“9,8,7,6,5,4,3,2,1. Com essas nove figuras Indianas, e o sinal 0, chamado sifr pelos Árabes, qualquer número pode ser escrito” (Leonardo de Pisa, conhecido como Fibonacci, em seu livro Liber Abaci, 1202).

Base 10 para outras bases

Paulo Ricardo Lisboa de Almeida

DSBD

UFPR
UNIVERSIDADE FEDERAL DO PARANÁ

Capacidade de representação

Na base 10, se temos apenas um dígito, quantas possibilidades de valores podemos armazenar?

9

Capacidade de representação

Na base 10, se temos apenas um dígito, quantas possibilidades de valores podemos armazenar?

10 possibilidades – Em uma casa, podemos armazenar qualquer valor $\in [0\text{-}9]$.

$\underline{9}$

Capacidade de representação

Na base 10, se temos apenas um dígito, quantas possibilidades de valores podemos armazenar?

10 possibilidades – Em uma casa, podemos armazenar qualquer valor $\in$ [0-9].

E se tivéssemos dois dígitos?

__ __

Capacidade de representação

Na base 10, se temos apenas um dígito, quantas possibilidades de valores podemos armazenar?

10 possibilidades – Em uma casa, podemos armazenar qualquer valor $\in$ [0-9].

E se tivéssemos dois dígitos?

10 possibilidades – Cada casa tem 10 possibilidades: 10×10 = 100.

$\underline{9}\,\underline{9}$

Capacidade de representação

Na base 10, se temos apenas um dígito, quantas possibilidades de valores podemos armazenar?

10 possibilidades – Em uma casa, podemos armazenar qualquer valor $\in$ [0-9].

E se tivéssemos dois dígitos?

10 possibilidades – Cada casa tem 10 possibilidades: 10×10 = 100.

E com *n* dígitos?

$\_\ \_\ \_\ \dots\ \_$

Capacidade de representação

Na base 10, se temos apenas um dígito, quantas possibilidades de valores podemos armazenar?

10 possibilidades – Em uma casa, podemos armazenar qualquer valor $\in$ [0-9].

E se tivéssemos dois dígitos?

10 possibilidades – Cada casa tem 10 possibilidades: 10×10 = 100.

E com *n* dígitos?

10^n

_ _ _ ... _

Capacidade de representação

Na base 2, se temos apenas um bit, quantas possibilidades de valores podemos armazenar?

E com dois bits?

E com *n* bits?

Capacidade de representação

Na base 2, se temos apenas um bit, quantas possibilidades de valores podemos armazenar?

2 possibilidades - Em uma casa, podemos armazenar qualquer valor $\in [0\text{-}1]$.

E com dois bits?

4 possibilidades - Cada casa tem 2 possibilidades: $2 \times 2 = 4$.

E com *n* bits?

2^n

Capacidade de representação

Em uma base β, se temos apenas uma casa, quantas possibilidades de valores podemos armazenar?

E com duas casas?

E com *n* casas?

Capacidade de representação

Em uma base β, se temos apenas uma casa, quantas possibilidades de valores podemos armazenar?

β possibilidades – Em uma casa, podemos armazenar qualquer valor $\in [0\text{-}(\beta\text{-}1)]$.

E com duas casas?

β^2 possibilidades – Cada casa tem β possibilidades: $\beta \times \beta = \beta^2$.

E com *n* casas?

β^n

Conversão decimal para binário

Considere o número 23 que deve ser convertido para binário.

Sabemos que o número binário vai ter o formato $(b_j b_{j-1} \ldots b_2 b_1 b_0)_2$, onde cada b_k é 0 ou 1.

O raciocínio para a conversão é pensar em quantas vezes:

1 unidade do 23 cabe em b_0;

2 unidades do 23 cabe em b_1;

4 unidades do 23 cabe em b_2;

...

Conversão decimal para binário

Considere o número 23 que deve ser convertido para binário.

Sabemos que o número binário vai ter o formato $(b_j b_{j-1} ... b_2 b_1 b_0)_2$, onde cada b_k é 0 ou 1.

O raciocínio para a conversão é pensar em quantas vezes:

1 unidade do 23 cabe em b_0;

2 unidades do 23 cabe em b_1;

4 unidades do 23 cabe em b_2;

...

