# Números e análise do erro

Adenilton J. da Silva
ajsilva@cin.ufpe.br

Centro de Informática
UFPE

Universidade Federal de Pernambuco

# Objetivos

- Entender como os números são representados no computador.
- Análise dos erros numéricos

## Seção 1

# Representação dos números inteiros

# Conversão de números

## Para a base decimal

- Os computadores armazenam os números utilizando uma base binária.

# Conversão de números

## Para a base decimal

- Os computadores armazenam os números utilizando uma base binária.
- No dia a dia utilizamos a base decimal. Por exemplo,

$$347 =$$

$$3 \cdot 10^2 + 4 \cdot 10^1 + 7 \cdot 10^0$$

# Conversão de números

## Para a base decimal

- Os computadores armazenam os números utilizando uma base binária.
- No dia a dia utilizamos a base decimal. Por exemplo,

$$347 =$$

$$3 \cdot 10^2 + 4 \cdot 10^1 + 7 \cdot 10^0$$

- Um número com $n$ dígitos na base $\beta$.

$$(d_{n-1}d_{n-2}\cdots d_1 d_0)_\beta =$$

$$d_{n-1} \cdot \beta^{n-1} + d_{n-2} \cdot \beta^{n-2} + \cdots d_1 \cdot \beta^1 + d_0 \cdot \beta^0$$

Onde $0 \leq d_j < \beta$

# Conversão de números

Para a base decimal

- $(1011)_2 =$

# Conversão de números

## Para a base decimal

- $(1011)_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 11$

# Conversão de números

## Para a base decimal

- $(1011)_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 11$
- $(312)_8 =$

# Conversão de números

## Para a base decimal

- $(1011)_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 11$
- $(312)_8 = 3 \cdot 8^2 + 1 \cdot 8 + 2 \cdot 8^0 = 202$

# Conversão de números

## Para a base decimal

- $(1011)_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 11$
- $(312)_8 = 3 \cdot 8^2 + 1 \cdot 8 + 2 \cdot 8^0 = 202$
- $(12E)_{16} =$

Na base 16 ou base hexadecimal

| | |
|---|---|
| A | 10 |
| B | 11 |
| C | 12 |
| D | 13 |
| E | 14 |
| F | 15 |

# Conversão de números

## Para a base decimal

- $(1011)_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 11$
- $(312)_8 = 3 \cdot 8^2 + 1 \cdot 8 + 2 \cdot 8^0 = 202$
- $(12E)_{16} = \ 1 \cdot 16^2 + 2 \cdot 16^1 + 14 \cdot 16^0 = 302$

Na base 16 ou base hexadecimal

| | |
|---|---|
| A | 10 |
| B | 11 |
| C | 12 |
| D | 13 |
| E | 14 |
| F | 15 |

# Inteiros

## Conversão decimal para binário

- Considere o número na base decimal 347 e seja $(d_j d_{j-1} \cdots d_1 d_0)_2$ a representação de 347 na base 2.

# Inteiros

## Conversão decimal para binário

- Considere o número na base decimal 347 e seja $(d_j d_{j-1} \cdots d_1 d_0)_2$ a representação de 347 na base 2.

$$347 = d_j \cdot 2^j + d_{j-1} \cdot 2^{j-1} + \cdots + d_1 \cdot 2 + d_0$$

# Inteiros

## Conversão decimal para binário

- Considere o número na base decimal 347 e seja $(d_j d_{j-1} \cdots d_1 d_0)_2$ a representação de 347 na base 2.

$$347 = d_j \cdot 2^j + d_{j-1} \cdot 2^{j-1} + \cdots + d_1 \cdot 2 + d_0$$

$$347 = 2 \cdot \left(d_j \cdot 2^{j-1} + d_{j-1} \cdot 2^{j-2} + \cdots + d_1\right) + d_0$$

# Inteiros

## Conversão decimal para binário

- Considere o número na base decimal 347 e seja $(d_j d_{j-1} \cdots d_1 d_0)_2$ a representação de 347 na base 2.

$$347 = d_j \cdot 2^j + d_{j-1} \cdot 2^{j-1} + \cdots + d_1 \cdot 2 + d_0$$

$$347 = 2 \cdot \left( d_j \cdot 2^{j-1} + d_{j-1} \cdot 2^{j-2} + \cdots + d_1 \right) + d_0$$

- O dígito $d_0$ é o resto da divisão de 347 por 2.

