquem espaços em branco no aprendizado.

Existem várias formas de se obter conhecimento: podemos aprender por meio de grupos de estudo, jogos educativos, softwares de aprendizado, videoaulas, entre outras maneiras.

As novas salas são divididas em pequenos grupos, nos quais cada aluno consegue seguir seu próprio ritmo de aprendizado. Assim, o professor consegue auxiliar cada um de forma singular, com enfoque nas maiores dificuldades do aluno, deixando aqueles que têm facilidade avançarem. Isso evita a desmotivação por todos estarem aprendendo a mesma coisa ou por não estarem entendendo nada (COSTA, 2012).

As matérias de 6º a 7º anos são divididas em 77 áreas de habilidades e conhecimento (LEVINE, 2009). Os estudantes são avaliados conforme o domínio em cada área. Com base nisso, um perfil de aprendizagem é gerado para cada aluno por meio do desempenho anterior.

Assim, os alunos são orientados para métodos específicos, como instrução em grupos pequenos e grandes, tutorias em pares ou individual, estudos independentes etc. O objetivo é que cada aluno possa abstrair o máximo de conhecimento possível das 77 habilidades.

## 8.1.2 Ensino a distância

De todas as modalidades de estudo que se promete para o futuro, a Educação a Distância (EaD) é a mais conhecida e utilizada atualmente.

Não há nenhuma novidade quando falamos que a educação é um problema sério. Como o estudo não era viável a todos até há pouco tempo, muitas pessoas não têm o ensino básico e ficaram com suas vidas profissionais estagnadas, pois não podem largar do trabalho para frequentar uma escola em tempo regular.

A Educação a distância foi criada inicialmente para suprir a necessidade da demanda social por educação. Após o fim da Primeira Guerra Mundial, já começaram a ser aplicados os modelos dessa modalidade de ensino por meio de correspondências, mas foi só por volta de 1960, com a explosão de propagandas, que esse formato se concretizou de fato. Com a constante expansão da tecnologia, foi possível cada vez mais consolidar esse novo método nas instituições.

Supõe-se que o grande marco da história da EaD seja a fundação da Universidade Aberta (originalmente *Open University*), na Inglaterra, em 1962. Para manter o ensino a distância entre as metodologias confiáveis e viáveis de acesso à educação, a universidade mantém um padrão de qualidade e objetiva.

Na atualidade, temos todas as ferramentas para fazer uma EaD com qualidade. A tecnologia está a nosso favor para melhorar a educação de todo o mundo. Por meio dela, foram criados o que chamamos de *Ambientes Virtuais de Aprendizagem* (AVA), softwares que viabilizam a comunicação entre o docente e o discente. Um bom exemplo são os portais de ensino, como o BlackBoard, que são utilizados para compartilhar conteúdos, encaminhar e-mail, escrever no fórum de dúvidas para gerar discussões com os demais alunos, entre outras funcionalidades.

O professor tem papel fundamental no ensino a distância, cabe a ele utilizar todas essas ferramentas da melhor forma, para transformar a experiência de aprendizado do aluno. Criar conteúdo, como videoaulas, disponibilizar livros e artigos e gerar discussões em grupos de estudo on-line são algumas das possibilidades do professor para transmitir o conteúdo para os alunos.

É importante utilizar de todos os meios para buscar a maior compreensão do estudante, pois, como esse tipo de ensino é focado no aprendiz, é essencial que existam diferentes maneiras de se aprender um conteúdo.

Mais importante do que um sistema preparado e um professor qualificado é o interesse do próprio aluno. Um dos objetivos principais do ensino a distância é criar autodidatas, que consigam aprender buscando a própria informação, perguntando para colegas próximos,

recorrendo à internet para solucionar problemas cotidianos. O plano é usar o professor como **um dos** pilares de ensino, não como a única **base**.

Empresas investem cada vez mais nesse tipo de curso para seus funcionários. Adultos que não tinham o Ensino Médio completo podem ter seus diplomas e seguir para o ensino superior. Jovens que buscam se profissionalizar, mas precisam trabalhar para pagar os estudos, agora podem estudar de acordo com seus horários disponíveis, no tempo livre do trabalho. Até mães que engravidam durante o ensino podem dar continuidade durante a gestação e licença-maternidade, os benefícios são incontáveis, percebe-se que essa iniciativa teve um grande impacto em todos os tipos de ensino, se não fosse a EaD nenhuma inovação seria possível agora, esse com certeza foi o primeiro passo para uma nova era na educação mundial.