Para isso, usa-se o processo de **sucessivas divisões inteiras.**

Conversão decimal para binário

Considere o número 23 que deve ser convertido para binário.

$$\begin{array}{r|l} 23 & 2 \\ \hline \end{array}$$

Conversão decimal para binário

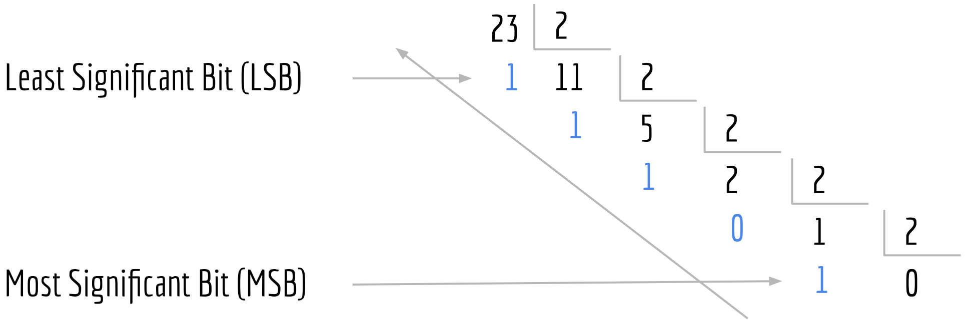
Considere o número 23 que deve ser convertido para binário.

23	2				
1	11	2			
	1	5	2		
		1	2	2	
			0	1	2
				1	0

Critério de parada: **quando o quociente chegar em zero, pare.**

Conversão decimal para binário

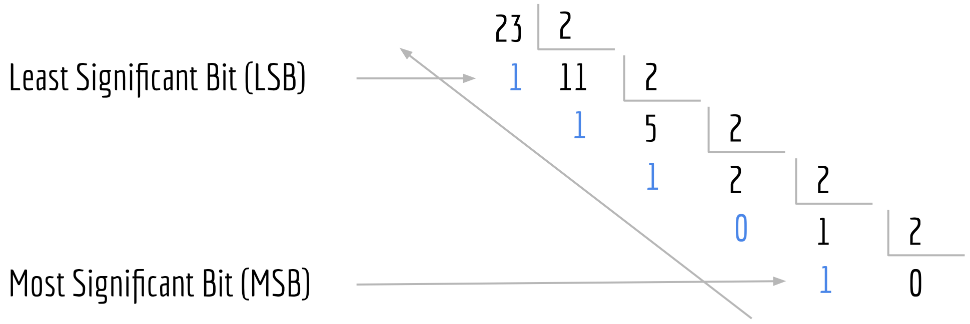
Considere o número 23 que deve ser convertido para binário.



Leia "de baixo para cima" os restos para ter o número convertido.

Conversão decimal para binário

Considere o número 23 que deve ser convertido para binário.



Logo, $23_{10} = 10111_2$

Leia “de baixo para cima” os restos para ter o número convertido.

De maneira geral

De maneira geral, como converter da base 10 para uma base β?

De maneira geral

De maneira geral, como converter da base 10 para uma base β?

Realize sucessivas divisões inteiras por β, utilizando o resto da divisão como valor convertido.

Exemplo

Considere o número 23 que deve ser convertido para a base 5.

Exemplo

Considere o número 23 que deve ser convertido para a base 5.

$$23_{10} = 43_5$$

$$\begin{array}{lll}
23 & \multicolumn{1}{|l}{5} & \\ \cline{2-2}
3 & 4 & \multicolumn{1}{|l}{5} \\ \cline{3-3}
 & 4 & 0
\end{array}$$

Valores racionais

Vamos considerar valores racionais com truncamento.

Por exemplo, 20/3 = 6,6666 com 4 casas após a vírgula e truncamento.

Valores racionais

Vamos considerar valores racionais com truncamento.

Por exemplo, 20/3 = 6,6666 com 4 casas após a vírgula e truncamento.

Para converter da base 10 para a base 2, considerando números depois da vírgula.

> Converta a parte inteira utilizando o método das divisões sucessivas.
>
> A parte fracionária será um valor $r_{10} \in [0,1)$.
>
> > Converter para um número $r_2 = ((0, d_1), d_2, \ldots, d_j)_2$, onde d_i, $1 \leq i \leq j$, é o bit na posição i do número.

Conversão

Para converter um valor $r_{10} \in [0,1)$ para binário, utilize sucessivas multiplicações por 2.

Algoritmo:

Entrada: r_{10}, entre 0 e 1.

Saída: r_2 representado por $((0, d_1),d_2 , \ldots, d_j)$.

