# Lisp へのこだわり
# Lisp, Lisp, Lisp and Lisp

和田 英一

## 概要

Lisp への賛辞は多い.
Lisp には「再帰の心」が浸みわたっている. Doug Hofstadter
なににもまして Lisp でプログラムを書くのは大いに楽しい. Jerry Sussman
The ELI project saved several man-years by using Lisp. John Ellis
Lisp が着実に進歩し, 流布してきたのは, コンピュータの研究者や利用者の
中に狂信的ともいえる愛好者を持つことができたから. 中西正和
The most intelligent way to misuse a computer. E.W.Dijkstra
など.

なぜわれわれは Lisp へこだわり続けるのであろうか. その理由を考えたい.

## Lisp との長い付合い

1958 年 McCarthy の論文 A basis for a mathematical theory of computation を読む

1960 年頃 Lisp1.5 Programmer's Manual を読む

1964 年 IBM 計算センターで Lisp が使えるようになる
　　卒論生の東山君が Lisp で卒論を書く

1965 年 東京大学大型計算機センターに Hitac 5020 設置
　　助手の飛沢さんと Lisp 処理系を開発

1971 年頃 Delft 工科大学 van der Poel 先生から PDP-8 Lisp を貰う
　　学科の PDP-12 で使用

1973 年～74 年 MIT に滞在
　　ProjectMac で MacLisp を使う

この頃から研究室の学生が多くの Lisp 処理系を書いた
　　大型計算機センターで FranzLisp が使えた

## Lisp のいいところ

かっこさえ我慢すれば入出力は超簡単
ラムダ算法をほぼ実現している→関数が第 1 級市民
データ構造は cons(あるいは list)とその入れ子しかない
(文字列は文字の list 後に vector も実装された)
評価の規則が簡単
Lisp で定義されている
4K 語のミニコンに実装出来るほど小さい言語
データがプログラムとして実行できる
bignum が用意してある

かっこが深くなり分かりにくくなれば pretty print する
かっこの対応を表示するエディタを使う

なんでも再帰に書きたくなる（scheme では徹底している）
一番難しいところからプログラムを書くのに最適

多くの方言があるが, 1 つを知っていればあとはすぐ慣れる
基本的な部分だけの処理系ならすぐ書ける
→(evalquote in simple fortran)
エディタや新しい言語の実装にも使える

基本が解釈系なので，プログラムを教えるのに都合がよい
対話性がよい テストしながらプログラムが作れる

## Lisp で困ること

vector はあるけれども，使いにくい
マクロが置き換えでないから難しい→玄人向き
漢字コードの扱い→ターミナルエミュレータの問題

合流するデータの表示がしにくい

```
(define one '(1)) (set-cdr! one one)
(labelprint one) => #1(1 . %1)

(define foo '(a b c d)) (define bar (list foo foo foo))
(labelprint bar) => (#1(a b c d) %1 %1)
```

のようなものもある

構造エディタはついにはやらなかった

長所に比べれば短所はたいしたことではない

## 最後に

Lisp がここまで広まり使い続けられた背後には

多くのユーザ，特にハッカーの貢献
MIT とその AIlab, ProjectMAC などの努力

があったことを忘れてはならない

Lisp に夢中なのは一部の変人だけなのかもしれない.