O ensino a distância também rompe fronteiras. Agora é possível que um estudante brasileiro tenha diploma em Harvard sem sair do Brasil ou, em uma escala mais local, um estudante do interior do Brasil se gradue em uma boa faculdade da capital, sem sair de casa.

## 8.1.3 Sala de aula invertida

Essa técnica de ensino utiliza tecnologias como videoaulas e apostilas on-line para que os alunos estudem em casa, enquanto a sala de aula fica reservada para exercícios.

O objetivo é fazer com que as aulas sejam mais produtivas e participativas, o professor passa a ser um tutor que conduz debates e responde às dúvidas. Cada vez mais caminhamos para uma sala de aula cooperativa, sem hierarquia, em que o aluno e o professor possam aprender um com o outro de forma livre e não seguir metodologias ultrapassadas.

Dessa forma, todos os alunos podem interagir e acrescentar algo; é possível que todos consigam aprender juntos e alcançar objetivos em equipe, cada um somando com as habilidades que tem mais afinidade (NETO, 2008, p. 22-23).

Esse novo conceito de dar aula já vinha sendo especulado desde 1990, mas foi oficializado pelos professores estadunidenses Jonathan Bergman, Aaron Sams e Karl Fisch, que começaram a gravar videoaulas para alunos atletas, que deviam se ausentar no período dos jogos. Eles observaram que, ao aplicar esse modelo, os alunos conseguiam acompanhar as discussões em sala mesmo sem ter comparecido às aulas presencialmente (ESPÍNDOLA, 2016).

A ideia da aula invertida não é descartar o ensino presencial, é unir o que nós desenvolvemos com as tecnologias que estão sendo descobertas. Em grandes universidades, como a British Columbia, nos Estados Unidos, a tática foi aplicada nas aulas de Física; em Harvard, professores de Matemática fizeram o teste em seus alunos e em ambas as situações houve um crescimento significativo de desempenho. Aumento de 20% na frequência, 40% na participação e ganhos na aprendizagem entre 49% a 74% (PAIVA, 2016).

Vídeo

# 8.2 Realidade virtual

Para todos aqueles que sempre acompanharam filmes de ficção científica, o conceito de realidade virtual não é tão distante. Mas saindo das telas de cinema, há muitas coisas que utilizam dessa tecnologia.

Tanto na área de entretenimento, com jogos e cinema, quanto em áreas sedentas por novas tendências, como a medicina, a realidade virtual (RV), estão em constante expansão e prometem inovar muito a experiência do usuário. Recentemente, muitas tentativas e ideias surgiram para esse campo e a tendência é só crescer.

Para falar das conquistas atuais e das expectativas que se aguarda para o futuro, é preciso primeiramente entender do que estamos falando. Podemos definir a realidade virtual como uma experiência de imersão do usuário no espaço virtual, que tem como objetivo simular sensações reais com elementos que existem apenas virtualmente.

Todo detalhe é importante para proporcionar ao usuário um ambiente que se aproxime da realidade, portanto, quanto mais sentidos forem estimulados pelos elementos virtuais, melhor será a experiência. Nem sempre é possível estimular sentidos como olfato ou paladar, mas como esse meio ainda está em progresso, podemos esperar grandes projetos para o futuro.

## 8.2.1 Óculos de realidade virtual

Foram criados para melhorar a experiência do utilizador. Conhecido também por *Head-mounted Display* (HDM), é um dispositivo de projeção com uma tela em frente aos olhos. Existem dois tipos de HDM: os monoculares, em que imagens são captadas apenas por um olho e os binoculares, que são capazes de captar com ambos os olhos, obtendo uma imagem estereoscópica (ADAMI, 2017).

O primeiro modelo surgiu com base na ideia de um óculos para realidade virtual, que foi introduzido em 1939. Chamado de *View-Master*, foi um simulador de visão estereoscópica que funcionava com uma espécie de roleta com pequenas fotos, seu objetivo era acrescentar entretenimento visual das pessoas, trazendo algo inovador para a época.

Foi a partir dos anos 1990, com a evolução dos videogames, que começaram a surgir diversos protótipos para desenvolvimento. Pode-se dizer que se não fosse o progresso na indústria de jogos, a realidade virtual não teria sido tão estimada como é na atualidade. Foi com a ambição dos desenvolvedores em imergir o jogador dentro do universo do seu jogo e fazer com que mais de um usuário possa interagir no mesmo ambiente que surgiu o superinteresse no desenvolvimento de dispositivos HDM.

