

Próximo:[Números](#), Anterior:[Introdução](#), Acima:[Topo](#) [Conteúdo][Índice]

## 2 Tipos de dados Lisp

Um objeto Lisp é um dado usado e manipulado por programas Lisp. Para nossos propósitos, um *tipo* ou tipo de *dados* é um conjunto de objetos possíveis.

Todo objeto pertence a pelo menos um tipo. Objetos do mesmo tipo têm estruturas semelhantes e geralmente podem ser usados nos mesmos contextos. Os tipos podem se sobrepor e os objetos podem pertencer a dois ou mais tipos. Consequentemente, podemos perguntar se um objeto pertence a um determinado tipo, mas não *ao* tipo de um objeto.

Alguns tipos de objetos fundamentais são incorporados ao Emacs. Estes, a partir dos quais todos os outros tipos são construídos, são chamados de *tipos primitivos*. Cada objeto pertence a um e apenas um tipo primitivo. Esses tipos incluem *integer*, *float*, *cons*, *symbol*, *string*, *vector*, *hash-table*, *subr*, *byte-code function* e *record*, além de vários tipos especiais, como *buffer*, relacionados à edição. (Consulte [Tipos de edição](#).)

Cada tipo primitivo tem uma função Lisp correspondente que verifica se um objeto é membro daquele tipo.

Lisp é diferente de muitas outras linguagens em que seus objetos são *autotipáveis*: o tipo primitivo de cada objeto está implícito no próprio objeto. Por exemplo, se um objeto é um vetor, nada pode tratá-lo como um número; Lisp sabe que é um vetor, não um número.

Na maioria das linguagens, o programador deve declarar o tipo de dados de cada variável, e o tipo é conhecido pelo compilador, mas não representado nos dados. Tais declarações de tipo não existem no Emacs Lisp. Uma variável Lisp pode ter qualquer tipo de valor e lembra qualquer valor que você armazene nela, tipo e tudo. (Na verdade, um pequeno número de variáveis do Emacs Lisp só pode assumir valores de um determinado tipo. Consulte [Variáveis com valores restritos](#).)

Este capítulo descreve a finalidade, a representação impressa e a sintaxe de leitura de cada um dos tipos padrão no GNU Emacs Lisp. Detalhes sobre como usar esses tipos podem ser encontrados em capítulos posteriores.

- [Representação Impressa](#) Como os objetos Lisp são representados como texto.
- [Sintaxe de leitura especial](#) Uma visão geral de todas as sequências especiais.
- [Comentários](#) Comentários e suas convenções de formatação.
- [Tipos de Programação](#) Tipos encontrados em todos os sistemas Lisp.
- [Tipos de Edição](#) Tipos específicos do Emacs.
- [Objetos Circulares](#) Leia a sintaxe para a estrutura circular.
- [Tipo de Predicados](#) Testes relacionados a tipos.
- [Predicados de Igualdade](#) Testes de igualdade entre quaisquer dois objetos.
- [Mutabilidade](#) Alguns objetos não devem ser modificados.

Próximo:[Números](#), Anterior:[Introdução](#), Acima:[Topo](#) [Conteúdo][Índice]