Programação de Computadores

Aula 01

Introdução

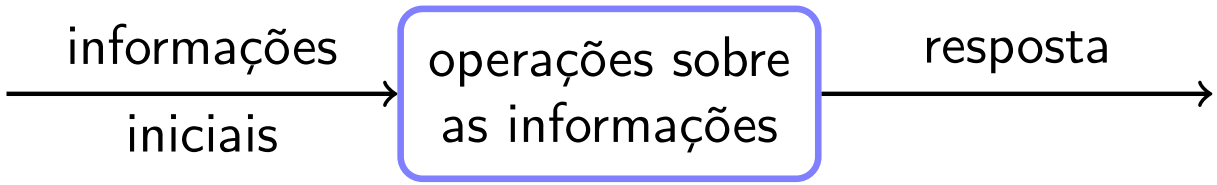
José Romildo Malaquias

Departamento de Computação
Universidade Federal de Ouro Preto

2011.1

Processamento de dados

- **Processamento de dados**: Série de operações que se aplica a um conjunto de dados (*entrada*) para obter outro conjunto de dados ou resultados (*saída*).



- Exemplos:
  - Procurar um número de telefone na lista telefônica e anotá-lo em uma caderneta. [manual]
  - Somar valores das compras no supermercado com o auxílio de uma calculadora. [semi-automático]
  - Utilizar o recurso de pesquisa de um arquivo, pelo seu nome, no computador. [automático]

Computador

- Um **computador** é uma máquina capaz de receber, armazenar e enviar dados, e de efetuar, sobre estes, seqüências previamente programadas de operações aritméticas (como cálculos) e lógicas (como comparações), com o objetivo de resolver problemas. [Dicionário Aurélio]
- Os computadores processam dados sob o controle de um conjunto de instruções chamados **programas de computador**.

Vantagens e desvantagens do uso do computador

- Vantagens:
  - Processa grande volume de dados com rapidez
  - Realiza cálculos com exatidão
  - Pode ser programado
- Desvantagens:
  - Não tem iniciativa
  - Não tem independência
  - Não é criativo e nem inteligente
  - Precisa receber instruções nos mínimos detalhes

Programa

- **Roteiro** que orienta o computador, mostrando-lhe a seqüência de operações necessárias para executar uma determinada tarefa
- Seqüência de instruções/comandos que dirigem a CPU na execução de alguma tarefa
- Exemplos: processador de texto, planilha, sistema operacional, navegador web, folha de pagamento

