This page is intentionally left blank.

Kazimir Majorinc

Prema LISPu 1.5 (II.)

Povijest Lispa 14.

Razmjena vještina
Hacklab u mami
24. studeni 2012.

1959.

McCarthy razvija „teoretski Lisp.“ Cijela zajednica razvija „praktični Lisp“.

McCarthy razvija time sharing.

David Luckham piše softver (za time sharing) u Lispu.

James R. Slagle ⟶

piše prvi program za simboličko integriranje uopće.

Advice Taker se ne spominje.

1960.

U ožujku objavljen interni priručnik „LISP I.“

U travnju objavljen „Rekurzivne funkcije ...“

Razvija se Lisp 1.2.

McCarthy u izvješću RLE QPR najavljuje Lisp 2.

LISP nije utjecajan izvan MIT.

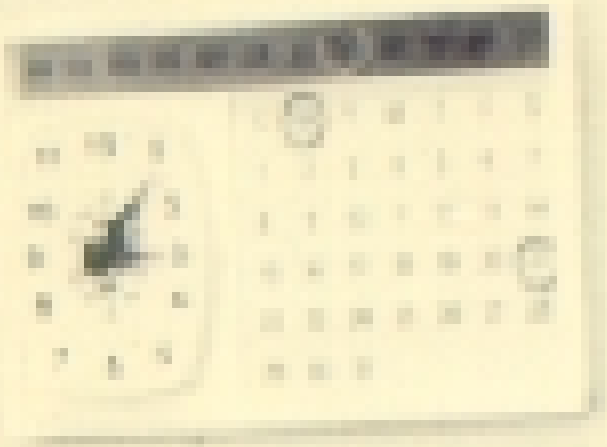
Usputno spomenut u **Joseph Smithovom** članku o ALGOL-u.

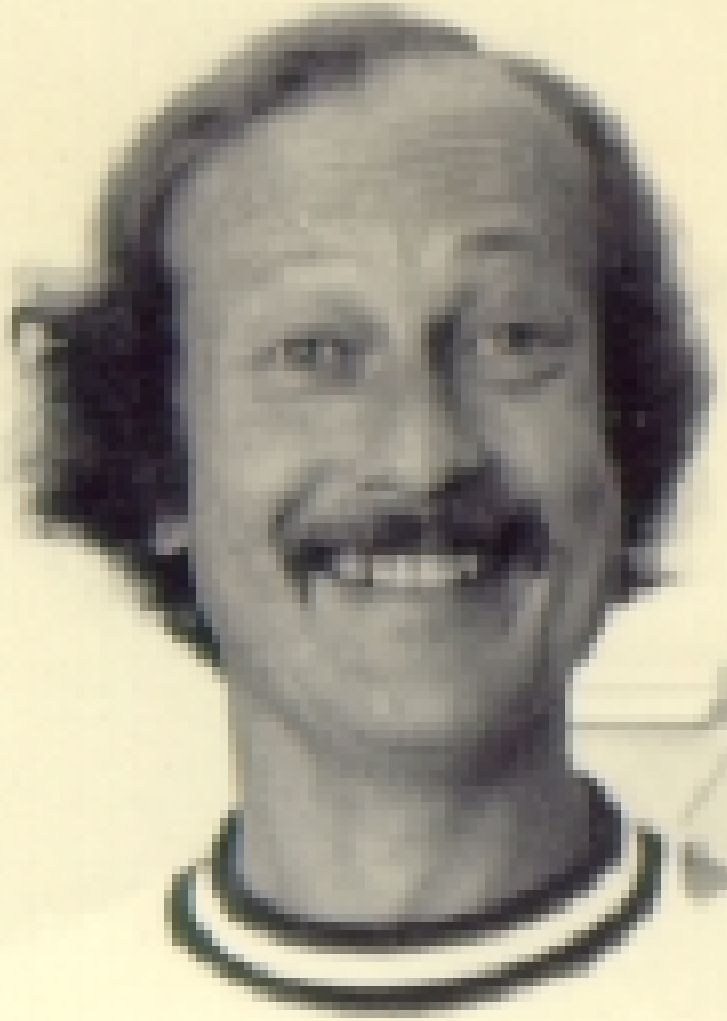
George Collins analizira upravljanje memorijom i predlaže brisanje dijelova lista (reference counting).

Advice Taker se ne spominje

David Luckham.

Pomagao Braytonu.





Zadužen za maplist.