Além disso, os óculos já foram utilizados também para treinamentos militares e simuladores de voos para pilotos. Atualmente, óculos estão bem populares, principalmente na indústria de jogos.

O Google também apresentou lançamentos nessa parte, seu modelo CardBoard partiu da ideia de tornar a realidade virtual acessível para todos que possuem um smartphone. A estrutura é de papelão e é possível encaixar o smartphone, de 4 a 6 polegadas, na parte que seria

o visor. Apesar de sua estrutura simples causar a impressão de um dispositivo mediano, engana-se quem subestima esse óculos. Alguns usuários que testaram afirmam que suas lentes geram uma ótima visão periférica e que a experiência de imersão é uma das melhores.

Outro modelo do Google que difere um pouco dos modelos atuais é o Google Glass. Ainda na versão beta, o dispositivo não ocupa todo o campo de visão do usuário, é preciso fixá-lo somente em um dos olhos, um pouco acima do campo de visão. Com a tecnologia de foco, o utilizador pode acessar rotas de mapas, músicas, previsão do tempo sem mudar seu campo de visão. Além disso, é possível também fazer chamadas de vídeo, tirar fotos e compartilhar na internet.

## 8.2.2 Realidade aumentada

Apesar do nome parecido, esse campo é uma subárea da realidade virtual. A realidade aumentada utiliza da RV para simular o ambiente virtual juntamente com a nossa realidade. Nessa situação, se um usuário estiver utilizando um óculos de realidade virtual, ele continua vendo o ambiente em que se encontra, mas com alguns elementos projetados pelo dispositivo.

Esse mecanismo utiliza imagens, sons e vídeos para fazer detecção do ambiente em volta. Isso é feito por meio de headsets ou smartphones e tablets, ou seja, qualquer dispositivo capaz de detectar e mapear uma área. Essa informação é processada e pode ser enviada para um dispositivo inteligente, que deve executar suas tarefas conforme forem programadas, para diferentes situações, por exemplo, movimentar-se quando colidir com algo, no caso de um robô.

Os avanços nesse meio foram beneficentes, principalmente para a medicina. Atualmente, é possível mapear – por meio de câmera infravermelha – veias subcutâneas que eram invisíveis ao olho nu. A imagem é processada por um computador e projetada sobre a pele. Também é possível a prestação de apoio em cirurgias complexas, isto é, o médico pode utilizar um óculos de realidade virtual para adicionar informações ao seu campo de visão e receber instruções em tempo real de algum especialista. É possível também conferir imagens de raio-X, tomografia ou ultrassom em tempo real.

Para áreas de resíduos sólidos, mineração, telecomunicações e grandes obras, os dispositivos drones fizeram um grande acréscimo nos últimos tempos. Empresas como a Dronemapp, localizada em Curitiba, utilizam da realidade aumentada juntamente com a inteligência geoespacial para mapear locais e obter dados como o ciclo de vida de aterros sanitários, volumes e estimativas de extração de minério, inspeção de torres e monitoramento de obras.

Os drones também são bastante utilizados na parte de entretenimento. A Drone Racing League, por exemplo, é uma competição de corrida de drones. Os pilotos são colocados em um cenário interativo e disputam utilizando óculos de realidade virtual e controles, simulando para os competidores uma corrida real.

Vídeo

## 8.3 Computação quântica

Se pararmos para pensar nos computadores atuais no meio tecnológico, podemos prever que não será preciso muito tempo até que se crie uma nova forma de trabalhar com os dados. Devido ao grande crescimento da tecnologia é certo que há uma grande limitação na forma como armazenamos e manipulamos nossa informação.

Atualmente, já há dificuldade na resolução de problemas mais complexos, tanto na área matemática quanto em outras áreas, como a medicina. A tecnologia passou a ser parte fundamental não somente no próprio meio, mas em todos os outros campos profissionais, uma vez que todos usam algum meio tecnológico para se comunicar, organizar tarefas e exercer suas profissões, sem contar os avanços da robótica em diversas áreas.

Para contornar os novos desafios causados pela evolução tecnológica, as grandes empresas responsáveis pelo desenvolvimento de novas tecnologias, como o Google e a NASA, estão direcionando a atenção e produção para os famosos computadores quânticos. Eles prometem solucionar problemas de forma mais rápida e eficaz, elevando a conexão do usuário para outro patamar. Assim, além de termos um tráfego de dados muito mais rápido e eficiente, também será possível descobrir novas soluções para os futuros obstáculos que ainda serão encontrados (AS INOVAÇÕES..., 2016).

