



THIS PAGE IS  
INTENTIONALLY  
LEFT BLANK.

Kazimir Majorinc

# LISP 1.5 (II.)

Povijest Lispa 15.



Razmjena vještina  
Hacklab u mami  
8. prosinca 2012.

# LISP 1.5 interpreter sistem

Programi u LISP 1.5 su nizovi parova S-izraza koji se predaju funkciji *evalquote*[*fn*;*(arg1,...,argn)*].

```
DEFINE ((
(MEMBER (LAMBDA (A X) (COND ((NULL X) F)
  ( (EQ A (CAR X) ) T) (T (MEMBER A (CDR X)))) )))
(UNION (LAMBDA (X Y) (COND ((NULL X) Y) ((MEMBER
  (CAR X) Y) (UNION (CDR X) Y)) (T (CONS (CAR X)
  (UNION (CDR X) Y)))) )))
(INTERSECTION (LAMBDA (X Y) (COND ((NULL X) NIL)
  ( (MEMBER (CAR X) Y) (CONS (CAR X) (INTERSECTION
  (CDR X) Y))) (T (INTERSECTION (CDR X) Y)) )))
))
INTERSECTION ((A1 A2 A3) (A1 A3 A5))
UNION ((X Y Z) (U V W X))
```

Lisp 1.5 interpreter se svodi na pozivanje funkcije eval.

**(LAMBDA(X Y)(CONS X Y)) (A (B))**

*eval*quote[(LAMBDA(X Y)(CONS X Y));(A (B))]

*apply*[(LAMBDA(X Y)(CONS X Y));(A (B));NIL]

*eval*[(CONS X Y);((X . A) (Y . B))] ...

**Varijable.** Simboli koji se koriste kao argumenti funkcije; njihova vrijednost se traži u a-listi. Ako ima više vrijednosti u a-listi, koristi se posljednja unešena.

**Konstante.** Konstante su varijable koje imaju vrijednost nezavisno od a-liste (association list).

*cset*[X;(A B C D)]

Vrijednost konstante, zajedno sa indikatorom **APVAL** spremljena je u p-listi (property list) simbola.

U stvarno implementiranom interpreteru, p-lista je jača od a-liste, tj. interpreter prvo traži vrijednost simbola u p-listi.

**Funkcije.** Za definiciju funkcija koristi se pseudo funkcija `define`

*define*[**MEMBER**;(LAMBDA ...)]

Vrijednost funkcije (lambda- ili label- lista) se upisuje u p-listu simbola, zajedno sa indikatorom **EXPR**. „Činjenica da je većina funkcija korištena u programu konstantna znači da je sistem bogat i još ga ne znamo pravilno koristiti.“

**Funkcije u mašinskom jeziku.** Imaju „indikator“ **SUBR** u listi svojstava (property list). Postoje u sistemu, a moguće je dodati nove ručno ili kompajliranjem. Deset do sto puta brže.

**Specijalne forme.** Eval izračunava argumente funkcije prije primjene funkcije. Primjerice, ako funkcija eval dobije argument **(CONS X Y)**, onda se prvo izračunavaju **X** i **Y** a onda primjenjuje **CONS**. Ali, **(QUOTE X)** ne izračunava vrijednost **X**.

Specijalne forme se razlikuju od funkcija na dva načina: mogu primiti neodređeni broj argumenata i argumenti se ne izračunavaju „prije nego što ih specijalna forma vidi.“

Specijalne forme imaju indikator **FEXPR** ili **FSUBR**.

# Proširenje jezika LISP.

„Čisti LISP“ - kakav je opisan u „Recursive functions“

„Elementarni LISP“ - opisan dosada.

**Funkcionalni argumenti.** Mogućnost da funkcije budu argumenti pri pozivu drugih funkcija. Primjerice,

$$\text{maplist}[x;fn] = [\text{null}[x] \rightarrow \mathbf{NIL}; \mathbf{T} \rightarrow \text{maplist}[\text{cdr}[x];fn]].$$

Lisp opisan u „Rekurzivnim funkcijama“ nije znao kako evaluirati lambda-izraze ako nisu na mjestu operatora.

$$\text{change}[a]=\text{maplist}[a;\lambda[j];\text{cons}[\text{car}[j];\mathbf{X}]]$$

Treba nam novo pravilo za prevođenje meta-izraza u S-izraze. Ako se javlja funkcionalni argument,  $fn$ , taj argument se prevodi u **(FUNCTION  $fn^*$ )**. Prijevod gornjeg M-izraza je

**(MAPLIST**

**A**

**(FUNCTION (LAMBDA(J)(CONS (CAR J) (QUOTE X))))).**

Interpreter vidi u dodatku.



**Logički izrazi i predikati.** Logički izrazi **AND** i **OR** primaju neograničeni broj argumenata, pa su onda specijalne forme. **NIL** tretiraju kao neistinu, sve drugo kao istinu.

**\*T\*** i **NIL** su simboli koji reprezentiraju istinu i laž.

**T** i **F** su simboli sa konstantnim vrijednostima **\*T\*** i **NIL**.

$eq[e1;e2]$  je proširen i na ne-atomarne  $e1$  i  $e2$ .

**Aritmetika.** Brojevi su predstavljeni kao specijalna vrsta atomskih simbola, konstante koji se izračunavaju same u sebe. Podržani brojevi sa fiksnim i pomičnim zarezom i brojevi u oktalnom sustavu. Implementirano dvadesetak funkcija, uključujući i funkcije sa bilo kojim brojem argumenata *plus*, *times*, *min*, *max*.

**Matrice.** Pseudo-funkcija *array* rezervira prostor, i pretvara ime varijable u funkciju.

*array*[(**ALPHA (7 10) LIST**)(**BETA (3 4 5) LIST**)]

*alpha*[7;10]

*alpha*[**SET**; **C**;7;10] znači *alpha*[7,10]=**C**