Zajednica osjeća stagnaciju i nezadovoljna je **McCarthyjevim** vodstvom.

Studenti, uključujući **Luckhama** postepeno napuštaju projekt.

1961.

Neki istraživači sa Yale University prelaze sa IPL na LISP

LISP korišten za istraživanje u „Cambridge Air Force Lab“

Webb Comfort iz IBM koristi asember za implementaciju lista i piše „*Veliki sustavi poput IPL ili Lispa nisu potrebni.*“

McCarthy na konferenciji izlaže „*Basis for mathematical theory of computation*“, svojevrsni nastavak „Recursive functions.“ Članak pod istim naslovom objavljen tek **1963**.

Do kraja **1961.** objavljeno 30-ak memoa. Uglavnom nisu pogodni za izlaganje. (Manje važni detalji, rane verzije ili su kasnije bolje opisani u nezaobilaznom priručniku 1.5.)

Aktivna grupa pod vodstvom kvantnog fizičara **Harolda V. McIntosha** u Research Institute for Advanced Studies (RIAS) u Baltimoreu. Implementirao Lisp na IBM 709 i 7090.

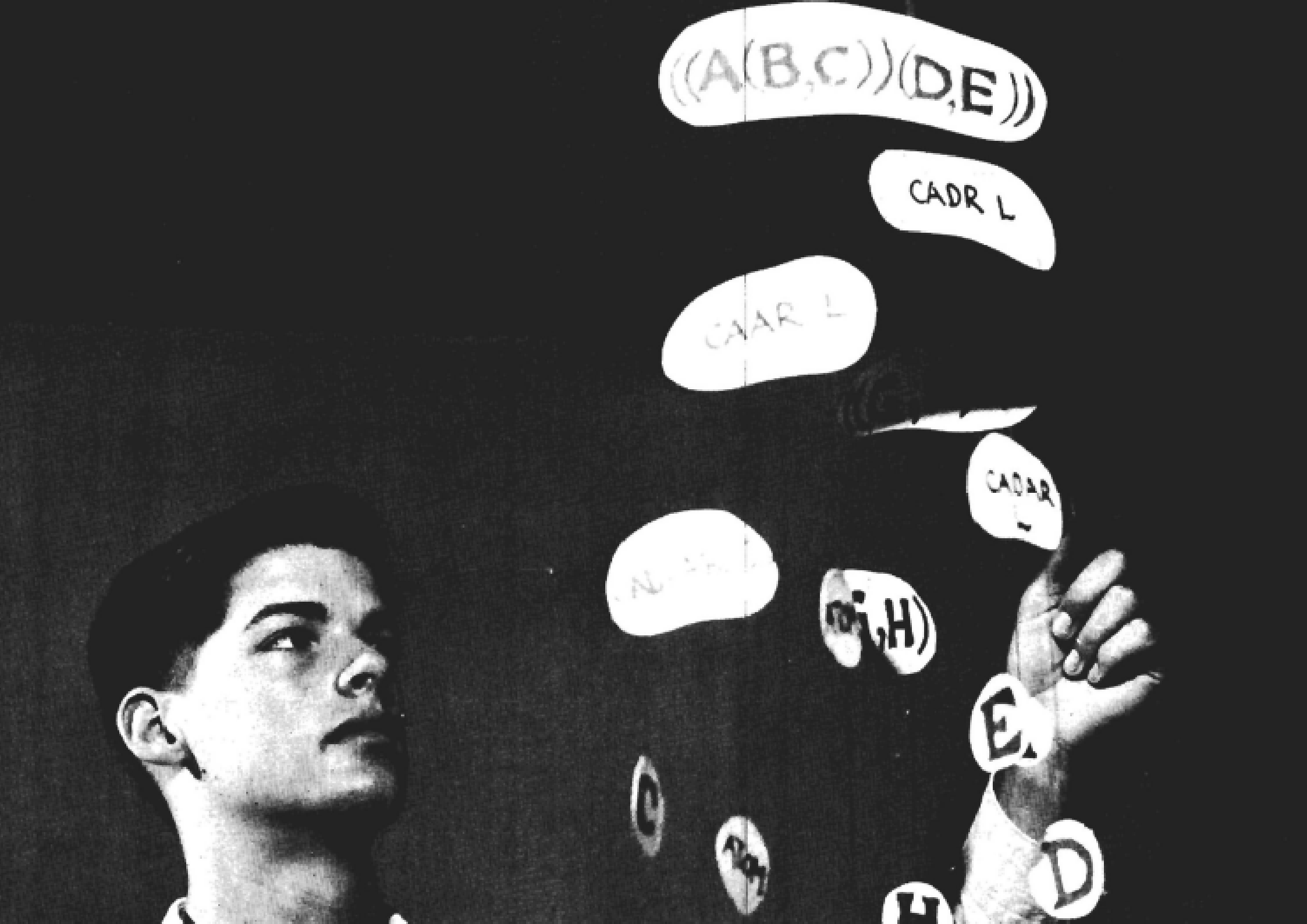
Handbook of LISP functions. Osnovne funkcije i funkcije koje su sami razvili.