# Inteiros

## Conversão decimal para binário

- Considere o número na base decimal 347 e seja $(d_j d_{j-1} \cdots d_1 d_0)_2$ a representação de 347 na base 2.

$$347 = d_j \cdot 2^j + d_{j-1} \cdot 2^{j-1} + \cdots + d_1 \cdot 2 + d_0$$

$$347 = 2 \cdot \left(d_j \cdot 2^{j-1} + d_{j-1} \cdot 2^{j-2} + \cdots + d_1\right) + d_0$$

- O dígito $d_0$ é o resto da divisão de 347 por 2.
- A divisão inteira de 347 por 2 é $173 = d_j \cdot 2^{j-1} + d_{j-1} \cdot 2^{j-2} + \cdots + d_1$

# Inteiros

Conversão decimal para binário

| n | n // 2 | n % 2 |
|---|---|---|
| 347 | | |

# Inteiros

Conversão decimal para binário

| n | n // 2 | n % 2 |
|---|---|---|
| 347 | 173 | 1 |

# Inteiros

## Conversão decimal para binário

| n | n // 2 | n % 2 |
|---|---|---|
| 347 | 173 | 1 |
| 173 | 86 | 1 |

# Inteiros

## Conversão decimal para binário

| n | n // 2 | n % 2 |
|---|---|---|
| 347 | 173 | 1 |
| 173 | 86 | 1 |
| 86 | 43 | 0 |

# Inteiros

## Conversão decimal para binário

| n | n // 2 | n % 2 |
|---|---|---|
| 347 | 173 | 1 |
| 173 | 86 | 1 |
| 86 | 43 | 0 |
| 43 | 21 | 1 |

# Inteiros

Conversão decimal para binário

| n | n // 2 | n % 2 |
|---|---|---|
| 347 | 173 | 1 |
| 173 | 86 | 1 |
| 86 | 43 | 0 |
| 43 | 21 | 1 |
| 21 | 10 | 1 |

# Inteiros

Conversão decimal para binário

| n | n // 2 | n % 2 |
|---|---|---|
| 347 | 173 | 1 |
| 173 | 86 | 1 |
| 86 | 43 | 0 |
| 43 | 21 | 1 |
| 21 | 10 | 1 |
| 10 | 5 | 0 |

# Inteiros

Conversão decimal para binário

| n | n // 2 | n % 2 |
|---|---|---|
| 347 | 173 | 1 |
| 173 | 86 | 1 |
| 86 | 43 | 0 |
| 43 | 21 | 1 |
| 21 | 10 | 1 |
| 10 | 5 | 0 |
| 5 | 2 | 1 |

# Inteiros

Conversão decimal para binário

| n | n // 2 | n % 2 |
|---|---|---|
| 347 | 173 | 1 |
| 173 | 86 | 1 |
| 86 | 43 | 0 |
| 43 | 21 | 1 |
| 21 | 10 | 1 |
| 10 | 5 | 0 |
| 5 | 2 | 1 |
| 2 | 1 | 0 |

# Inteiros

## Conversão decimal para binário

| n | n // 2 | n % 2 |
|---|---|---|
| 347 | 173 | 1 |
| 173 | 86 | 1 |
| 86 | 43 | 0 |
| 43 | 21 | 1 |
| 21 | 10 | 1 |
| 10 | 5 | 0 |
| 5 | 2 | 1 |
| 2 | 1 | 0 |
| 1 | 0 | 1 |

# Inteiros

## Conversão decimal para binário

| n | n // 2 | n % 2 |
|---|---|---|
| 347 | 173 | 1 |
| 173 | 86 | 1 |
| 86 | 43 | 0 |
| 43 | 21 | 1 |
| 21 | 10 | 1 |
| 10 | 5 | 0 |
| 5 | 2 | 1 |
| 2 | 1 | 0 |
| 1 | 0 | 1 |

Representação binária de 347

$(101011011)_2$.

