Existem 10 tipos de pessoas no mundo. As que entendem binário e as que não entendem.

Base 10 para outras bases

Paulo Ricardo Lisboa de Almeida

DSBD

UFPR
UNIVERSIDADE FEDERAL DO PARANÁ

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 é 0 ou 1.

O raciocínio para a conversão é pensar em quantas vezes

1 unidade do 23 cabe em b0

2 unidades do 23 cabe em b1

4 unidades do 23 cabe em b2

...

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 cara b é 0 ou 1.

O raciocínio para a conversão é pensar em quantas vezes

- 1 unidade do 23 cabe em b0
- 2 unidades do 23 cabe em b1
- 4 unidades do 23 cabe em b2
- ...

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

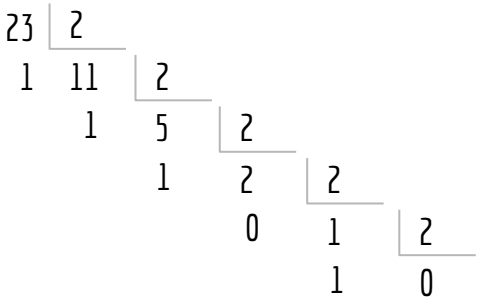
Conversão decimal para binário

Considere o número 23 que deve ser convertido para binário

23	2

Conversão decimal para binário

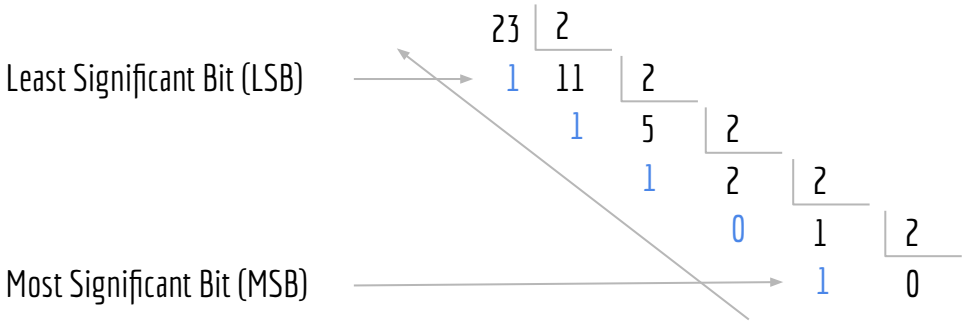
Considere o número 23 que deve ser convertido para binário



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

Conversão decimal para binário

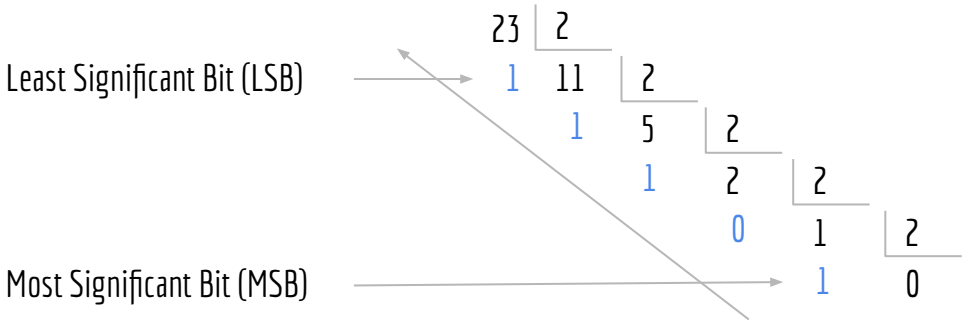
Considere o número 23 que deve ser convertido para binário



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 qualquer?

De maneira geral

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

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

$23_{10} = 43_{5}$

23	5	
3	4	5
	4	0

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 $r_2 = ((0, d_1),d_2 , ..., d_j)_2$, 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 $((0, d_1), d_2, ..., d_j)$

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

Exemplo

Converter $3{,}125_{10}$ para binário

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

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

Exemplo

Converter $3{,}125_{10}$ para binário

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

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

Exemplo

Converter $3{,}125_{10}$ para binário

K	F	$0{,}d_1d_2d_3d_4\ldots$
1	0,125	0,

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

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

1: $k = 1, F = r_{10}$
2: **Faça:**
3: $F = 2 \times F$
4: $d_k = parteInteira(F)$
5: $F = F - d_k$
6: $k = k + 1$
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
	0,25	

1: $k = 1, F = r_{10}$
2: **Faça:**
3: $F = 2 \times F$
4: $d_k = parteInteira(F)$
5: $F = F - d_k$
6: $k = k + 1$
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
1	0,25	

1: $k = 1, F = r_{10}$
2: **Faça:**
3: $F = 2 \times F$
4: $d_k = parteInteira(F)$
5: $F = F - d_k$
6: $k = k + 1$
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
1	0,25	
	0,5	