Primjerice. `(RANK L S)` - sort, `POLISH` i `POSTPOLISH`. Tornjevi Hanoja i slično.

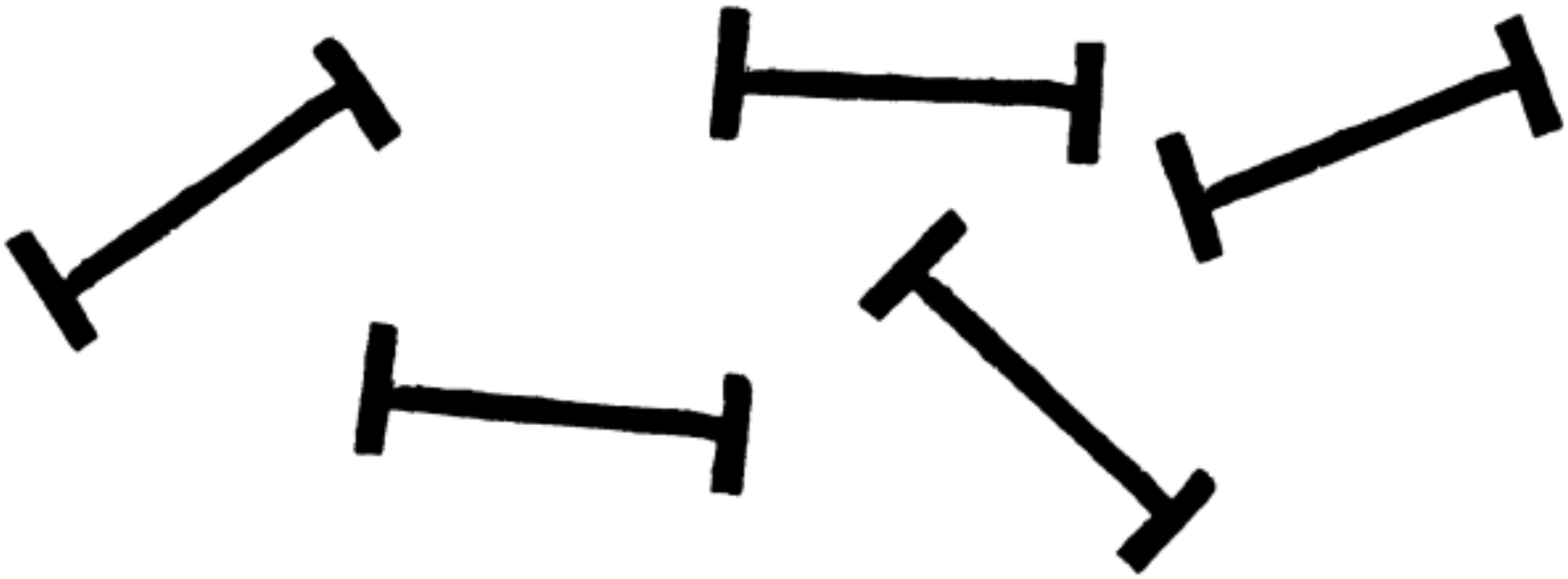
Slijedi nekoliko ilustracija iz Handbooka.

