

Próximo:[Predicados em números](#), Anterior:[Noções básicas de números inteiros](#), Acima:[Números](#) [[Conteúdo](#)][[Índice](#)]

3.2 Noções Básicas de Ponto Flutuante

Os números de ponto flutuante são úteis para representar números que não são integrais. O intervalo de números de ponto flutuante é o mesmo que o intervalo do tipo de dados C `double` na máquina que você está usando. Em todos os computadores suportados pelo Emacs, este é o formato de ponto flutuante IEEE binário64, que é padronizado pelo IEEE Std 754-2019 e é discutido mais adiante no artigo de David Goldberg “[O que todo cientista de computação deve saber sobre aritmética de ponto flutuante](#)”. Em plataformas modernas, as operações de ponto flutuante seguem de perto o padrão IEEE-754; no entanto, os resultados nem sempre são arredondados corretamente em algumas plataformas obsoletas, principalmente x86 de 32 bits.

A sintaxe de leitura para números de ponto flutuante requer um ponto decimal, um expoente ou ambos. Sinais opcionais ('+' ou '-') precedem o número e seu expoente. Por exemplo, '1500,0', '+15e2', '15,0e+2', '+1500000e-3', e '.15e4' são cinco maneiras de escrever um número de ponto flutuante cujo valor é 1500. São todas equivalentes. Assim como o Common Lisp, o Emacs Lisp requer pelo menos um dígito após qualquer ponto decimal em um número de ponto flutuante; '1500.' é um número inteiro, não um número de ponto flutuante.

O Emacs Lisp trata `-0.0` como numericamente igual ao zero comum em relação a comparações numéricas como `=`. Isso segue o padrão de ponto flutuante IEEE-0.0, que diz que `0.0` e `-0.0` são numericamente iguais, embora outras operações possam distingui-los.

O padrão de ponto flutuante IEEE suporta infinito positivo e infinito negativo como valores de ponto flutuante. Ele também fornece uma classe de valores chamada NaN, ou “não é um número”; funções numéricas retornam tais valores nos casos em que não há resposta correta. Por exemplo, `(/ 0.0 0.0)` retorna um NaN. Um NaN nunca é numericamente igual a qualquer valor, nem mesmo a si mesmo. NaNs carregam um sinal e um significando, e funções não numéricas tratam dois NaNs como iguais quando seus sinais e significandos concordam. Significandos de NaNs são dependentes da máquina, assim como os dígitos em sua representação de string.

Quando NaNs e zeros com sinal estão envolvidos, funções não numéricas como `, , e eql` determinam se `sxhash-eql` os valores são indistinguíveis, não se são numericamente iguais. Por exemplo, quando `x` e `y` são o mesmo NaN, `retorna` enquanto usa comparação numérica e `retorna`; inversamente, `retorna` enquanto `retorna`. `sxhash-equalgethash(equal x y)t(= x y)nil(equal 0.0 -0.0)nil(= 0.0 -0.0)t`

Aqui estão as sintaxes lidas para esses valores especiais de ponto flutuante:

infinitude

'1,0e+INF' e '-1,0e+INF'

não-um-número

'0,0e+NaN' e '-0,0e+NaN'

As seguintes funções são especializadas para lidar com números de ponto flutuante:

Função: isnan x

Esse predicado retorna tse seu argumento de ponto flutuante for um NaN, nil caso contrário.

Função: frexp x

Essa função retorna uma célula cons , onde s e e são respectivamente o significando e o expoente do número de ponto flutuante x . (s . e)

Se x é finito, então s é um número de ponto flutuante entre 0,5 (inclusive) e 1,0 (exclusivo), e é um inteiro e $x = s * 2^{**} e$. Se x é zero ou infinito, então s é o mesmo que x . Se x é um NaN, então s também é um NaN. Se x é zero, então e é 0.

Função: ldexp s e

Dado um significando numérico s e um expoente inteiro e , esta função retorna o número de ponto flutuante $s * 2^{**} e$.

Função: copysign $x1$ $x2$

Esta função copia o sinal de $x2$ para o valor de $x1$ e retorna o resultado. $x1$ e $x2$ devem ser de ponto flutuante.

Função: logb x

Esta função retorna o expoente binário de x . Mais precisamente, se x é finito e diferente de zero, o valor é o logaritmo de base 2 de $|x|$, arredondado para um número inteiro. Se x é zero ou infinito, o valor é infinito; se x é um NaN, o valor é um NaN.

```
(log 10)
⇒ 3
(logb 10.0e20)
⇒ 69
(log 0)
⇒ -1.0e+INF
```

Próximo:[Predicados em números](#), Anterior:[Noções básicas de números inteiros](#), Acima:[Números](#) [[Conteúdo](#)] [[Índice](#)]