

Contents

1 Building Abstractions with Procedures 1

A Notes on LISP 2

    A.1 Recursive Functions of Symbolic Expressions and Their Computation by Machine, McCarthy  
        1960 . . . . . 2

    A.1.1 Introduction . . . . . 2

    A.1.2 Functions and Function Definitions . . . . . 2

Index 4

# 1 Building Abstractions with Procedures

**Definition (computational process)**

Abstract beings that inhabit computers.

**Definition (data)**

Computational processes manipulate other abstract things called ***data*** as they evolve.

**Definition (program)**

A pattern of rules by which the evolution of a computational process is directed.

**Definition (programming language)**

That in which programs are carefully composed from symbolic expressions that prescribe the tasks we want our computational processes to perform.

**Definition (bug, glitch)**

Small errors.

**Definition (debug)**

Remove bugs.

## Programming in Lisp

See the [appendix](#).

## Appendix

### A Notes on LISP

#### A.1 *Recursive Functions of Symbolic Expressions and Their Computation by Machine*, McCarthy 1960

[Link to article.](#)

##### A.1.1 Introduction

LISP:

- "LIS<sup>t</sup> Processor"
- Developed for the IBM 704 computer by the Artificial Intelligence group at M.I.T.
- Designed to facilitate experiments with a proposed system called the *Advice Tracker*:
  - A programming system for manipulating expressions representing formalized declarative and imperative sentences so that the Advice Tracker system could make deductions.
  - Originally proposed in November 1958.
- Came to be based on a scheme for representing the partial recursive functions of a certain class of symbolic expressions, independent of any electronic computer.

In this article:

1. Describe a formalism for defining functions recursively.
2. Describe S-expressions and S-functions, give examples, and describe the universal S-function **apply** which plays the theoretical role of a universal Turing machine and the practical role of an interpreter.
3. Describe the representation of S-expressions in the memory of the IBM 704 by list structures ... and the representation of S-functions by program.
4. Mention the main features of the LISP programming system for the IMB 704.
5. Another way of describing computations with symbolic expressions.
6. Give a recursive function interpretation of flow charts.

##### A.1.2 Functions and Function Definitions

###### **Definition (partial function)**

A function that is defined only on part of its domain.

###### **Definition (propositional expression)**

A *propositional expression* is an expression whose possible values are *T* (for truth) and *F* (for falsity)

###### **Definition (predicate)**

A function whose range consists of the truth values *T* and *F*.

**Definition (conditional expression)**

A device for expressing the dependence of quantities on propositional quantities, denoted:

$$(p_1 \rightarrow e_1, \dots, p_n \rightarrow e_n)$$

where the  $p$ 's are propositional expressions and the  $e$ 's are expression of any kind, read "If  $p_1$  then  $e_1$ , otherwise if  $p_2$  then  $e_2$ , ... otherwise if  $p_n$  then  $e_n$ " or " $p_1$  yields  $e_1$ , ...,  $p_n$  yields  $e_n$ ."

## Index

bug, [1](#)

computational process, [1](#)

conditional expression, [3](#)

data, [1](#)

debug, [1](#)

glitch, [1](#)

partial function, [2](#)

predicate, [2](#)

program, [1](#)

programming language, [1](#)

propositional expression, [2](#)