Zuse's Plankalkül

Konrad Zuse built a series of complex and sophisicated computers from electromechanical relays. They were bombed but his Z4 was saved. He moved and worked on a programming language for it alone in a remote Bavarian village.

He wrote the language Plankalkül which translates to *program calculus*. The language was developed in relations to his Ph.D. dissertation. It is dated to 1945 but was published in 1972.

Remarkably complete

The Plankalkül language was very complete. Its most advanced features are in the area of data structures.
The language's simplest data type is the bit, Integer and floating-point numerics types were built from the bit type.

> *The floating-point type used twos-complement notation and the "hidden bit"*
> *scheme currently used to avoid storing the most significant bit of the normalized fraction part of a floating-point value.*

It also had arrays and records (records are called structs in C-based languages). Records could include nested records.

It had no explicit *goto* but it inclided iterative statements similar to ADA's *for*, and the *Fin* command which specified an exit out of a given number of iteration loop nestings.

It had *if statements* but not *else*.

Pseudo-code

Pseudo-code is used in a different sense here.

Short Code

Shortcode was developed for the **BINAC** computer, it was later transfered to a **UNIVAC I** it was the primary means of programming those machines.

```
01 -    06 abs value      1n (n+2)nd power
02 )    07 +              2n (n+2)nd root
03 =    08 pause          4n if <= n
04 /    09 (              58 print and tab
```

Available operations in Shortcode

Speedcoding

Speedcoding was developed by John Backus for the **IBM 701**.
Speedcoding interpreter effectively converted the 701 to a virtual three-address floating-point calculator.

Conditional and unconditional branches and input/output conversions were also part of the virtual architecture.

It included the novel facility to automatically increment address registers. This didn't appear in hardware before the UNIVAC 1107 computers of 1962.

UNIVAC "Compiling" system

Between 1951 and 1953, a team led by Grace Hopper at UNIVAC developed a series of "Compiling" systems named A-0, A-1 and A-2 that expanded a pseudo-code into machine code subprograms in the same way as macros are expanded into assembly language.

IBM 704 and Fortran

Design

The environment in which the Fortran was developed was as follows:

1. Computers had small memories and were slow and relatively unreliable.
2. The primary use of computers was for scientific computations.
3. There were no existing efficient and effective ways to program computers.
4. Because of the high cost of computers compared to the cost of programmers, speed of the generated object code was the primary goal of the first Fortran compilers.

Fortran I Overview

All of Fortran I's control statements were based on 704 instructions.
Fortran I has no data-typing statements. Variables I, J, K, L, M, N were implicitly integers. i, j, k where used by the science community and was therefore used L, M, N was added as additional variables.

Fortran halved the efficiency of the compiler compared to by hand.

Fortran II

Fortran II added independent compilation and fixed bugs.

Fortrans IV, 77, 90, 95, 2003, 2008

Fortran IV was widely used and standardized as Fortran 66.
Fortran 66 introduced explicit type declarations for variables, a logical if construct and capability of passing subprograms as parameters to other subprograms.

Later replaced by Fortran 77. 77 added character string handling, logical loop control statements and in with an optional else.

Fortran 90 added dynamic arrays, records, pointers, multiple selection statements, modules and recursion of subprograms. Fortran 90 recommended some features for removal for later versions

