Conversão entre Diferentes Bases Numéricas

Prof. Americo Cunha

Universidade do Estado do Rio de Janeiro – UERJ

americo.cunha@uerj.br

www.americocunha.org

Como converter um natural de decimal para binário?

Para converter um natural N, escrito em base decimal, para a base binária utilizamos os restos de sucessivas divisões por 2.

$$N = q \times d + r$$

- N – dividendo
- d – divisor
- q – quociente
- r – resto

Qual a representação binária de 156?

Como converter um natural de decimal para binário?

Para converter um natural N, escrito em base decimal, para a base binária utilizamos os restos de sucessivas divisões por 2.

$$N = q \times d + r$$

- N – dividendo
- d – divisor
- q – quociente
- r – resto

Qual a representação binária de 156?

- $156 = 2 \times 78 + 0$

Como converter um natural de decimal para binário?

Para converter um natural N, escrito em base decimal, para a base binária utilizamos os restos de sucessivas divisões por 2.

$$N = q \times d + r$$

- N – dividendo
- d – divisor
- q – quociente
- r – resto

Qual a representação binária de 156?

- $156 = 2 \times 78 + 0$
- $78 = 2 \times 39 + 0$

Como converter um natural de decimal para binário?

Para converter um natural N, escrito em base decimal, para a base binária utilizamos os restos de sucessivas divisões por 2.

$$N = q \times d + r$$

- N – dividendo
- d – divisor
- q – quociente
- r – resto

Qual a representação binária de 156?

- $156 = 2 \times 78 + 0$
- $78 = 2 \times 39 + 0$
- $39 = 2 \times 19 + 1$

Como converter um natural de decimal para binário?

Para converter um natural N, escrito em base decimal, para a base binária utilizamos os restos de sucessivas divisões por 2.

$$N = q \times d + r$$

- N – dividendo
- d – divisor
- q – quociente
- r – resto

Qual a representação binária de 156?

- $156 = 2 \times 78 + 0$
- $78 = 2 \times 39 + 0$
- $39 = 2 \times 19 + 1$
- $19 = 2 \times 9 + 1$

Como converter um natural de decimal para binário?

Para converter um natural N, escrito em base decimal, para a base binária utilizamos os restos de sucessivas divisões por 2.

$$N = q \times d + r$$

- N – dividendo
- d – divisor
- q – quociente
- r – resto

Qual a representação binária de 156?

- $156 = 2 \times 78 + 0$
- $78 = 2 \times 39 + 0$
- $39 = 2 \times 19 + 1$
- $19 = 2 \times 9 + 1$
- $9 = 2 \times 4 + 1$

Como converter um natural de decimal para binário?

Para converter um natural N, escrito em base decimal, para a base binária utilizamos os restos de sucessivas divisões por 2.

$$N = q \times d + r$$

- N – dividendo
- d – divisor
- q – quociente
- r – resto

Qual a representação binária de 156?

- $156 = 2 \times 78 + 0$
- $78 = 2 \times 39 + 0$
- $39 = 2 \times 19 + 1$
- $19 = 2 \times 9 + 1$
- $9 = 2 \times 4 + 1$
- $4 = 2 \times 2 + 0$

Como converter um natural de decimal para binário?

Para converter um natural N, escrito em base decimal, para a base binária utilizamos os restos de sucessivas divisões por 2.

$$N = q \times d + r$$

- N – dividendo
- d – divisor
- q – quociente
- r – resto

Qual a representação binária de 156?

- $156 = 2 \times 78 + 0$
- $78 = 2 \times 39 + 0$
- $39 = 2 \times 19 + 1$
- $19 = 2 \times 9 + 1$
- $9 = 2 \times 4 + 1$
- $4 = 2 \times 2 + 0$
- $2 = 2 \times 1 + 0$

Como converter um natural de decimal para binário?

Para converter um natural N, escrito em base decimal, para a base binária utilizamos os restos de sucessivas divisões por 2.

$$N = q \times d + r$$

- N – dividendo
- d – divisor
- q – quociente
- r – resto

Qual a representação binária de 156?

