CS 365 — Programming Language Concepts

Introduction

Jan 13, 2014

Are programming languages languages?

In some universities, the systems used [to program computers] (BASIC, PASCAL, LISP, etc), metaphorically referred to as "programming languages," can now be substituted for the study of French, German, or Russian, the faster to make the students computer literate. These are not, of course, languages at all; they are coding systems.

Theodore Roszak, "Computers and Reason"

Are programming languages languages?

The typical person's understanding of the language he uses, however, is not so profound as to prevent him from labeling someting as a language that might resemble one only superficially. Thus, it is not surprising that the systems of notation which we use to communicate with our compuers came to be known as "programming languages." In fact, the very first programmer, Lady Lovelace, seems to have had the idea of a programming language as early as 1846...
In view of the venerable past of the programming language concept, it would be pedantic to attempt to demonstrate that programming languages are not "real" languages. Languages are what they say they are, and we ar perfectly entitiled to include systems of communication between man and computer under the same rubric as systems of communication between man and man or beast and beast.

Gerald Weinberg, The Psychology of Computer Programming

Abstraction

Programmers are more effective if shielded from, not exposed to, the innards of modules not their own.

Fred Brooks, MMM

The effective exploitation of his powers of abstraction must be regarded as one of the most vital activities of a competent programmer.

Edsgar Dijkstra, Turing talk (EWD 340)

Paradigms

Most books rigorously adhere to the sacred division of languages into "functional¡", "imperative", "object-oriented", and "logic" camps. I conjecture that this desire for taxonomy is an artifact of our science-envy from the early days of our discipline: a misguided attempt to follow the practice of science rather than its spirit. We are, however, a science of the artificial. What else to make of a language like Python, Ruby, or Perl? Their designers have no patience for the niceties of these Linnaean hierarchies; they borrow features as they wish, creating melanges that utterly defy characterization. How do we teachPL in this post-Linnaean era?

Shriram Krishnamurthi, "Teaching Programming Languages in a Post-Linnaean Age"