Próximo:Tipo de predicados, Anterior:Tipos de edição, Acima:Tipos de dados Lisp [Conteúdo][Índice]

2.6 Leia Sintaxe para Objetos Circulares

Para representar estruturas compartilhadas ou circulares dentro de um complexo de objetos Lisp, você pode usar as construções do leitor '`# n =`' e '`# n #`'.

Use antes de um objeto para rotulá-lo para referência posterior; posteriormente, você pode usar para referenciar o mesmo objeto em outro lugar. Aqui, *n* é algum número inteiro. Por exemplo, aqui está como fazer uma lista na qual o primeiro elemento se repete como o terceiro elemento: *#n=#n#*

```
(#1=(a) b#1#)
```

Isso difere da sintaxe comum como esta

```
((a)b(a))
```

o que resultaria em uma lista cujos primeiro e terceiro elementos se parecem, mas não são o mesmo objeto Lisp. Isso mostra a diferença:

```
(prog1 nil
  (setq x '(#1=(a) b #1#)))
(eq (nº 0 x) (nº 2 x))
     ⇒ t
(setq x '((a) b (a)))
(eq (nº 0 x) (nº 2 x))
     ⇒ nada
```

Você também pode usar a mesma sintaxe para fazer uma estrutura circular, que aparece como um elemento dentro de si. Aqui está um exemplo:

```
#1=(um #1#)
```

Isso cria uma lista cujo segundo elemento é a própria lista. Veja como você pode ver que realmente funciona:

```
(prog1 nil
  (setq x '#1=(a #1#)))
(eq x (cadr x))
     ⇒ t
```

A impressora Lisp pode produzir essa sintaxe para registrar a estrutura circular e compartilhada em um objeto Lisp, se você vincular a variável `print-circle` a um não `nil`valor. Consulte Variáveis de saída .

Próximo:Tipo de predicados, Anterior:Tipos de edição, Acima:Tipos de dados Lisp [Conteúdo][Índice]