- $156 = 2 \times 78 + 0$
- $78 = 2 \times 39 + 0$
- $39 = 2 \times 19 + 1$
- $19 = 2 \times 9 + 1$
- $9 = 2 \times 4 + 1$
- $4 = 2 \times 2 + 0$
- $2 = 2 \times 1 + 0$
- $1 = 2 \times 0 + 1$

Como converter um natural de decimal para binário?

Para converter um natural N, escrito em base decimal, para a base binária utilizamos os restos de sucessivas divisões por 2.

$$N = q \times d + r$$

- N – dividendo
- d – divisor
- q – quociente
- r – resto

Qual a representação binária de 156?

- $156 = 2 \times 78 + 0$
- $78 = 2 \times 39 + 0$
- $39 = 2 \times 19 + 1$
- $19 = 2 \times 9 + 1$
- $9 = 2 \times 4 + 1$
- $4 = 2 \times 2 + 0$
- $2 = 2 \times 1 + 0$
- $1 = 2 \times 0 + 1$
- $0 = 2 \times 0 + 0$

Como converter um natural de decimal para binário?

Para converter um natural N, escrito em base decimal, para a base binária utilizamos os restos de sucessivas divisões por 2.

$$N = q \times d + r$$

- N – dividendo
- d – divisor
- q – quociente
- r – resto

Qual a representação binária de 156?

- $156 = 2 \times 78 + 0$
- $78 = 2 \times 39 + 0$
- $39 = 2 \times 19 + 1$
- $19 = 2 \times 9 + 1$
- $9 = 2 \times 4 + 1$
- $4 = 2 \times 2 + 0$
- $2 = 2 \times 1 + 0$
- $1 = 2 \times 0 + 1$
- $0 = 2 \times 0 + 0$

$$(156)_{10} = (10011100)_2$$

Por que o algoritmo funciona?

Por que o algoritmo funciona?

Seja N_0 um inteiro positivo cuja representação binária possui $k+1$ dígitos, i.e., $N_0 = (b_k\, b_{k-1}\, \cdots\, b_1\, b_0)_2$

Por que o algoritmo funciona?

Seja N_0 um inteiro positivo cuja representação binária possui $k+1$ dígitos, i.e., $N_0 = (b_k\, b_{k-1}\, \cdots\, b_1\, b_0)_2$

$$N_0 = b_k \times 2^k + b_{k-1} \times 2^{k-1} + \cdots + b_1 \times 2^1 + b_0$$

Por que o algoritmo funciona?

Seja N_0 um inteiro positivo cuja representação binária possui $k+1$ dígitos, i.e., $N_0 = (b_k\, b_{k-1}\, \cdots\, b_1\, b_0)_2$

$$
\begin{aligned}
N_0 &= b_k \times 2^k + b_{k-1} \times 2^{k-1} + \cdots + b_1 \times 2^1 + b_0 \\
&= 2 \times \left(b_k \times 2^{k-1} + b_{k-1} \times 2^{k-2} + \cdots + b_2 \times 2^1 + b_1 \right) + b_0
\end{aligned}
$$

Por que o algoritmo funciona?

Seja N_0 um inteiro positivo cuja representação binária possui $k+1$ dígitos, i.e., $N_0 = (b_k\, b_{k-1}\, \cdots\, b_1\, b_0)_2$

$$\begin{aligned} N_0 &= b_k \times 2^k + b_{k-1} \times 2^{k-1} + \cdots + b_1 \times 2^1 + b_0 \\ &= 2 \times \underbrace{\left(b_k \times 2^{k-1} + b_{k-1} \times 2^{k-2} + \cdots + b_2 \times 2^1 + b_1\right)}_{N_1} + b_0 \end{aligned}$$

Por que o algoritmo funciona?

Seja N_0 um inteiro positivo cuja representação binária possui $k+1$ dígitos, i.e., $N_0 = (b_k\, b_{k-1} \cdots b_1\, b_0)_2$

$$\begin{aligned}
N_0 &= b_k \times 2^k + b_{k-1} \times 2^{k-1} + \cdots + b_1 \times 2^1 + b_0 \\
&= 2 \times \underbrace{\left(b_k \times 2^{k-1} + b_{k-1} \times 2^{k-2} + \cdots + b_2 \times 2^1 + b_1\right)}_{N_1} + b_0
\end{aligned}$$

$$N_1 = 2 \times \left(b_k \times 2^{k-2} + b_{k-1} \times 2^{k-3} + \cdots + b_3 \times 2^1 + b_2\right) + b_1$$

Por que o algoritmo funciona?