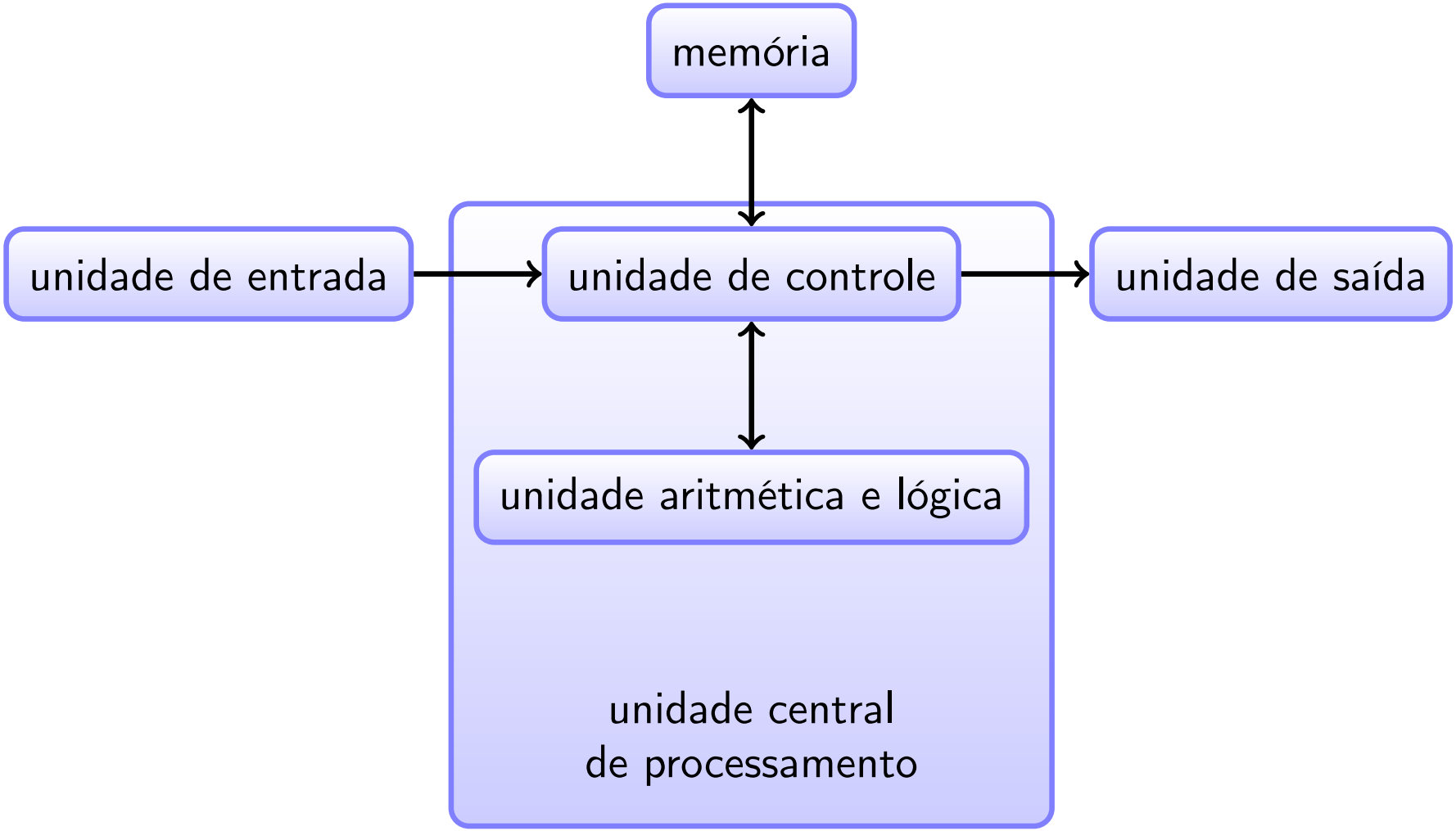
Hardware e *software*

- **Hardware**: conjunto de componentes mecânicos, elétricos e eletrônicos com os quais são construídos os computadores e equipamentos periféricos
- **Software**: conjunto de programas que permitem usufruir da capacidade de processamento fornecida pelo hardware

Arquitetura de *von Neumann*

- A arquitetura básica dos computadores atuais foi definida inicialmente por John von Neumann.
- O computador é constituído de:
  - unidades de entrada
  - unidades de saída
  - memória
  - unidade central de processamento, composta por
    - unidade de controle
    - unidade lógica e aritmética

Organização do Computador



Dispositivos de entrada e saída

- **Unidade de entrada**: traduz informações de um dispositivo de entrada, como letras, números, imagens, etc, para um código que a unidade central de processamento entende.
  Exemplos: teclado, *mouse*, *scanner*, *pen-drive*, câmera de vídeo.
- **Unidade de saída**: converte os dados da forma interna de codificação do computador para uma forma adequada à interpretação humana, ou que possa servir como entrada para outros dispositivos.
  Exemplos: monitor de vídeo, impressora, *plotter*, *pen-drive*.

Unidade central de processamento

- **Unidade Lógica e Aritmética (ULA)**: realiza operações aritméticas (cálculos) e lógicas (comparações, decisões).
- **Unidade de controle**: controla o fluxo de informações entre todas as unidades do computador e permite a execução das instruções na seqüência correta.

Memória

- **Memória**: armazena dados e intruções (programas)
- **bit** (binary digit - dígito binário) é a menor unidade de informação e pode assumir dois valores (0 ou 1)
- Unidades:

1 byte	8 bits
1 kilobyte (KB)	1024 bytes
1 megabyte (MB)	1024 kilobytes
1 gigabyte (GB)	1024 megabytes
1 terabyte (TB)	1024 gigabytes

Observe que nestas unidades os prefixos significam multiplicar por $2^{10} = 1024$, e não por $10^3 = 1000$.