Seguindo leis da física quântica, os computadores prometidos para o futuro são bem mais rápidos do que os modelos clássicos, que têm por base a mecânica tradicional. Conhecidos por computadores quânticos, atualmente são utilizados para resolver problemas de otimização. Ou seja, caso que demoraria muito tempo para ser resolvido em uma máquina comum, como o clássico *algoritmo do caixeiro viajante*[1], levaria apenas alguns segundos em um computador quântico.

Diferentemente das máquinas atuais, que têm seus bits representados por 0 ou 1, os bits quânticos podem conter os valores 0 e 1 simultaneamente. A capacidade de estar em mais de um estado ao mesmo tempo denomina-se *superposição,* essa incerteza é o que nos permite codificar muito mais informação em um computador menor do que o modelo clássico. Outro fator é que esse tipo de computador não trabalha com cálculos feitos de forma linear, isso faz com que seja possível calcular todas as possíveis soluções de um problema ao mesmo tempo (IBM, 2017).

Existem diferentes formas de implementar um computador quântico. Pode ser feito por meio de pequenas **partículas**, desde que obedeçam às leis da mecânica quântica. Também é possível por meio de **átomos**, que têm seu estado de superposição representado por **excitados** ou **não excitados**, também fótons, que podem estar em dois lugares ao mesmo tempo, ou com prótons e nêutrons e até mesmo elétrons e pósitrons, que podem estar **para cima** e **para baixo** e se movimentar na velocidade da luz.

1 Nesse problema, pretende-se calcular a melhor trajetória que um viajante deveria utilizar para visitar um conjunto de cidades em uma única vez (por exemplo, a cidade A, B, C, D e E) partindo de qualquer cidade inicial e terminando em qualquer cidade final. O problema se torna exponencialmente mais complexo a cada cidade adicional colocada.

Recentemente, engenheiros da NASA e Google anunciaram que o computador D-Wave, pode solucionar um problema de otimização 100 milhões de vezes mais rápido do que o computador clássico, ou seja, é possível fazer em 1 segundo o que demoraríamos 10 mil anos.

## 8.3.1 A história da computação quântica

A história da computação quântica começa quando o físico norte-americano Richard Phillips Feynman apresentou a primeira proposta de utilizar um fenômeno quântico para resolver problemas de física quântica no computador. Isso ocorreu em 1981, na primeira **Conferência de Computação Física** do MIT (AS INOVAÇÕES, 2016).

Após 4 anos, o físico israelense da Universidade de Oxford David Deutsch desenvolveu um computador quântico universal, capaz de simular outro computador quântico da mesma complexidade. Funciona do mesmo modo que uma Máquina de Turing[2], é um modelo abstrato de computador que utiliza somente das propriedades lógicas de seu funcionamento. Essa realização garantiu que o físico entrasse para a história da computação quântica, garantindo o lugar de pioneiro no campo dos computadores quânticos (SILVA, 2013).

Em 1994, com a descoberta do matemático estadunidense Peter Shor, todos voltaram suas atenções para os computadores quânticos. A grande descoberta foi de um algoritmo, na atualidade mais conhecido como *Algoritmo de Shor*, que permite um computador quântico fatorar grandes números inteiros rapidamente. O algoritmo resolve o problema da fatoração e também o de logaritmo discreto. Ele poderia quebrar, em teoria, diversos sistemas de criptografia (SILVA, 2013).

Com a rápida evolução da computação quântica, muitos erros foram surgindo. Em 1996, a primeira proposta de correção desses erros foi feita com a seguinte ideia: erros quânticos poderiam ser sobrescritos e corrigidos.

Essa proposta, porém, teve uma série de limitações. Alguns erros eram corrigidos, mas não os do sistema de correção. De lá para cá foram feitas algumas sugestões de melhorias, mas a pesquisa ainda segue ativa.

Foi também em 1996 que o cientista da computação Lov Grover descobriu o algoritmo de pesquisa em bases de dados quânticas. A descoberta teve um pouco menos de atenção que outros algoritmos para fatoração, logs discretos ou simulações físicas, mas era aplicável em uma quantidade muito maior de problemas.

Em 1999, foram construídos os primeiros computadores quânticos baseados em montagem térmica.

