FOCAL (programming language)

From Wikipedia, the free encyclopedia

FOCAL, (abbreviation of **FO**rmula **CAL**culator), is an interpreted programming language resembling JOSS.

Largely the creation of Richard Merrill, FOCAL was initially written for and had its largest impact on the Digital Equipment Corporation's (DEC's) PDP-8 computers. Merrill wrote the original (1968) and classic FOCAL-69 interpreters for the PDP-8. Digital itself described FOCAL as "a JOSS-like language."

Like early versions of BASIC, FOCAL was a complete programming environment in itself, requiring no operating system. As in MUMPS, most commands could be, and in practice were, abbreviated to a single letter of the alphabet. Creative choices of words were used to make each command uniquely defined by its leading character. Digital made available several European-language versions in which the command words were translated into the target language.

Contents

Efficiency

FOCAL ran on very low-end PDP-8 systems, even systems with only 4K words of memory and lacking mass storage. The FOCAL interpreter was written in very tight assembly language and typically used only 3K 12-bit words, leaving a somewhat limiting 1K words to hold the user program, and variables.

If the system was upgraded by adding one or more extra 4K banks of memory, FOCAL could use that extra memory, either for a single user, or split the extra memory across several time-sharing users. FOCAL made extensive use of interrupt-driven terminal I/O, so it could keep four teleprinters busily whirring with nary a pause.

Memory space was too precious for storing textual error messages, so FOCAL used a clever work-around: error messages instead displayed the start address of the error-managing routine as a floating-point number. For example, in the FOCAL-69 implementation the division by zero error was managed by code starting at memory address octal 4333; division by zero at a program line numbered `01.10` produced the error message `?28.73 @ 01.10`, where `28.73` represents memory page 28 plus an offset of 73 words. Most FOCAL sites had an "error number to error message" listing taped up near the terminals.

Comparisons between FOCAL and BASIC were inevitable since both languages were common on smallish computers of the same era. FOCAL lacked inherent support for strings as data elements which could be assigned to variables. This is generally thought to be a serious deficiency as compared to the string capabilities in most BASICs. This deficiency, while serious, was not as utterly crippling as it might sound. A surprising amount of string usage in FOCAL programs is devoted to formatting user output. Since FOCAL output was character-stream-oriented, outputting two strings sequentially could sometimes substitute for concatenating them, and procedural tools could be written for performing complex formatted output.

A limited amount of string input could be done, so a program could ask simple Yes/No questions, but this was really a kludge. For example, if you typed "HELLO" at an input statement, FOCAL would convert the H to "8", then interpret the "E" as starting an exponent, then it would try to compute "8" to the "LLO" power, which would take several seconds of CPU time and result in a value of 0.76593020E+103, not a particularly helpful response.

It is generally agreed that FOCAL was more efficient in its use of resources than comparable BASIC systems. On a typical machine of the day (often with 6 to 24 kilobytes of core memory), FOCAL could handle larger and more complex programming tasks than BASIC.

FOCAL's PDP-8 implementation used a floating point representation that represented numbers as four 12-bit words, with thirty-six bits of mantissa and twelve bits of exponent. This allowed for both significantly higher precision and a significantly wider range of values than most other low-end programming systems, and made it a reasonable choice for serious numerical work. This high precision, and good choices for default decimal output formatting, meant that difficulties with binary-to-decimal rounding were not evident to beginning users.

Spinoffs

The Coca-Cola Corporation used a customized version of FOCAL called COKE.

FOCAL was later implemented on the PDP-7/9, PDP-11, PDP-12, PDP-5 and LINC-8.

The FOCAL manual showed how to add commands to the FOCAL parser, so many sites added specialized commands for operating custom hardware.

The Digital Equipment Computer Users' Society collected many patches and enhancements for FOCAL. There were even major enhanced offshoots of FOCAL, such as FOCAL-W, which added many features, including better mass storage file I/O and even virtual variable memory.

In Russia, it saw use as late as in early 1990s in mass-produced home computers of the Elektronika BK series.

Sample session with Focal on a PDP15

```
FOCAL15 V6B
*01.10 ASK "IN WHAT YEAR WERE YOU BORN?", YEAR
*01.20 SET YEAROFFOCAL=YEAR-1969+1
*01.30 IF (YEAROFFOCAL) 02.10,02.10,01.40
*01.40 TYPE "YOU WERE BORN IN THE YEAR ",YEAROFFOCAL," OF FOCAL!",!
*01.50 GOTO 01.10
*02.10 TYPE "YOU ARE TOO OLD FOR FOCAL, POPS",!
*02.20 GOTO 01.10
*GO
IN WHAT YEAR WERE YOU BORN?:1969
YOU WERE BORN IN THE YEAR     1.0000 OF FOCAL!
IN WHAT YEAR WERE YOU BORN?:1950
YOU ARE TOO OLD FOR FOCAL, POPS
IN WHAT YEAR WERE YOU BORN?:
```

This program takes your year of birth and calculates what year A.F. (after Focal) you were born in.

Program lines in a Focal program are grouped into linegroups and line numbers within that group. The first line of the program line 01.10 is line 10 of group 01. The line numbers are the targets of the **GOTO** and the **IF** statements.

The **ASK** statement prompts on the attached teleprinter for input, while the **TYPE** statement outputs text on the teleprinter. Multiple items can be output to the teleprinter by appending each item after a comma. An exclamation mark (!) causes a linefeed and carriage return to be sent.

The **SET** statement assigns a value to a variable. This value can be the result of an expression.

The conditional **IF** statement can receive up to three line numbers as parameters (so-called Arithmetic IF). The program branches to the first linenumber if the result of the expression in parentheses is less than zero, to the second if the result is zero and to the third if the result is above zero.

On the **GO** command, Focal begins to run the program.

Focal prompts with a single asterisk (*) at the beginning of the line when it is expecting input.

See also

- MUMPS programming language
- Non-English-based programming languages

External links

- DEC's FOCAL 1969 Promotional Booklet (http://www.cs.uiowa.edu/~jones/pdp8/focal/)
- The Computer History Simulation Project (Focal is available as a free download here) (http://simh.trailing-edge.com/)
- ftp://www.cozx.com/pub/langs/focal.tar.gz C-source version that runs under several operating systems including Linux
- the C-source for a modern DOS version suitable for teaching (http://www.catb.org/retro)
- Abramov V.A. Dialogue language FOCAL (Russian) (http://www.libex.ru/detail/book91959.html)

ISBN 5-06-001785-0
- Osetinsky L.G. FOCAL for mini-computers (Russian) (http://www.vntb.ru/kniga255.html) ISBN 5-217-00323-5

Categories: Domain-specific programming languages | JOSS programming language family

---