Memória principal

- **Memória principal** ou memória primária:
  - a unidade de processamento pode endereçá-la diretamente
  - sem ela o computador não pode funcionar
  - contém a informação necessária para o processador num determinado momento
  - sem elas o computador não pode fucionar
  - Incluem
    - memória RAM (volátil: perde o conteúdo ao ser desligada)
    - memória ROM (não volátil: não perde o conteúdo ao ser desligada)
    - registradores
    - memória *cache*

Memória secundária

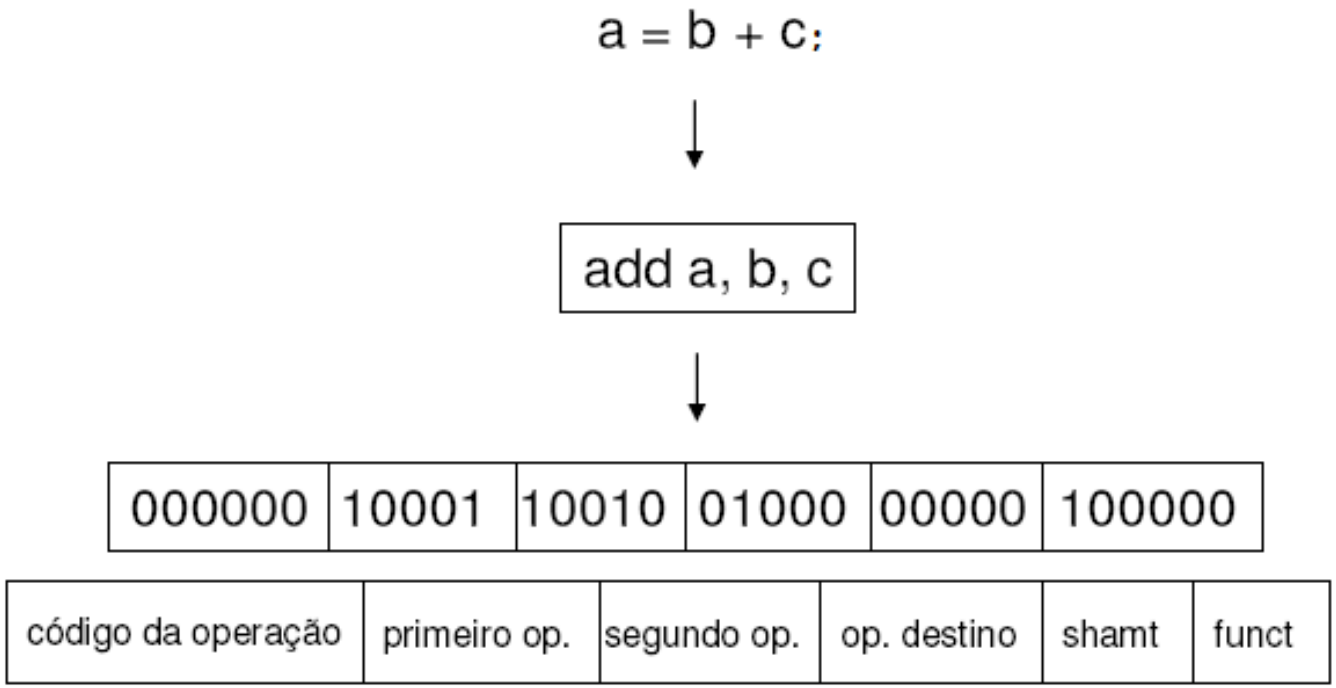
- **Memória secundária** ou memória auxiliar:
  - não podem ser endereçadas diretamente pelo processador
  - sua informação deve ser carregada na memória principal antes de poder ser tratada pelo processador
  - geralmente são não voláteis (permite guardar os dados permanentemente)
  - inccluem
    - discos rígidos
    - fitas magnéticas

Sistema Operacional

- **Sistemam operacional** é um conjunto de programas que supervisiona e controla as atividades do computador.
- Atua como intermediário entre os demais programas e o hardware do computador.
- Normalmente é executado automaticamente quando o computador é ligado.
- Funções:
  - gerência de memória
  - gerência de processador
  - gerência de arquivos
  - gerência de dispositivos de entrada e saída
- Exemplos: Windows 7, Linux, FreeBSD, MacOS, Solaris

Linguagem de máquina

- **Linguagem de máquina**: conjunto de instruções codificadas como sequências de zeros e uns que o processador entende.
- **Instrução**: Informa à Unidade de Controle qual operação a ser realizada e onde estão os dados na memória.
- Exemplo:



