Author: William Z Chadwick
Date Created: 08/30/2022
Date Modified: 08/30/2022
Description: Research notes in response to prompt.

---

1. What are the four pillars of Object-Oriented Programming?

The four pillars of Object-Oriented Programming are 1st, Abstraction, 2nd, Encapsulation, 3rd, Inheritance, 4th, Polymorphism.

Abstraction is basically like reframing code in general terms, like the medieval or scientific concepts of *genera* and *species*, general and special, etc. The general is an abstraction, in this sense, which includes many special instances. Think of how Jazz is a *genre*, and Louis Armstrong and Mile Davis as Jazz players are both *specific instances* within that genre.

Due to my interest in languages, history, and how these engage with philosophical ideas, ethics and tech, and also due to my growing feeling that it was worthwhile to understand this idea of "abstraction," I recently picked up a book on Fortran titled, *Abstracting Away the Machine: The History of the Fortran Programming Language (FORmula TRANslation)*, written by Mark Jones Lorenzo.

While I think this "abstracting away" is not quite the same abstraction as the first pillar of Object Oriented Programming (OOP), it is similar in several respects, which I will discuss in my own words by drawing on the little of Lorenzo's book which I have read so far.

For one, Fortran allowed programmers to work with different machines without having to know the machine specific coding-language of each machine. To illustrate from what I understand, it had been the case before Fortran that a programmer working with three different machines would not be able to write one program for all three machines. Instead, she would have to write one program for machine a (in the corresponding language for a), another program for machine b (in the corresponding language for b), and yet a third program for machine c (in the corresponding language for c).

After the invention of Fortran, ideally at least, programmers could approach their task more efficiently. Instead of having to know and write working programs in the corresponding languages for machines a, b, and c, programmers only needed to know the general shape of the program (formula) which they needed to write in one language that could then be compiled (translated) into machine-readable code (which I think usually means some form of binary).

Because a program written in Fortran could be used with different machines as opposed to requiring that each machine had a unique program written just for its unique set-up, they called the Fortran coding language itself a "High Level Language." In a sense, this "abstracts away the machine" because a programmer could write a program at a comparatively general level (in an HLL) rather than how to write in, for instance, binary, or whatever very specific language (probably assembly language?) a

specific machine was built to have work in its very hardware (this would be a low-level coding language, the species under the genera).

Lorenzo writes,

“A high level computer language *abstracts away the machine* because the programmer need not be an expert in the machinations of computer hardware components, not understand the blizzard of zeros and ones that comprise a machine language, nor even juggle the mnemonics of an assembler to successfully program a computer. High-level languages (HLLs) automate, hide, or otherwise abstract away the underlying operations of the machine, allowing the programmer to focus on the coding problem at hand rather than having to worry about managing memory or conforming to the operations of a particular kind of processor. All of those details are hidden from view. Instead, programming in a form of natural language, rather than an unreadable machine code, becomes the order of the day, with low-level numeric coding giving way to high-level symbolic, algebraic-style manipulation.”

He lists various the HLLs, “ALGOL, BASIC, C/C++, COBOL, Java, LISP, LOGO, Pascal, PL/I, Python, and Visual Basic” and adds, “….[b]ut all HLLs owe a huge debt of gratitude to FORTRAN … which was the first mature high-level programming language to achieve widespread adoption.”

I am excited to read more.

---

2. What is your favorite thing you learned this week?

My favorite thing I've learned this week is stuff about abstractions. I’m learning a ton, and looking forward to learning more. My research above turned into a much longer blog post, which I posted on my blog, here:

https://perilousresonance.wordpress.com/2022/09/01/302-abstracting-away-the-universe-the-evolution-of-consciousness-the-ethics-of-care-and-the-dangers-of-abstraction/