O primeiro modelo híbrido não comercial foi proposto pela empresa canadense D-Wave, em 2007. Com processador de 16 qubits, foi a primeira máquina capaz de resolver problemas de lógica, encontrar soluções para jogos de raciocínio, entre outras tarefas práticas.

2 Como vimos no Capítulo 1, Turin demonstrou que um conjunto de instruções e dados poderiam ser utilizados em seu computador para resolver qualquer cálculo um conceito chamado *Turin Completude*. A Máquina de Turin é qualquer equipamento que implemente esse conceito.

Em 2010, a D-Wave lançou seu primeiro modelo para comercialização, o D-Wave One, que possui um processador de 128 qubits e, desde então, vem aumentando seus qubits nas gerações seguintes. Em 2013, com o D-Wave Two, o processador aumentou para 512 qubits; em 2015, com o D-Wave 2X com 1 quiloqubit. Em 2017, foi lançado o D-Wave 2000Q, com 2 mil qubits e recursos avançados de controle (DWAVE QUANTUM COMPUTING COMPANY, 2017).

Ainda no mesmo ano de lançamento do D-Wave 2000Q, a IBM lançou o IBM Q, uma plataforma comercial que oferece uma série de Application Programming Interface (API) para desenvolvedores e cientistas.

O sistema Quantum Experience é hospedado na nuvem, para permitir que qualquer computador tradicional possa ter acesso aos recursos quânticos. Para interagir com processador quântico da IBM, basta se cadastrar no portal, a partir disso já é possível rodar algoritmos, ter acesso a tutoriais, criar simulações e construir aplicações e serviços com base nesse tipo de tecnologia.

Algumas áreas que podem se beneficiar com essa evolução:

- Fornecimento e logística – otimizar entregas e suprimentos de produtos.
- Serviços financeiros – busca novos modelos de negócio e redução de risco em investimentos.
- Inteligência artificial – possibilitar que sistemas inteligentes aprendam de forma mais rápida consultando imagens e vídeos.
- Segurança na nuvem – utilização das leis da física quântica para ampliar a segurança de dados.
- Remédios e descoberta de materiais – descoberta de medicamentos e novos materiais com base em novas reações e interações químicas.

A plataforma Quantum Experience já conta com 40 mil usuários e foram feitos em média 270 mil experimentos únicos registrados. Mas a empresa garante que essa é somente a primeira fase do projeto, o objetivo é ampliar para 50 qubits nos próximos anos (DWAVE QUANTUM COMPUTING COMPANY, 2017).

## Atividades

**1.** Descreva o que é um qubit e como ele difere de um bit tradicional

**2.** Pesquise dois equipamentos de realidade virtual modernos de diferentes fabricantes e compare suas diferenças.

**3.** Cite as diferenças e semelhanças entre o ensino a distância (EaD) e a sala de aula invertida.

## Referências

ADAMI, A. Realidade virtual. **InfoEscola**. Disponível em: <http://www.infoescola.com/tecnologia/realidade-virtual/>. Acesso em: 6 ago. 2017.

AS INOVAÇÕES que tornam a computação quântica uma realidade factível. **ComputerWorld**, 8 set. 2016. Disponível em: <http://computerworld.com.br/inovacoes-que-tornam-computacao-quantica-uma-realidade-factivel>. Acesso em: 18 dez. 2017.

COSTA, M. M. School of One leva ensino personalizado à rede pública. **Porvir**, 19 nov. 2012. Disponível em: <http://porvir.org/school-leva-ensino-personalizado-a-rede-publica>. Acesso em: 18 dez. 2017.

COUTINHO, D. O que é realidade virtual? Entenda melhor como funciona a tecnologia. **Techtudo**, 29 set. 2015. Disponível em: <http://www.techtudo.com.br/noticias/noticia/2015/09/o-que-e-realidade-virtual-entenda-melhor-como-funciona-a-tecnologia.html>. Acesso em: 18 dez. 2017.

DWAVE QUANTUM COMPUTING COMPANY. Disponível em: <https://www.dwavesys.com/>. Acesso em: 18 dez. 2017.

ESPÍNDOLA, R. Como funciona a sala de aula invertida? **EDools**, 12 ago. 2016. Disponível em: <http://www.edools.com/sala-de-aula-invertida>. Acesso em: 6 ago. 2017.

IBM. **What is Quantum Computing**. Disponível em: <https://www.research.ibm.com/ibm-q/learn/what-is-quantum-computing>. Acesso em: 18 dez. 2017.