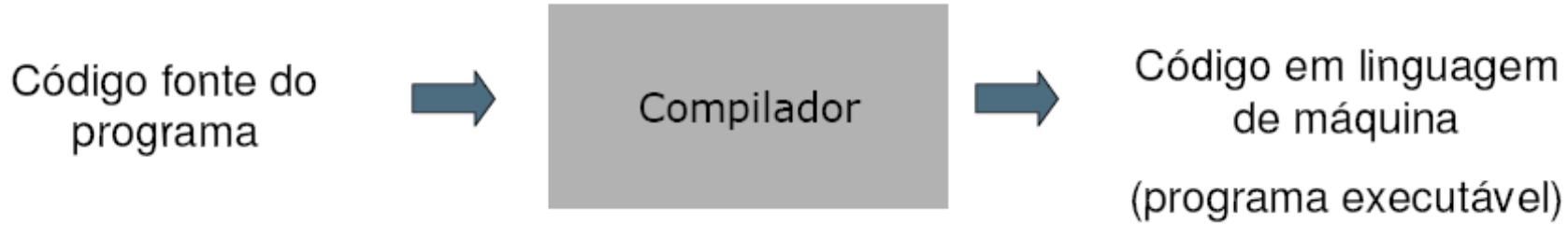
Compilação I

- Software permite realizar processamento de dados

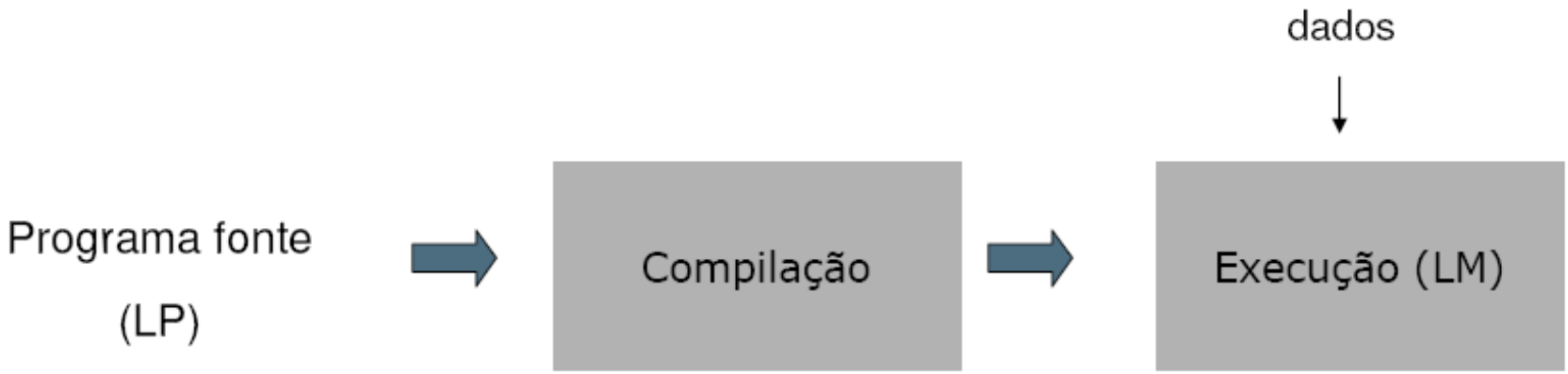
- Programador desenvolve software
- **Linguagem de Programação** (LP): linguagem que tanto o computador quanto o desenvolvedor entedem

Compilação II

- Todo programa escrito em uma LP tem que ser traduzido para linguagem de máquina para ser executado
- **Compilador**: programa tradutor