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
1	0,25	
	0,5	0,00

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
1	0,25	
	0,5	0,00
	0,5	

1: $k = 1, F = r_{10}$
2: **Faça:**
3: $F = 2 \times F$
4: $d_k = parteInteira(F)$
5: $F = F - d_k$
6: $k = k + 1$
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
1	0,25	
	0,5	0,00
2	0,5	

1: $k = 1, F = r_{10}$
2: **Faça:**
3: $F = 2 \times F$
4: $d_k = parteInteira(F)$
5: $F = F - d_k$
6: $k = k + 1$
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
1	0,25	
	0,5	0,00
2	0,5	
	1	

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
1	0,25	
	0,5	0,00
2	0,5	
	1	0,001

1: $k = 1, F = r_{10}$
2: **Faça:**
3: $F = 2 \times F$
4: $d_k = parteInteira(F)$
5: $F = F - d_k$
6: $k = k + 1$
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
1	0,25	
	0,5	0,00
2	0,5	
	1	0,001
	0	

1: $k = 1, F = r_{10}$
2: **Faça:**
3: $F = 2 \times F$
4: $d_k = parteInteira(F)$
5: $F = F - d_k$
6: $k = k + 1$
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
1	0,25	
	0,5	0,00
2	0,5	
	1	0,001
3	0	

1: $k = 1, F = r_{10}$
2: **Faça:**
3: $F = 2 \times F$
4: $d_k = parteInteira(F)$
5: $F = F - d_k$
6: $k = k + 1$
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
1	0,25	
	0,5	0,00
2	0,5	
	1	0,001
3	0	

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

Exemplo

Dessa forma

$3_{10} = 11_2$

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

Então $3{,}125_{10} = 11{,}001_2$

Exercício

Converta $0{,}1_{10}$ para binário

Exercício

Converta $0{,}1_{10}$ para binário

$0{,}1_{10} = 0{,}0001100110\overline{0011}\ldots_2$

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

Um computador binário só pode armazenar uma aproximação para o número.

Exercício

Converta $0{,}1_{10}$ para binário

$0{,}1_{10} = 0{,}0001100\overline{1100}1\ldots_2$

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

Um computador binário só pode armazenar uma aproximação para o número.

Prova: compile e execute o programa em C a seguir:

```
#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;
}
```

Exercício

Converta $0,1_{10}$ para binário

$0,1_{10} = 0,0001100110\overline{0011}..._2$

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

Um computador binário só pode armazenar uma aproximação para o número.

Prova: compile e execute o programa em C a seguir:

Obs.: não é só um problema de conversão de bases. Os cálculos em ponto flutuante também geram erros numéricos. Você vai aprender isso em outras disciplinas.

```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 temos 2 bits (duas casas binárias) para representar um número. Nesse caso, o maior número que podemos representar é $11_2 = 4_{10}$.
    a. E se tivéssemos 3 bits, qual seria o maior número (coloque sua resposta na base 10)?
    b. E se tivéssemos 8 bits, qual seria o maior número (coloque sua resposta na base 10)?
    c. De maneira geral, com *n* bits, qual o maior número que podemos representar?
2. 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?
3. 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

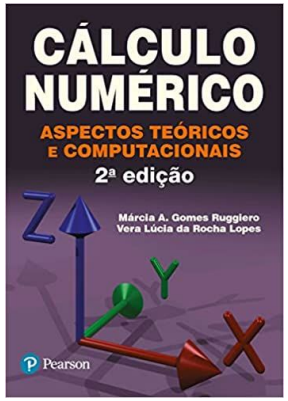
Exercícios

4. Converta os valores para binário

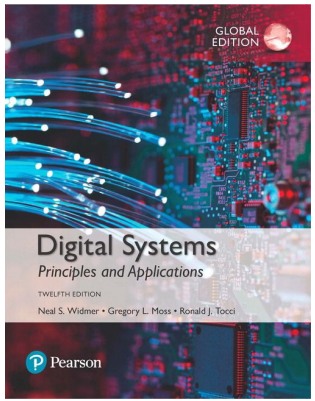
A. 0,375
B. 0,25
C. 47,1217
D. 255,59375

Referências

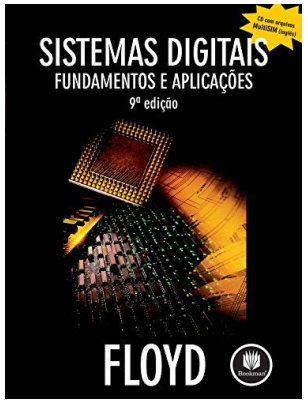
Marcia A. G. Ruggiero, Vera L. R. Lopes. Cálculo numérico aspectos teóricos e computacionais. 1996.



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



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



Licença