LANDIM, W. Futuro da realidade virtual. **Tecmundo**, 29 dez. 2009. Disponível em: <https://www.tecmundo.com.br/3d/3281-o-futuro-da-realidade-virtual.htm>. Acesso em: 18 dez. 2017.

LARA, E. EaD: vantagens da educação a distância. **Portal Educação**, 27 fev. 2009. Disponível em: <https://www.portaleducacao.com.br/conteudo/artigos/pedagogia/ead-vantagens-da-educacao-a-distancia/7671>. Acesso em: 18 dez. 2017.

LAZARO, D. As inúmeras possibilidades da computação quântica e a dobradura temporal. **CIO**, 7 mar. 2017. Disponível em: <http://cio.com.br/tecnologia/2017/03/07/as-inumeras-possibilidades-da-computacao-quantica-e-a-dobradura-temporal>. Acesso em: 18 dez. 2017

LEVINE, A. E. The School of One: The School of Tomorrow. **Huffington Post**, 16 nov. 2009. Disponível em: <http://www.huffingtonpost.com/arthur-e-levine/the-school-of-one-the-sch_b_288695.html>. Acesso em: 6 ago. 2017.

MARTINS, E. É hora de descobrir os segredos da computação quântica. **Tecmundo**, 28 ago. 2009. Disponível em: <https://www.tecmundo.com.br/computacao-quantica/2666-e-hora-de-descobrir-os-segredos-da-computacao-quantica.htm>. Acesso em: 18 dez. 2017.

MERANUS, J. Education Entrepreneurs: Joel Rose & Chris Rush. **NewsSchools**, 11 set. 2012. Disponível em: <http://www.newschools.org/news/new-classrooms>. Acesso em: 18 dez. 2017.

NEW YORK CITY DEPARTMENT OF EDUCATION. **School of One in New York City**: an implementation Guide. Nova York, 2015. Disponível em: <https://www.slideshare.net/iZoneNYC/school-of-one-in-new-york-city-an-implementation-guide>. Acesso em: 18 dez. 2017.

PAIVA, T. Como funciona a sala de aula invertida? **Carta Educação**, 24 ago. 2016. Disponível em: <http://www.cartaeducacao.com.br/reportagens/como-funciona-a-sala-de-aula-invertida>. Acesso em: 18 dez. 2017.

PONTIN, J. A Giant Leap Forward Computing? Maybe Not. **The New York Times**, 8 abr. 2007. Disponível em: <http://www.nytimes.com/2007/04/08/business/yourmoney/08slip.html?pagewanted=1&ei=5088&en=571f33b3b7cd5684&ex=1333684800&partner=rssnyt&emc=rss>. Acesso em: 18 dez. 2017.

QUANTUM FOR QUANTS. Quantum Computing. Disponível em: <http://www.quantumforquants.org/quantum-computing>. Acesso em: 6 ago. 2017.

SILVA, F. G. G. P. O. **O impacto da computação quântica na criptografia moderna**. 142 f. Tese (Mestrado em Engenharia Informática) – Universidade do Minho, Braga, 2013. Disponível em: <https://repositorium.sdum.uminho.pt/handle/1822/27977>. Acesso em: 18 dez. 2017.

SIMÃO NETO, A. **Cenários e modalidades de EAD**. Curitiba: IESDE, 2008.

VALERIO NETTO, A., MACHADO, L. S., OLIVEIRA, M. C. F. Realidade virtual: definições, dispositivos e aplicações. **Revista Eletrônica de Iniciação Científica**, 2002. Disponível em: <http://www.di.ufpb.br/liliane/publicacoes/2002_reic.pdf>. Acesso em: 18 dez. 2017.

## ☑ Resolução

**1.** O qubit trata-se do Bit Quântico. Graças à propriedade de superposição ele pode ter os valores 0 e 1 simultaneamente, o que é impossível num bit tradicional

**2.** Resposta pessoal.

**3.** O ensino a distância é a modalidade de ensino que permite ao aluno aprender de casa, sendo assistido pelo professor de maneira remota, por meio de ferramentas on-line. Já a sala de aula invertida é um modelo presencial; o aluno tem contato com o professor. A semelhança é que ele também usa ferramentas de EaD, como vídeos e fóruns para estudar o conteúdo que seria lecionado tradicionalmente em uma sala de aula. A diferença é que o aluno ainda tem contato com professores e colegas em sala, porém com enfoque na resolução de atividades em grupo.


Código Logístico
57096

Fundação Biblioteca Nacional
ISBN 978-85-387-6365-9
9 788538 763659