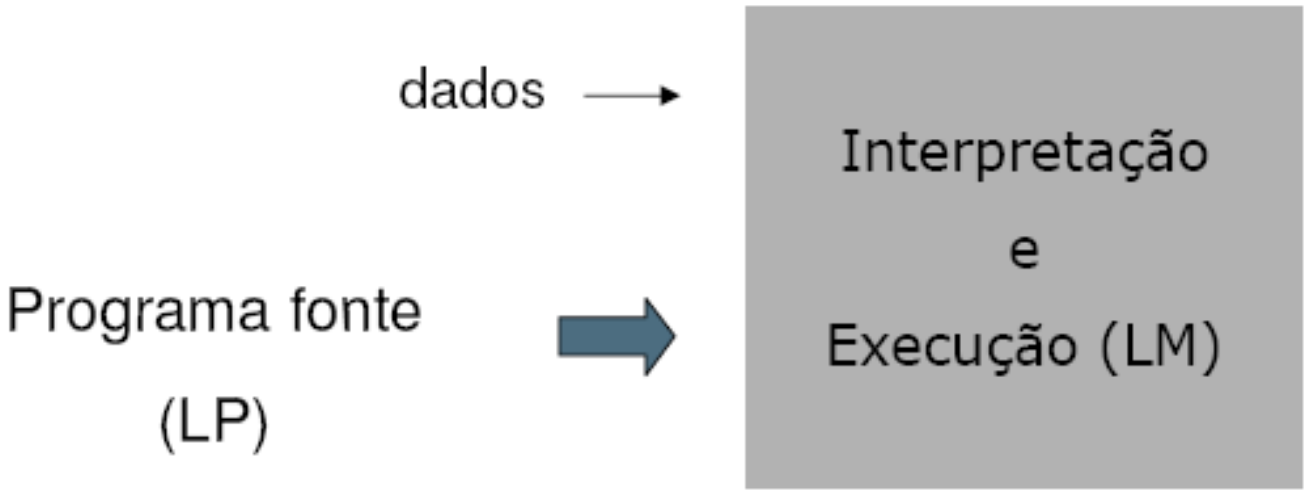


Compilação III



- O código executável é executado na máquina sem precisar do código-fonte
- Vantagem: execução é rápida pois já está na linguagem da máquina
- Desvantagem: não portabilidade para máquinas com arquiteturas diferentes

Interpretação



- Programa interpretador age como um simulador de um computador virtual que entende as instruções da LP
- Cada instrução do código-fonte analisada e executada de cada vez.
- Não há tradução para linguagem de máquina.

Etapas do Desenvolvimento de software

- **Análise**: Estuda-se o problema para definir os dados de entrada, o processamento e os dados de saída
- **Algoritmo**: Descreve-se o problema por meio de ferramentas (fluxograma, pseudocódico(português estruturado), etc)
- **Codificação**: Algoritmo é transformado em código da linguagem de programação escolhida

Exemplo

Somar três números

- Análise?
- Algoritmo?
- Codificação?

Representação de dados

- Computadores usam *chaves elétricas* para representar números e caracteres.
- Cada chave pode estar ligada (1) ou desligada (0).
- Uma combinação de estados representa um dado, por exemplo um número ou um caracter.
- Portanto os computadores usam um sistema binário.
- *Existem 10 tipos de pessoas, as que entendem números binários e as que não entendem.*

Sistemas de numeração

- **Numeral** é um símbolo ou grupo de símbolos que representa um número em um deteminado instante da evolução do homem.
- Os símbolos "11", "onze"e "XI"(onze em latim) são numerais diferentes, representativos do mesmo número, apenas escrito em idiomas e épocas diferentes.
- **Sistema de numeração** é um sistema em que um conjunto de números é representado por numerais de uma forma consistente.
- Um sistema de numeração deve:
  - representar uma grande quantidade de números úteis
  - dar a cada número uma única representação
  - refletir as estruturas algébricas e aritméticas dos números

Sistemas de numeração posicional

- Em um **sistema de numeração posicional** o valor de um algarismo depende da posição do algarismo dentro de uma ordem de valores.
- Exemplo: o número 2989 no sistema decimal é formado por
  - 2 unidades de milhar,
  - 9 centenas,
  - 8 dezenas e
  - 9 unidades

  ou seja,

$$2989 = 2000 + 900 + 80 + 9$$

  ou ainda,

$$2989 = 2 \times 10^3 + 9 \times 10^2 + 8 \times 10^1 + 9 \times 10^0$$

Sistemas de numeração não posicional

- Em um **sistema não posicional** não é só a posição do algarismo que importa para composição do número.
- Exemplo: sistema de numeração romano

$$(DCCLXIV)_{\text{romano}} = 500 + 100 + 100 + 50 + 10 - 1 + 5 = 764$$

Base de um sistemas de numeração

- A **base** de um sistema de numeração posicional é a quantidade de algorismos utilizados na representação dos números nesta base.

sistema de numeração	**base**	**algarismos**
binário	2	0, 1
	5	0, 1, 2, 3, 4
octal	8	0, 1, 2, 3, 4, 5, 6, 7
decimal	10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
hexadecimal	16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

- Um algarismo binário é também chamado de **bit** (**bi**nary digi**t**).
- Com n dígitos podemos representar B^n valores diferentes na base B.