Seja N_0 um inteiro positivo cuja representação binária possui $k+1$ dígitos, i.e., $N_0 = (b_k\, b_{k-1} \cdots b_1\, b_0)_2$

$$
\begin{aligned}
N_0 &= b_k \times 2^k + b_{k-1} \times 2^{k-1} + \cdots + b_1 \times 2^1 + b_0 \\
&= 2 \times \underbrace{\left(b_k \times 2^{k-1} + b_{k-1} \times 2^{k-2} + \cdots + b_2 \times 2^1 + b_1\right)}_{N_1} + b_0
\end{aligned}
$$

$$
N_1 = 2 \times \underbrace{\left(b_k \times 2^{k-2} + b_{k-1} \times 2^{k-3} + \cdots + b_3 \times 2^1 + b_2\right)}_{N_2} + b_1
$$

Por que o algoritmo funciona?

Seja N_0 um inteiro positivo cuja representação binária possui $k+1$ dígitos, i.e., $N_0 = (b_k\, b_{k-1} \cdots b_1\, b_0)_2$

$$
\begin{aligned}
N_0 &= b_k \times 2^k + b_{k-1} \times 2^{k-1} + \cdots + b_1 \times 2^1 + b_0 \\
&= 2 \times \underbrace{\left(b_k \times 2^{k-1} + b_{k-1} \times 2^{k-2} + \cdots + b_2 \times 2^1 + b_1\right)}_{N_1} + b_0 \\
N_1 &= 2 \times \underbrace{\left(b_k \times 2^{k-2} + b_{k-1} \times 2^{k-3} + \cdots + b_3 \times 2^1 + b_2\right)}_{N_2} + b_1 \\
&\vdots
\end{aligned}
$$

Por que o algoritmo funciona?

Seja N_0 um inteiro positivo cuja representação binária possui $k+1$ dígitos, i.e., $N_0 = (b_k\, b_{k-1} \cdots b_1\, b_0)_2$

$$
\begin{aligned}
N_0 &= b_k \times 2^k + b_{k-1} \times 2^{k-1} + \cdots + b_1 \times 2^1 + b_0 \\
&= 2 \times \underbrace{\left(b_k \times 2^{k-1} + b_{k-1} \times 2^{k-2} + \cdots + b_2 \times 2^1 + b_1\right)}_{N_1} + b_0 \\
N_1 &= 2 \times \underbrace{\left(b_k \times 2^{k-2} + b_{k-1} \times 2^{k-3} + \cdots + b_3 \times 2^1 + b_2\right)}_{N_2} + b_1 \\
&\vdots \\
N_{k-1} &= 2 \times b_k + b_{k-1}
\end{aligned}
$$

Por que o algoritmo funciona?

Seja N_0 um inteiro positivo cuja representação binária possui $k+1$ dígitos, i.e., $N_0 = (b_k\, b_{k-1} \cdots b_1\, b_0)_2$

$$
\begin{aligned}
N_0 &= b_k \times 2^k + b_{k-1} \times 2^{k-1} + \cdots + b_1 \times 2^1 + b_0 \\
&= 2 \times \underbrace{\left(b_k \times 2^{k-1} + b_{k-1} \times 2^{k-2} + \cdots + b_2 \times 2^1 + b_1 \right)}_{N_1} + b_0 \\
N_1 &= 2 \times \underbrace{\left(b_k \times 2^{k-2} + b_{k-1} \times 2^{k-3} + \cdots + b_3 \times 2^1 + b_2 \right)}_{N_2} + b_1 \\
&\vdots \\
N_{k-1} &= 2 \times \underbrace{b_k}_{N_k} + b_{k-1}
\end{aligned}
$$

Por que o algoritmo funciona?