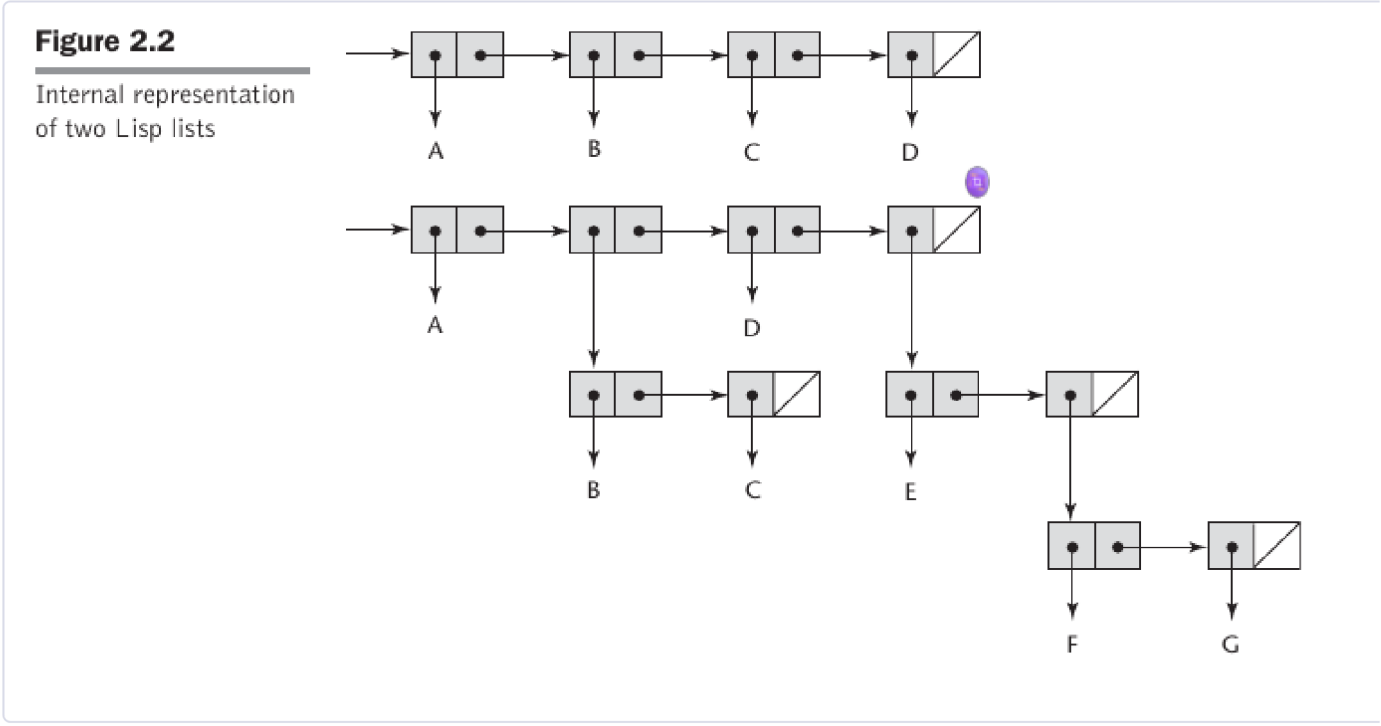
More on Fortran read page 67 in Sebesta.

Functional Programming: Lisp

Lisp is a functional language that was introduced in relation to *AI Development*. The earliest version was refered to as *Pure Lisp* as it was purely a functional language. It was developed as no other standard language existed for AI work. It was developed as an implementation of the program **Advice Taker**.

Data Structures

Pure Lisp consists of two data structures: *atoms* and *lists*. Atoms are symbols in the form of identifiers or numeric literals. Lists are denoted by parentheses and can be nested. Internally lists are stored as single-linked lists structures, in which each node has two pointers and represents a list element.



Internal representation of two Lisp lists

Syntax

Lisps syntax is exactly like the data. Parenthesized lists.

```
(A B C D)
```

When interpreted as data, it is a list of four elements. When viewed as code, it is the application of the function named A to the three parameters B, C, and D.

Descendants

Scheme and Common Lisp are two descendants of Lisp. Scheme is suitable for courses in functional programming and general introductions to programming while Common Lisp has a large number of data types and structures. Common Lisp was developed to create a more portable for the programs written.
Common Lisp is recognized as flexible.

Related languages

Other related languages are ML, Miranda and Caml.

The First Step Toward Sophistication: ALGOL 60

ALGOL 60 was created due to the plethora of different languages that existed for different machines. No single language was created as a standard for most machines. ACM (Association for Computing Machinery) formed a committee to study and recommend action to create a machine-independent scientific

programming language. Fortran could not be considered as it was solely owned by IBM.

In Germany, GAMM was on the same track and invited ACM to join. They together started the design and creation of ALGOL. It was firstly named IAL.

ALGOL was in many was a decendant of Fortran.

ALGOL took inspiration for the assignment statement from Plankalkül which used `=>`, but changed it to `:=` due to character set limitations.

AT THIS POINT PETER REALISED IT WAS NO LONGER WORTH IT READING. THIS HAS BEEN 2 HOURS OF READING WASTED :(