Próximo:Predicados relacionados à lista, Acima:Listas [Conteúdo][Índice]

5.1 Listas e Células Contras

Listas em Lisp não são um tipo de dados primitivo; eles são construídos a partir de *células contra* (consulte Tipo de célula contra). Uma célula contra é um objeto de dados que representa um par ordenado. Ou seja, ele tem dois slots, e cada slot *contém* , ou *se refere a* , algum objeto Lisp. Um slot é conhecido como CAR e o outro é conhecido como CDR . (Esses nomes são tradicionais; veja Contras Tipo de Célula .) CDR é pronunciado “could-er”.

Dizemos que “o CAR desta célula cons é” qualquer objeto que seu slot CAR possua atualmente, e da mesma forma para o CDR .

Uma lista é uma série de células contras encadeadas, de modo que cada célula se refere à próxima. Há uma célula contras para cada elemento da lista. Por convenção, os CAR s das células cons contêm os elementos da lista, e os CDR s são usados para encadear a lista (essa assimetria entre CAR e CDR é inteiramente uma questão de convenção; no nível das células cons, o CAR e slots CDR têm propriedades semelhantes). Portanto, o slot CDR de cada célula contras em uma lista refere-se à célula contras a seguir.

Também por convenção, o CDR da última célula contras em uma lista é `nil`. Chamamos tal `nil`estrutura terminada de *lista própria* [3] . No Emacs Lisp, o símbolo `nil`é tanto um símbolo quanto uma lista sem elementos. Por conveniência, considera-se que o símbolo `nil`tem `nil`como seu CDR (e também como seu CAR).

Portanto, o CDR de uma lista adequada é sempre uma lista adequada. O CDR de uma lista própria não vazia é uma lista própria contendo todos os elementos, exceto o primeiro.

Se o CDR da última célula cons de uma lista for algum valor diferente de `nil`, chamamos a estrutura de *lista pontilhada* , pois sua representação impressa usaria a notação de par pontilhado (consulte Notação de par pontilhado). Há uma outra possibilidade: o CDR de algumas células contras poderia apontar para uma das células contras anteriores na lista. Chamamos essa estrutura de *lista circular* .

Para alguns propósitos, não importa se uma lista é própria, circular ou pontilhada. Se um programa não procurar o suficiente na lista para ver o CDR da célula de contras final, ele não se importará. No entanto, algumas funções que operam em listas exigem listas adequadas e sinalizam erros se receberem uma lista pontilhada. A maioria das funções que tentam encontrar o final de uma lista entra em loops infinitos se receber uma lista circular.

Como a maioria das células cons são usadas como parte de listas, nos referimos a qualquer estrutura feita de células cons como uma *estrutura de lista* .

Notas de rodapé

(3)

Às vezes, também é chamada de *lista verdadeira* , mas geralmente não usamos essa terminologia neste manual.

Próximo:Predicados relacionados à lista, Acima:Listas [Conteúdo][Índice]