Seja N_0 um inteiro positivo cuja representação binária possui $k+1$ dígitos, i.e., $N_0 = (b_k\, b_{k-1} \cdots b_1\, b_0)_2$

$$
\begin{aligned}
N_0 &= b_k \times 2^k + b_{k-1} \times 2^{k-1} + \cdots + b_1 \times 2^1 + b_0 \\
&= 2 \times \underbrace{\left(b_k \times 2^{k-1} + b_{k-1} \times 2^{k-2} + \cdots + b_2 \times 2^1 + b_1\right)}_{N_1} + b_0 \\
N_1 &= 2 \times \underbrace{\left(b_k \times 2^{k-2} + b_{k-1} \times 2^{k-3} + \cdots + b_3 \times 2^1 + b_2\right)}_{N_2} + b_1 \\
&\vdots \\
N_{k-1} &= 2 \times \underbrace{b_k}_{N_k} + b_{k-1} \\
N_k &= b_k
\end{aligned}
$$

Por que o algoritmo funciona?

Seja N_0 um inteiro positivo cuja representação binária possui $k+1$ dígitos, i.e., $N_0 = (b_k\, b_{k-1} \cdots b_1\, b_0)_2$

$$
\begin{aligned}
N_0 &= b_k \times 2^k + b_{k-1} \times 2^{k-1} + \cdots + b_1 \times 2^1 + b_0 \\
&= 2 \times \underbrace{\left(b_k \times 2^{k-1} + b_{k-1} \times 2^{k-2} + \cdots + b_2 \times 2^1 + b_1\right)}_{N_1} + b_0 \\
N_1 &= 2 \times \underbrace{\left(b_k \times 2^{k-2} + b_{k-1} \times 2^{k-3} + \cdots + b_3 \times 2^1 + b_2\right)}_{N_2} + b_1 \\
&\vdots \\
N_{k-1} &= 2 \times \underbrace{b_k}_{N_k} + b_{k-1} \\
N_k &= b_k
\end{aligned}
$$

$N_j = 2 \times N_{j+1} + b_j \longrightarrow$ *b_j é o resto da divisão de N_j por 2*

E se tivermos um número fracionário?

$$0,625$$

E se tivermos um número fracionário?

$$0,625 = 5/8 = 1/2 + 1/8 = 2^{-1} + 2^{-3}$$

E se tivermos um número fracionário?

$$0,625 = 5/8 = 1/2 + 1/8 = 2^{-1} + 2^{-3}$$

$$0,625 = 0 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}$$

E se tivermos um número fracionário?

$$0,625 = 5/8 = 1/2 + 1/8 = 2^{-1} + 2^{-3}$$

$$0,625 = {\color{blue}0} \times 2^0 + {\color{blue}1} \times 2^{-1} + {\color{blue}0} \times 2^{-2} + {\color{blue}1} \times 2^{-3}$$

E se tivermos um número fracionário?

$$0,625 = 5/8 = 1/2 + 1/8 = 2^{-1} + 2^{-3}$$

$$0,625 = 0 \times 2^{0} + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}$$

$$(0,625)_{10} = (0,101)_{2}$$

E se tivermos um número fracionário?

$$0,625 = 5/8 = 1/2 + 1/8 = 2^{-1} + 2^{-3}$$

$$0,625 = 0 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}$$

$$(0,625)_{10} = (0,101)_2$$

$\Longrightarrow$ Representação finita

E se tivermos um número fracionário?

$$0,1$$

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

Multiplicando por 2:

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

Multiplicando por 2:

$$0,2 = \underbrace{b_{-1}}_{\text{inteiro}} + \underbrace{b_{-2} \times 2^{-1} + b_{-3} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

Multiplicando por 2:

$$0,2 = \underbrace{b_{-1}}_{\text{inteiro}} + \underbrace{b_{-2} \times 2^{-1} + b_{-3} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

$$\Longrightarrow b_{-1} = 0$$

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

Multiplicando por 2:

$$0,2 = \underbrace{b_{-1}}_{\text{inteiro}} + \underbrace{b_{-2} \times 2^{-1} + b_{-3} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

$$\Longrightarrow b_{-1} = 0$$

$$0,4 = \underbrace{b_{-2}}_{\text{inteiro}} + \underbrace{b_{-3} \times 2^{-1} + b_{-4} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