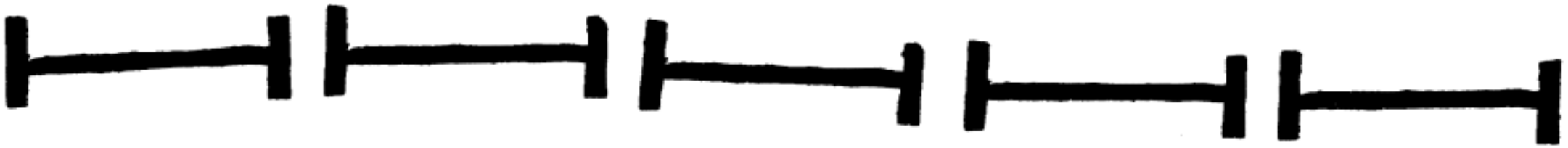
A HANDBOOK OF LISP FUNCTIONS

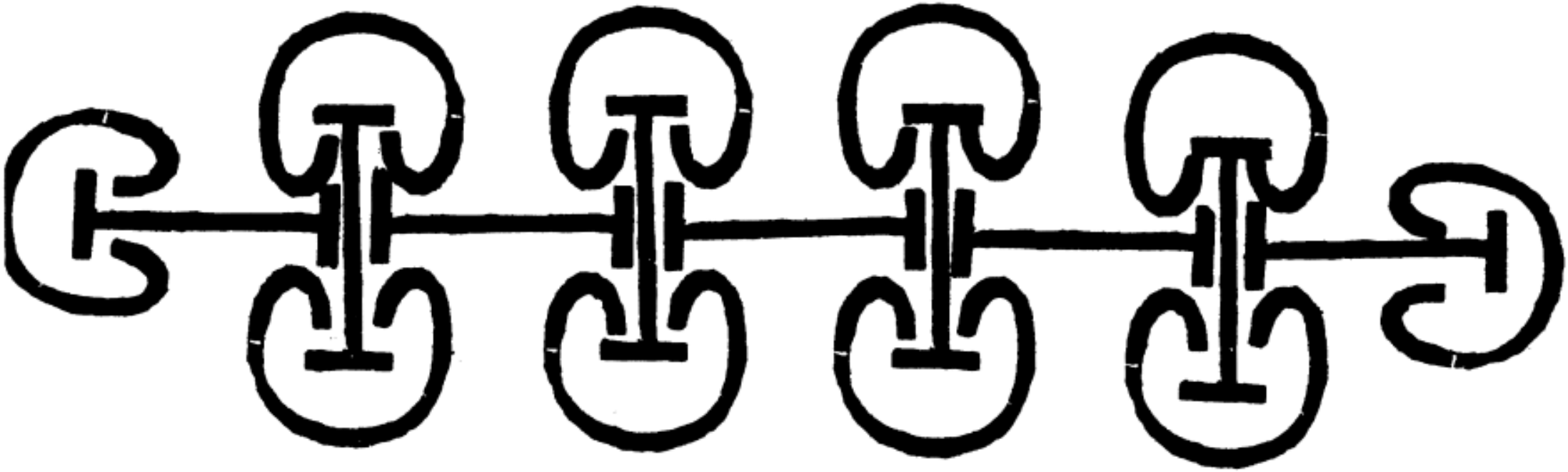
BIAS TECHNICAL REPORT 61-11

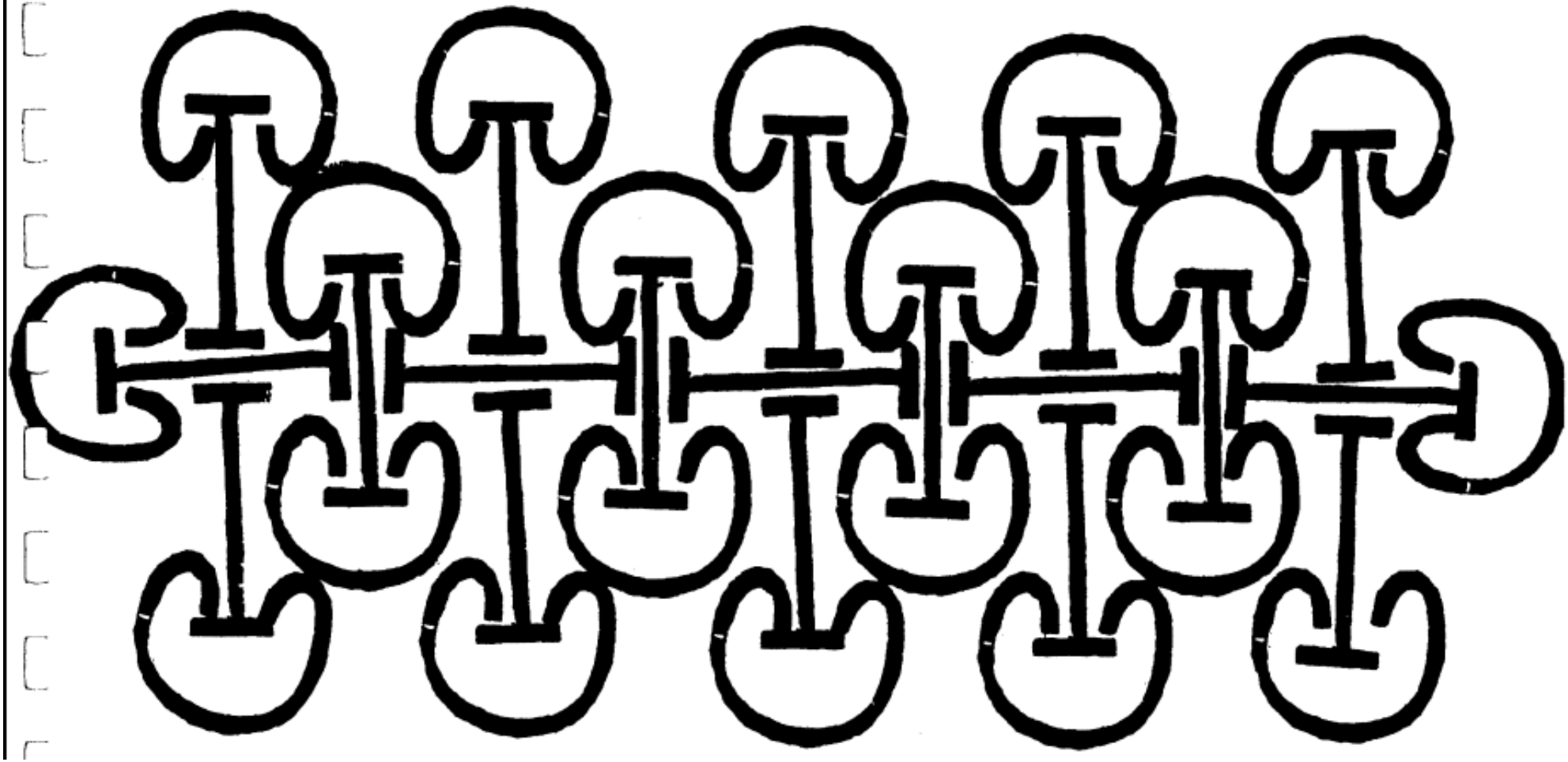

((A(B,C))(D,E))
CADR L
CAAR L
CADAR
E
D

The five primitive LISP functions









1962.

McCarthy nastavlja rad na „Math. theory of computation“.

McCarthy, Memo 33, *On efficient ways of evaluating certain recursive functions*.

McCarthy piše rekurzivnu funkciju koja računa Fibonaccijeve brojeve relativno brzo, tako što pamti sve međurezultate.

fibo[1]=1;

fibo[2]=1;

fibo[m]=fibo[m-1]+fibo[m-2]

Kako?

Koristi listu koja se naziva *known* u kojoj su parovi **(1,1)**,**(2,1)**,**(3,2)**,**(4,3)** itd. (obrnutim redom).

Definira funkcije

present[m;known] -- vraća **T** ili **F** ovisno je li **(**m**,**fm**)** u knownu.

fibo[m;known] -- vraća fm tražeći ga po knownu.

prob[m;known] -- koristeći već zadani known, vraća sličnu takvu listu, ali veću, sve do m. **((**m**,**fm**),** ..., **(3,2),(2,1),(1,1))**

Uobičajeno bi bilo:

```
prob[m;known]=
    [   present[m;known] ⟶ known
        T → append[   ((m, [ m=1 → 1;
                             m=2 → 1;
                             T →  fibo[m-1;prob[m-1;known]] +
                                  fibo[m-2;prob[m-2;known]] ] ))
                      prob[m-1,known]]]
```

Umjesto kritičnog djela McCarthy „faktorizira“.

```
lambda[[v];  fibo[m-1;v]+fibo[m-2;v]]
          [prob[m-1;known]]
```

I onda još jednom to isto (uočite: funkcionalno programiranje).

Uočite: funkcionalno programiranje; nema pridruživanja.

Problem for the student:

Write a general procedure that will transform any LISP recursive calculation into one that is guaranteed to evaluate the function no more than once for any argument. (10 points)

Prove by recursion induction or otherwise that the new function is always equivalent to the old. (50 points)

Michie, Donald, "Memo Functions and Machine Learning, Nature 1968.