$k = 1, F = r_{10}$
Faça:
    $F = 2 \times F$
    $d_k = parteInteira(F)$
    $F = F - d_k$
    $k = k + 1$
Enquanto $(F > 0)$

Exemplo

Converter $3{,}125_{10}$ para binário.

Primeiro converter a parte inteira utilizando o método das divisões sucessivas $3_{10} = 11_2$.

$k = 1, F = r_{10}$

Faça:

$\quad F = 2 \times F$

$\quad d_k = parteInteira(F)$

$\quad F = F - d_k$

$\quad k = k + 1$

Enquanto $(F > 0)$

Exemplo

Converter $3{,}125_{10}$ para binário.

Utilizar o algoritmo para converter a parte fracionária $r_{10} = 0{,}125$.

$k = 1, F = r_{10}$

Faça:

$\quad F = 2 \times F$

$\quad d_k = parteInteira(F)$

$\quad F = F - d_k$

$\quad k = k + 1$

Enquanto $(F > 0)$

Exemplo

Converter $3{,}125_{10}$ para binário.

K	F	$0,d_1d_2d_3d_4...$
1	0,125	0,

$k = 1, F = r_{10}$

Faça:

$\quad F = 2 \times F$

$\quad d_k = parteInteira(F)$

$\quad F = F - d_k$

$\quad k = k + 1$

Enquanto $(F > 0)$

Exemplo

Converter $3{,}125_{10}$ para binário.

K	F	$0,d_1d_2d_3d_4...$
1	0,125	0,
	0,25	

$k = 1, F = r_{10}$

Faça:

$\quad F = 2 \times F$

$\quad d_k = parteInteira(F)$

$\quad F = F - d_k$

$\quad k = k + 1$

Enquanto $(F > 0)$

Exemplo

Converter $3{,}125_{10}$ para binário.

K	F	$0,d_1d_2d_3d_4\ldots$
1	0,125	0,
	0,25	0,0

$$k = 1, F = r_{10}$$
Faça:
$$F = 2 \times F$$
$$\boxed{d_k = parteInteira(F)}$$
$$F = F - d_k$$
$$k = k + 1$$
Enquanto $(F > 0)$

Exemplo

Converter $3{,}125_{10}$ para binário.

K	F	$0,d_1d_2d_3d_4$...
1	0,125	0,
	0,25	0,0
	0,25	

$k = 1, F = r_{10}$
Faça:
    $F = 2 \times F$
    $d_k = parteInteira(F)$
    $\boxed{F = F - d_k}$
    $k = k + 1$
Enquanto $(F > 0)$

Exemplo

Converter $3{,}125_{10}$ para binário.

K	F	$0{,}d_1d_2d_3d_4...$
1	0,125	0,
	0,25	0,0
2	0,25	

$k = 1, F = r_{10}$

Faça:

$F = 2 \times F$

$d_k = parteInteira(F)$

$F = F - d_k$

$k = k + 1$

Enquanto $(F > 0)$

Exemplo

Converter $3{,}125_{10}$ para binário.

K	F	$0,d_1d_2d_3d_4...$
1	0,125	0,
	0,25	0,0
2	0,25	
	0,5	

$k = 1, F = r_{10}$

Faça:

$\quad F = 2 \times F$

$\quad d_k = parteInteira(F)$

$\quad F = F - d_k$

$\quad k = k + 1$

Enquanto $(F > 0)$

Exemplo

Converter $3{,}125_{10}$ para binário.

K	F	$0,d_1d_2d_3d_4$...
1	0,125	0,
	0,25	0,0
2	0,25	
	0,5	0,00

1: $k = 1, F = r_{10}$
2: **Faça:**
3: $\quad F = 2 \times F$
4: $\quad \boxed{d_k = parteInteira(F)}$
5: $\quad F = F - d_k$
6: $\quad k = k + 1$
7: **Enquanto** $(F > 0)$

Exemplo

Converter $3,125_{10}$ para binário.

K	F	$0,d_1d_2d_3d_4...$
1	0,125	0,
	0,25	0,0
2	0,25	
	0,5	0,00
	0,5	

$k = 1, F = r_{10}$

Faça:

$\quad F = 2 \times F$

$\quad d_k = parteInteira(F)$

$\quad \boxed{F = F - d_k}$

$\quad k = k + 1$

Enquanto $(F > 0)$

Exemplo

Converter $3{,}125_{10}$ para binário.