Multiplicando por 2:

$$0,2 = \underbrace{b_{-1}}_{\text{inteiro}} + \underbrace{b_{-2} \times 2^{-1} + b_{-3} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

$$\Longrightarrow b_{-1} = 0$$

$$0,4 = \underbrace{b_{-2}}_{\text{inteiro}} + \underbrace{b_{-3} \times 2^{-1} + b_{-4} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

$$\Longrightarrow b_{-2} = 0$$

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

Multiplicando por 2:

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

Multiplicando por 2:

$$0,8 = \underbrace{b_{-3}}_{\text{inteiro}} + \underbrace{b_{-4} \times 2^{-1} + b_{-5} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

Multiplicando por 2:

$$0,8 = \underbrace{b_{-3}}_{\text{inteiro}} + \underbrace{b_{-4} \times 2^{-1} + b_{-5} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

$$\Longrightarrow b_{-3} = 0$$

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

Multiplicando por 2:

$$0,8 = \underbrace{b_{-3}}_{\text{inteiro}} + \underbrace{b_{-4} \times 2^{-1} + b_{-5} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

$$\implies b_{-3} = 0$$

$$1 + 0,6 = \underbrace{b_{-4}}_{\text{inteiro}} + \underbrace{b_{-5} \times 2^{-1} + b_{-6} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

Multiplicando por 2:

$$0,8 = \underbrace{b_{-3}}_{\text{inteiro}} + \underbrace{b_{-4} \times 2^{-1} + b_{-5} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

$$\implies b_{-3} = 0$$

$$1 + 0,6 = \underbrace{b_{-4}}_{\text{inteiro}} + \underbrace{b_{-5} \times 2^{-1} + b_{-6} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

$$\implies b_{-4} = 1$$

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

Multiplicando por 2:

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

Multiplicando por 2:

$$1 + 0,2 = \underbrace{b_{-5}}_{\text{inteiro}} + \underbrace{b_{-6} \times 2^{-1} + b_{-7} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

Multiplicando por 2:

$$1 + 0,2 = \underbrace{b_{-5}}_{\text{inteiro}} + \underbrace{b_{-6} \times 2^{-1} + b_{-7} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

$$\Longrightarrow b_{-5} = 1$$

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

Multiplicando por 2:

$$1 + 0,2 = \underbrace{b_{-5}}_{\text{inteiro}} + \underbrace{b_{-6} \times 2^{-1} + b_{-7} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

$$\Longrightarrow b_{-5} = 1$$

$$0,4 = \underbrace{b_{-6}}_{\text{inteiro}} + \underbrace{b_{-7} \times 2^{-1} + b_{-8} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

Multiplicando por 2:

$$1 + 0,2 = \underbrace{b_{-5}}_{\text{inteiro}} + \underbrace{b_{-6} \times 2^{-1} + b_{-7} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

$$\Longrightarrow b_{-5} = 1$$

$$0,4 = \underbrace{b_{-6}}_{\text{inteiro}} + \underbrace{b_{-7} \times 2^{-1} + b_{-8} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

$$\Longrightarrow b_{-6} = 0$$

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

Multiplicando por 2:

$$1 + 0,2 = \underbrace{b_{-5}}_{\text{inteiro}} + \underbrace{b_{-6} \times 2^{-1} + b_{-7} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

$$\Longrightarrow b_{-5} = 1$$

$$0,4 = \underbrace{b_{-6}}_{\text{inteiro}} + \underbrace{b_{-7} \times 2^{-1} + b_{-8} \times 2^{-2} + \cdots}_{\text{fracionário}}$$

$$\Longrightarrow b_{-6} = 0$$

A partir desse ponto os coeficientes começam a se repetir!