# Seção 2

## `Representação de números com parte fracionária`

# Números com parte fracionária

## Exemplos

- $0.234 = 2 \cdot 10^{-1} + 3 \cdot 10^{-2} + 4 \cdot 10^{-3}$

# Números com parte fracionária

## Exemplos

- $0.234 = 2 \cdot 10^{-1} + 3 \cdot 10^{-2} + 4 \cdot 10^{-3}$
- $(0.1011)_2 =$

# Números com parte fracionária

## Exemplos

- $0.234 = 2 \cdot 10^{-1} + 3 \cdot 10^{-2} + 4 \cdot 10^{-3}$
- $(0.1011)_2 = 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} + 1 \cdot 2^{-4} = 0.6875$

# Números com parte fracionária

## Exemplos

- $0.234 = 2 \cdot 10^{-1} + 3 \cdot 10^{-2} + 4 \cdot 10^{-3}$
- $(0.1011)_2 = 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} + 1 \cdot 2^{-4} = 0.6875$
- $(.17)_8 = 1 \cdot 8^{-1} + 7 \cdot 8^{-2} = 0.234375$

# Números com parte fracionária

## Conversão decimal para binário

- O número 0.125 possui uma representação binária $(0.d_1 d_2 d_3 \ldots)_2$

# Números com parte fracionária

## Conversão decimal para binário

- O número 0.125 possui uma representação binária $(0.d_1d_2d_3\ldots)_2$

$$0.125 = d_1 \cdot 2^{-1} + d_2 \cdot 2^{-2} + d_3 \cdot 2^{-3} + \cdots$$

# Números com parte fracionária

## Conversão decimal para binário

- O número 0.125 possui uma representação binária $(0.d_1 d_2 d_3 \dots)_2$

$$0.125 = d_1 \cdot 2^{-1} + d_2 \cdot 2^{-2} + d_3 \cdot 2^{-3} + \cdots$$

- Ao múltiplicar os dois lados da equação por 2

# Números com parte fracionária

## Conversão decimal para binário

- O número 0.125 possui uma representação binária $(0.d_1 d_2 d_3 \dots)_2$

$$0.125 = d_1 \cdot 2^{-1} + d_2 \cdot 2^{-2} + d_3 \cdot 2^{-3} + \cdots$$

- Ao múltiplicar os dois lados da equação por 2

$$2 \cdot 0.125 = d_1 + d_2 \cdot 2^{-1} + d_3 \cdot 2^{-2} + \cdots$$

# Números com parte fracionária

## Conversão decimal para binário

- O número 0.125 possui uma representação binária $(0.d_1 d_2 d_3 \ldots)_2$

$$0.125 = d_1 \cdot 2^{-1} + d_2 \cdot 2^{-2} + d_3 \cdot 2^{-3} + \cdots$$

- Ao múltiplicar os dois lados da equação por 2

$$2 \cdot 0.125 = d_1 + d_2 \cdot 2^{-1} + d_3 \cdot 2^{-2} + \cdots$$

- $d_1$ é a parte inteira de $2 \cdot 0.125$.

# Números com parte fracionária

## Conversão decimal para binário

| $n$ | $2 \cdot n$ | parte inteira |
|---|---|---|
| 0.625 | | |

# Números com parte fracionária

## Conversão decimal para binário

| $n$ | $2 \cdot n$ | parte inteira |
|---|---|---|
| 0.625 | 1.25 | 1 |

# Números com parte fracionária

## Conversão decimal para binário

| $n$ | $2 \cdot n$ | parte inteira |
|---|---|---|
| 0.625 | 1.25 | 1 |
| 0.25 | | |

# Números com parte fracionária

## Conversão decimal para binário

| $n$ | $2 \cdot n$ | parte inteira |
|---|---|---|
| 0.625 | 1.25 | 1 |
| 0.25 | 0.5 | 0 |

# Números com parte fracionária

## Conversão decimal para binário

| $n$ | $2 \cdot n$ | parte inteira |
|---|---|---|
| 0.625 | 1.25 | 1 |
| 0.25 | 0.5 | 0 |
| 0.5 | 1 | 1 |

# Números com parte fracionária

## Conversão decimal para binário

| $n$ | $2 \cdot n$ | parte inteira |
|---|---|---|
| 0.625 | 1.25 | 1 |
| 0.25 | 0.5 | 0 |
| 0.5 | 1 | 1 |

**Representação binária de 0.625**

$(0.101)_2$.