Geração dos números naturais na base b

- Avançamos o dígito menos significativo (mais à esquerda) de 0 até $b-1$. Em seguida avançamos de $b-1$ para 0, e o digito imediatamente à sua esquerda avança seguindo a mesma regra.
- Exemplo no sistema decimal:

0	10	20		90	100	110		990	1000	1010	
1	11	21		91	101	111		991	1001	1011	
2	12	22		92	102	112		992	1002	1012	
3	13	23		93	103	113		993	1003	1013	
4	14	24	...	94	104	114	...	994	1004	1014	...
5	15	25		95	105	115		995	1005	1015	
6	16	26		96	106	116		996	1006	1016	
7	17	27		97	107	117		997	1007	1017	
8	18	28		98	108	118		998	1008	1018	
9	19	29		99	109	119		999	1009	1019	

Geração dos números naturais na base b

- Exemplo no sistema binário:

binário	decimal
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	10
1011	11
1100	12
⋮	⋮

Conversão para o sistema decimal

- Conversão de um número $a_n a_{n-1} \cdots a_0$ em uma base B para a base 10:

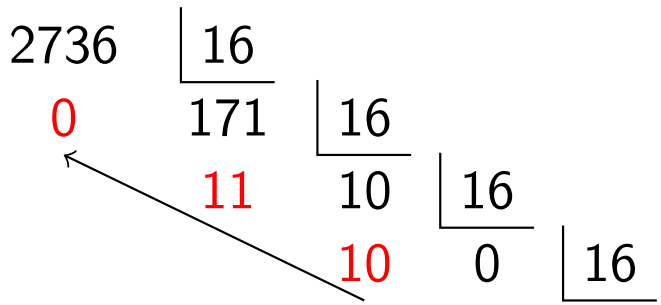
$$X = a_n B^n + a_{n-1} B^{n-1} + \cdots + a_0 B^0$$

onde a_i é um algorismo da base B, e n é um valor que representa a posição mais significativa do número.

- Exemplos:
$(3547)_8 = 3 \times 8^3 + 5 \times 8^2 + 4 \times 8^1 + 7 \times 8^0 = 1895$
$(10010111)_2 =$
$1 \times 2^7 + 0 \times 2^6 + 0 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 227$
$(60C4A)_{16} = 6 \times 16^4 + 0 \times 16^3 + 116 \times 16^2 + 4 \times 16^1 + 10 \times 16^0 = 24772$

Conversão a partir do sistema decimal I

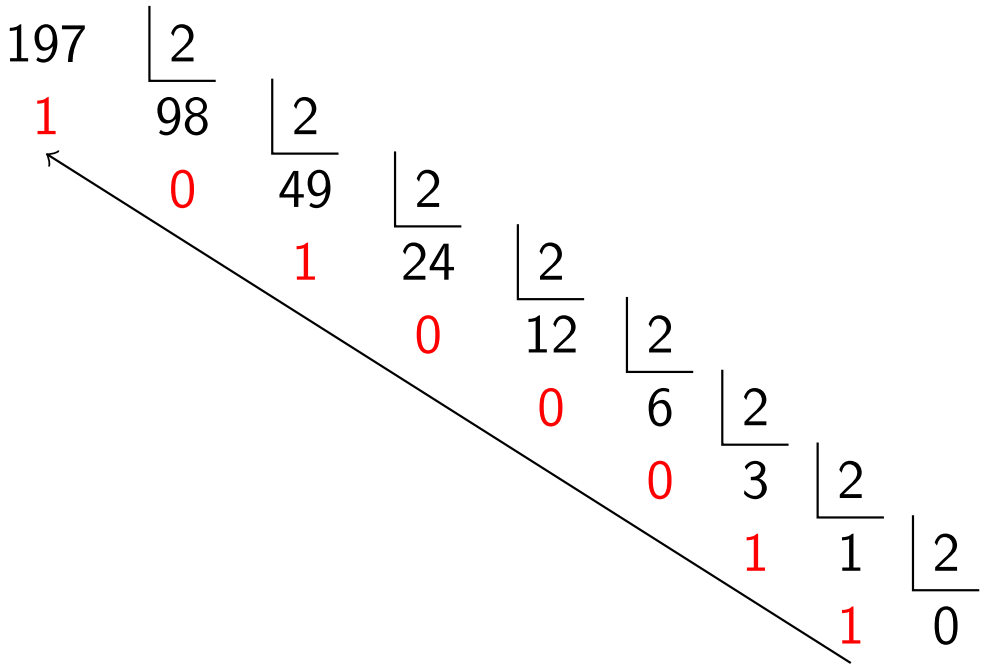
- Conversão de um número $a_n a_{n-1} \cdots a_0$ na base 10 para a base B: faz-se divisões sucessivas por B e toma-se os restos das divisões no sentido ascendente.
- Exemplo: $(2736)_{10} = (?)_{16}$



