car/cdr recursion (1/2)

- リストのトップレベルの要素だけでなく，内部の要素をも走査の対象とする時
- 奇妙な構造のリストを対象とする時

例 (sum-tree '(2 3 (1 (5)) (((2)))) ⇒ 13
(flat '(3 2 (1 (5)))) ⇒ (3 2 1 5)

57

car/cdr recurstion (2/2)

- リストを2分木とみなす.

各ノードは，アトムかconsセルとなる.

↓

アトムにぶつかるまで，car部とcdr部に再帰呼出しをかける.

58

Tail recursion

- 再帰呼出しの最終の値がそのまま関数全体の値になる.
- Tail recursionを使うと，再帰呼出しをjumpで実行できる → 効率が上がる
  (lisp compilerはこれをしている)

59

Fact のtail recursion版

- 途中結果を引数で渡していくことでtail recursionにする
- ```
  (defun tr-fact (n)
    (tr-fact-body n 1))
  (defun tr-fact-body (n kekka)
   (if (= n 1) kekka
       (tr-fact-body (- n 1) (* n kekka))))
  ```

60
  ```

- [練習] revやcount-downをtail recursionで書き換えなさい

61