K	F	$0,d_1d_2d_3d_4$...
1	0,125	0,
	0,25	0,0
2	0,25	
	0,5	0,00
3	0,5	

$k = 1, F = r_{10}$
Faça:
$\quad F = 2 \times F$
$\quad d_k = parteInteira(F)$
$\quad F = F - d_k$
$\quad \boxed{k = k + 1}$
Enquanto $(F > 0)$

Exemplo

Converter $3{,}125_{10}$ para binário.

K	F	$0{,}d_1d_2d_3d_4\ldots$
1	0,125	0,
	0,25	0,0
2	0,25	
	0,5	0,00
3	0,5	
	1	

$k = 1, F = r_{10}$
Faça:
$\quad \boxed{F = 2 \times F}$
$\quad d_k = parteInteira(F)$
$\quad F = F - d_k$
$\quad k = k + 1$
Enquanto $(F > 0)$

Exemplo

Converter $3{,}125_{10}$ para binário.

K	F	$0,d_1d_2d_3d_4...$
1	0,125	0,
	0,25	0,0
2	0,25	
	0,5	0,00
3	0,5	
	1	0,001

$$
\begin{array}{l}
k = 1, F = r_{10} \\
\textbf{Faça:} \\
\quad F = 2 \times F \\
\quad \boxed{d_k = parteInteira(F)} \\
\quad F = F - d_k \\
\quad k = k + 1 \\
\textbf{Enquanto} \ \ (F > 0)
\end{array}
$$

Exemplo

Converter $3,125_{10}$ para binário.

K	F	$0,d_1d_2d_3d_4...$
1	0,125	0,
	0,25	0,0
2	0,25	
	0,5	0,00
3	0,5	
	1	0,001
	0	

1: $k = 1, F = r_{10}$
2: **Faça:**
3: $\quad F = 2 \times F$
4: $\quad d_k = parteInteira(F)$
5: $\quad F = F - d_k$
6: $\quad k = k + 1$
7: **Enquanto** $(F > 0)$

Exemplo

Converter $3{,}125_{10}$ para binário.

K	F	$0{,}d_1d_2d_3d_4\ldots$
1	0,125	0,
	0,25	0,0
2	0,25	
	0,5	0,00
3	0,5	
	1	0,001
4	0	

$k = 1, F = r_{10}$

Faça:

$\quad F = 2 \times F$

$\quad d_k = parteInteira(F)$

$\quad F = F - d_k$

$\quad \boxed{k = k + 1}$

Enquanto $(F > 0)$

Exemplo

Converter $3{,}125_{10}$ para binário.

K	F	$0,d_1d_2d_3d_4...$
1	0,125	0,
	0,25	0,0
2	0,25	
	0,5	0,00
3	0,5	
	1	0,001
4	0	

$k = 1, F = r_{10}$
Faça:
$\quad F = 2 \times F$
$\quad d_k = parteInteira(F)$
$\quad F = F - d_k$
$\quad k = k + 1$
Enquanto $(F > 0)$

Exemplo

Dessa forma:

$3_{10} = 11_2$

$0{,}125_{10} = 0{,}001_2$

Então $3{,}125_{10} = 11{,}001_2$

Faça você mesmo

Converta $0{,}1_{10}$ para binário.

Faça você mesmo

Converta $0{,}1_{10}$ para binário.

$0{,}1_{10} = 0{,}000110011\overline{0011}\ldots_2$

$0{,}1_{10}$ **não tem representação finita em binário.**

Um computador armazena uma aproximação do número.

Faça você mesmo

Converta $0,1_{10}$ para binário.

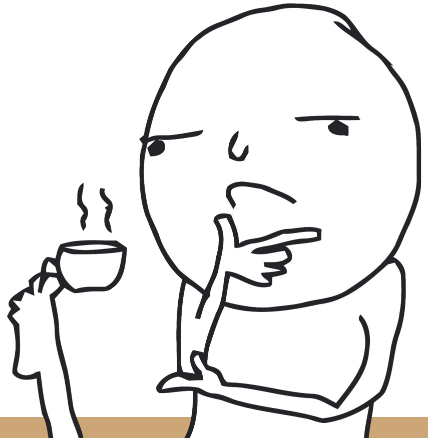
$0,1_{10} = 0,000110011\overline{0011}\ldots_2$

$0,1_{10}$ **não tem representação finita em binário.**

Um computador armazena uma aproximação do número.

Em decimal teríamos uma dízima.