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

- $b_{-1} = 0$

- $b_{-2} = 0$
- $b_{-3} = 0$
- $b_{-4} = 1$
- $b_{-5} = 1$

- $b_{-6} = 0$
- $b_{-7} = 0$
- $b_{-8} = 1$
- $b_{-9} = 1$

- $b_{-10} = 0$
- $b_{-11} = 0$
- $b_{-12} = 1$
- $b_{-13} = 1$

$\cdots$

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

- $b_{-1} = 0$

- $b_{-2} = 0$
- $b_{-3} = 0$
- $b_{-4} = 1$
- $b_{-5} = 1$

- $b_{-6} = 0$
- $b_{-7} = 0$
- $b_{-8} = 1$
- $b_{-9} = 1$

- $b_{-10} = 0$
- $b_{-11} = 0$
- $b_{-12} = 1$
- $b_{-13} = 1$

...

$$(0,1)_{10} = \left(0,0\overline{0011}\right)_2$$

E se tivermos um número fracionário?

$$0,1 = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + b_{-3} \times 2^{-3} + \cdots$$

- $b_{-1} = 0$

- $b_{-2} = 0$
- $b_{-3} = 0$
- $b_{-4} = 1$
- $b_{-5} = 1$

- $b_{-6} = 0$
- $b_{-7} = 0$
- $b_{-8} = 1$
- $b_{-9} = 1$

- $b_{-10} = 0$
- $b_{-11} = 0$
- $b_{-12} = 1$
- $b_{-13} = 1$

$\cdots$

$$(0,1)_{10} = \left(0,0\overline{0011}\right)_2$$

$\Longrightarrow$ Representação infinita periódica (dízima em base 2)

E se quisermos ir de binário para decimal?

$$(1101,01)_2$$

E se quisermos ir de binário para decimal?

$$(1101,01)_2 = (1101)_2 + (0,01)_2$$

E se quisermos ir de binário para decimal?

$$(1101,01)_2 = (1101)_2 + (0,01)_2$$

$$(1101)_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

E se quisermos ir de binário para decimal?

$$(1101,01)_2 = (1101)_2 + (0,01)_2$$

$$(1101)_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 13$$

E se quisermos ir de binário para decimal?

$$(1101,01)_2 = (1101)_2 + (0,01)_2$$

$$(1101)_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 13$$

$$(0,01)_2 = 0 \times 2^{-1} + 1 \times 2^{-2}$$

E se quisermos ir de binário para decimal?

$$(1101,01)_2 = (1101)_2 + (0,01)_2$$

$$(1101)_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 13$$

$$(0,01)_2 = 0 \times 2^{-1} + 1 \times 2^{-2} = 0{,}25$$

E se quisermos ir de binário para decimal?

$$(1101,01)_2 = (1101)_2 + (0,01)_2$$

$$(1101)_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = \ 13$$

$$(0,01)_2 = 0 \times 2^{-1} + 1 \times 2^{-2} = \ 0{,}25$$

E se quisermos ir de binário para decimal?

$$(1101,01)_2 = (1101)_2 + (0,01)_2$$

$$(1101)_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = \ 13$$

$$(0,01)_2 = 0 \times 2^{-1} + 1 \times 2^{-2} = \ 0{,}25$$

$$\boxed{(1101,01)_2 = (13,25)_{10}}$$

Algoritmo de conversão entre decimal e binário

Input: $N = (d_0\ ,\ 0)_{10}$

```
k = 0
Compute r and q such that N = 2 × q + r
b_0 = r
while  q ≠ 0 do
    k = k + 1
    N = q
    Compute r and q such that N = 2 × q + r
    b_k = r
end while
```

Output: $N = (b_k\, b_{k-1}\ \cdots\ b_1\, b_0)_2$

Exercício computacional:

Implemente o algoritmo acima no ambiente GNU Octave.

Para pensar em casa ...

Exercício teórico:

E se tivermos um número irracional? Pense como você poderia obter uma representação binária para π.

Exercício computacional:

Implemente no ambiente GNU Octave um algoritmo para obter uma representação binária para um dado decimal fracionário.

Exercício teórico-computacional:

Pense num algoritmo para converter um decimal inteiro para uma base arbitrária β. Implemente esse algoritmo no ambiente GNU Octave.

Como citar esse material?

A. Cunha, *Conversão entre Diferentes Bases Numéricas*, Universidade do Estado do Rio de Janeiro – UERJ, 2020.