Mas $(10)_{10} = (A)_{16}$, e $(11)_{10} = (B)_{16}$
então $(2736)_{10} = (AB0)_{16}$
$(197)_{10} = (11000101)_2$

Conversão a partir do sistema decimal II

- Exemplo: $(197)_{10} = (?)_2$



$(197)_{10} = (11000101)_2$

Conversão entre hexadecimal e binário I

- Representação do conjunto dos símbolos do sistema hexadecimal mediante grupos de quatro bits:

hexadecimal	binário	decimal	hexadecimal	binário	decimal
0	0000	0	8	1000	8
1	0001	1	9	1001	9
2	0010	2	A	1010	10
3	0011	3	B	1011	11
4	0100	4	C	1100	12
5	0101	5	D	1101	13
6	0110	6	E	1110	14
7	0111	7	F	1111	15

Conversão entre hexadecimal e binário II

- Exemplo: $(A56B)_{16} = (?)_2$

A	5	6	B
1010	0101	0110	1011

$(A56B)_{16} = (1010\,0101\,0110\,1011)_2$

Conversão entre hexadecimal e binário III

- Exemplo: $(1000\,1111\,0011\,0101\,1001)_2 = (?)_{16}$

1000	1111	0011	0101	1101
8	F	3	5	D

$(10001111001101011001)_2 = (8F35D)_{16}$

Codificação binária I

- O **sistema de codificação** é utilizado em computação para que as informações possam ser armazenadas no computador.
- A codificação é a escrita de um processo ou conjunto de dados de uma forma simbólica, aceitável por uma unidade de processamento de dados.
- A **codificação binária** utiliza o sistema de numeração binário para representar as informações no computador.
- O menor item de informação é denominado **bit** (binary digit). Ele pode representar dois valores "0" ou "1".
- Uma sequência de bits é denominada **byte**. Normalmente um byte é formado por 8 bits.
- Uma sequência de bytes é denominada **palavra**.
- O tamanho exato do byte e da palavra depende da arquitetura do computador.

Codificação binária II

- Exemplo: uma palavra de quatro bytes de 8 bits pode representar diferentes dados: 11000010 01001111 01001101 00000000
  - número inteiro 3259976960
  - sequência de caracteres `BOM`
  - instrução em linguagem de máquina

Codificação de caracteres

- **Codificação ASCII** (American Standard Coded for Information Interchange)
  - utiliza 7 bits de um byte de 8 bits
  - número de valores possíveis: $2^7 = 128$
- **Codificação ASCII estendida**
  - utiliza 8 bits de um byte de 8 bits
  - número de valores possíveis: $2^8 = 256$
  - os códigos de 0 a 127 são idênticos à codificação ASCII
  - os códigos de 128 a 255 representam caracteres nacionais
- **Codificação EBCDIC** (Extended, Binary Coded With Decimal Interchanged Code)
  - utiliza 8 bits de um byte de 8 bits
  - número de valores possíveis: $2^8 = 256$

Exercícios

Fazer as conversões entre os sistemas de numeração indicados:

1. $(1010)_2 = (?)_{10}$
2. $(153)_{10} = (?)_2$
3. $(AF53)_{16} = (?)_{10}$
4. $(101100110100)_2 = (?)_{16}$
5. $(518610)_{10} = (?)_{16}$
6. $(100010)_2 = (?)_{16}$
7. $(568)_{10} = (?)_2$
8. $(AF53)_{16} = (?)_2$
9. $(75014)_{10} = (?)_8$
10. $(175010)_8 = (?)_{10}$

Perguntas?