Em binário talvez possamos chamar de **binarízima.**



Falha Catastrófica

25 Fev. 1991, Guerra do Golfo.

Sistema antimísseis Patriot.

O sistema falhou ao tentar interceptar um míssil Scud saudita lançado contra uma base americana.

28 Pessoas morreram e 100 ficaram feridas.

Falha Catastrófica

Causa: sistemas de radar dependem fortemente do tempo (de relógio).

O tempo no sistema era computado a cada 0.1 segundos, e armazenado utilizando 24 bits.

Aprendemos que isso não pode ser representado perfeitamente em binário.

O erro se acumulou durante 24 horas, até que não foi mais possível calcular a precisão a rota dos mísseis.

https://www-users.cse.umn.edu/~arnold/disasters/patriot.html
http://www.cs.unc.edu/~smp/COMP205/LECTURES/ERROR/lec23/node4.html
https://www-users.cse.umn.edu/~arnold/disasters/Patriot-dharan-skeel-siam.pdf

Explicando

O conteúdo da aula explica **em partes** o motivo desse programa não exibir exatamente o resultado esperado.

```
#include<stdio.h>

int main(){
	double teste = 0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1;

	printf("%.16f\n", teste);

	return 0;
}
```

Explicando

O conteúdo da aula explica **em partes** o motivo desse programa não exibir exatamente o resultado esperado

Na verdade as coisas são um pouco mais complicadas.

O hardware/software **geralmente** utiliza algo que chamamos de **ponto flutuante.**

Mas o que o ponto flutuante faz é adicionar mais fontes de erro, além das discutidas durante a aula.

Obviamente ele também tem suas vantagens. Aprenda na disciplina de **Arquitetura de Computadores**.

```c
#include<stdio.h>

int main(){
	double teste = 0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1;

	printf("%.16f\n", teste);

	return 0;
}
```

Exercícios

1. Considere que vamos armazenar uma medição na memória. Considerando que a medição pode ser qualquer valor entre 0 e os valores informados a seguir, indique quantos bits precisamos, no mínimo, para armazenar essas medições.
    a. 5
    b. 15
    c. 16
    d. 190
    e. De maneira geral, para um número n_{10}, quantos bits no mínimo são necessários para se representar esse número na base 2?
2. Converta os seguintes números da base decimal para as bases especificadas:

A. 251_{10} para base 2
B. 128_{10} para base 2
C. 143_{10} para base 8
D. 73_{10} para base 8

Exercícios

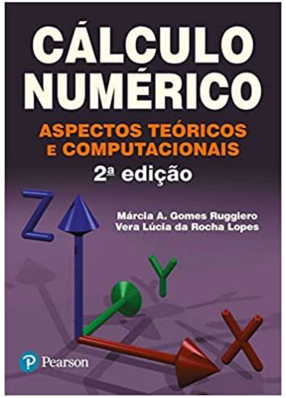
3. Converta os valores para binário:

A. 0,375
B. 0,25
C. 47,1217
D. 255,59375

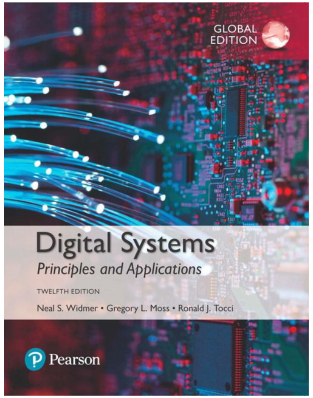
4. Em binário, um número que termina com 1 é ímpar, e um número que termina com 0 é par. Como você pode provar que isso é verdade para qualquer número binário? Descreva, mesmo que informalmente.

Referências

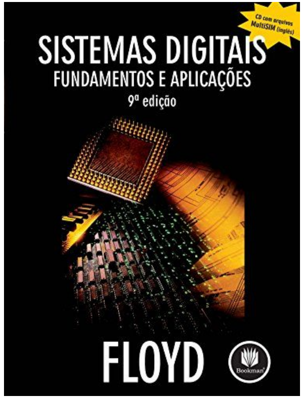
Marcia A. G. Ruggiero, Vera L. R. Lopes. Cálculo numérico aspectos teóricos e computacionais. 1996.



Ronald J. Tocci, Gregory L. Moss, Neal S. Widmer. Sistemas digitais. 10a ed. 2017.



Thomas Floyd. Widmer. Sistemas Digitais: Fundamentos e Aplicações. 2009.



Licença