# Seção 3

## Aritmética de ponto flutuante

# Aritmética de ponto flutuante

- Os números com parte fracionária são representados no computador com o ponto flutuante.

$$\pm(d_0.d_1 d_2 d_3 \cdots d_t) \cdot \beta^e$$

# Aritmética de ponto flutuante

- Os números com parte fracionária são representados no computador com o ponto flutuante.

$$\pm(d_0.d_1 d_2 d_3 \cdots d_t) \cdot \beta^e$$

  - $\beta$ é a base (nos computadores digitais $\beta = 2$).

# Aritmética de ponto flutuante

- Os números com parte fracionária são representados no computador com o ponto flutuante.

$$\pm(d_0.d_1 d_2 d_3 \cdots d_t) \cdot \beta^e$$

  - $\beta$ é a base (nos computadores digitais $\beta = 2$).
  - $t$ é o número de bits da mantissa.

# Aritmética de ponto flutuante

- Os números com parte fracionária são representados no computador com o ponto flutuante.

$$\pm(d_0.d_1 d_2 d_3 \cdots d_t) \cdot \beta^e$$

  - $\beta$ é a base (nos computadores digitais $\beta = 2$).
  - $t$ é o número de bits da mantissa.
  - $e$ é o expoente (representação offset binário)

# Aritmética de ponto flutuante

- Os números com parte fracionária são representados no computador com o ponto flutuante.

$$\pm(d_0.d_1 d_2 d_3 \cdots d_t) \cdot \beta^e$$

  - $\beta$ é a base (nos computadores digitais $\beta = 2$).
  - $t$ é o número de bits da mantissa.
  - $e$ é o expoente (representação offset binário)
- Dizemos que um número de ponto flutuante está na forma normal se o primeiro dígito da mantissa $d_0 = 1$.

# Aritmética de ponto flutuante

- Os números com parte fracionária são representados no computador com o ponto flutuante.

$$\pm(d_0.d_1 d_2 d_3 \cdots d_t) \cdot \beta^e$$

  - $\beta$ é a base (nos computadores digitais $\beta = 2$).
  - $t$ é o número de bits da mantissa.
  - $e$ é o expoente (representação offset binário)
- Dizemos que um número de ponto flutuante está na forma normal se o primeiro dígito da mantissa $d_0 = 1$.
- Um número de ponto flutuante subnormal é menor que $1 \cdot 2^{e_{min}}$ e neste caso $d_0$ pode ser diferente de zero.

# Ponto flutuante

## Padrão IEEE

- Base binária
- Sinal 1 bit
- Mantissa 52 bits
- Expoente 11 bits

# Ponto flutuante

python

```
import sys
print(sys.float_info)
```

sys.float_info(max=1.7976931348623157e+308, max_exp=1024, max_10_exp=308, min=2.2250738585072014e-308, min_exp=-1021, min_10_exp=-307, dig=15, mant_dig=53, epsilon=2.220446049250313e-16, radix=2, rounds=1)

# Seção 4

`Erros`

# Erros

## Tipos de erros

- Arredondamento - devido a precisão dos números de ponto flutuante.
- Discretização - Por exemplo, substituindo um limite com $x$ tendendo a 0 por $x = \epsilon \approx 0$.
- Modelagem - Ao desconsiderar algumas variáveis do problema.
- Entrada - Possíveis erros de medição.

# Erros

## Exemplo

```
import numpy as np

raiz = np.sqrt(2)
print(raiz ** 2)
```

# Erros

Exemplo

```
import numpy as np

raiz = np.sqrt(2)
print(raiz ** 2)
```

saída: 2.0000000000000004

# Erros

- O *erro absoluto* é a diferença entre o valor real e o valor aproximado de uma variável.
- O *erro relativo* é o erro absoluto dividido pelo valor real da variável.
- Exemplos raiz ** 2
  - Erro absoluto: 4.440892098500626e-16
  - Erro relativo: 2.220446049250313e-16 %

# Recomendações práticas

- Não devemos utilizar == para comparar floats.

```python
import numpy as np

raiz = np.sqrt(2)
print(raiz**2 == 2)
```

# Recomendações práticas

Mais detalhes nos exercícios.

- Cuidado com as operações que modificam a ordem de magnitude. Por exemplo,
  - Divisão por valores próximos a zero.
  - Subtração de valores próximos.

# Referências


- Métodos Numéricos, José Dias dos Santos & Zanoni Carvalho da Silva, Editora Universitária UFPE, 3ª Edição
- Solomon, Justin. Numerical algorithms: methods for computer vision, machine learning, and graphics. CRC press, 2015.