

## Re: Any experience with "The Last One"?

**Source:** [http://coding.derkeiler.com/Archive/C\\_CPP/comp.lang.c/2003-11/4678.html](http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2003-11/4678.html)

---

**From:** Programmer Dude (*Chris\_at\_Sonnack.com*)

**Date:** 11/26/03

Date: Wed, 26 Nov 2003 13:39:24 -0600

Sidney Cadot wrote:

- > *I'd estimate it takes at least five years time for a talented*
- > *person to know computers well enough to program well in C.*

I'm forgetting the number, now, but there is a number of years that is considered the *\*minimum\** to learn *\*any\** complex skill truly well. It's something like five or seven or ten. That general area.

- > *The gentle (but not easy) path would probably be via a friendly*
- > *assembly language (PDP-11 used to be an excellent choice) to C,*
- > *preferably in an environment that allows some direct access to*
- > *hardware, with feedback.*

And which allows you to—as I used to say—go stomping through the operating system in metal boots. (—:

Which recalls a memory: a BOS error. (Big Orange Switch :—)

- > *The latter I think is important for people to get a real feel for*
- > *bits, bytes, and performance.*

I'd agree. As you mention later, knowing assembly made pointers so obvious it's hard sometimes to see why they confuse people. I'm not sure I'd go so far as to say assembly is *\*required\**, but I very much think it makes for better programmers.

- > *I think Pascal is a near-ideal language for learning structured*
- > *programming. The fact that it's strongly typed is a big plus I think.*

Certainly strong typing is a skill that should be understood. But there is considerable debate on the actual value of strong typing (I'll get back to this below).

- > *This is all to get background just to use C. This misses out on*
- > *functional programming (which is great to appreciate /elegance/ in*
- > *algorithms, if you can handle the abstraction)...*

comp.lang.c: Re: Any experience with "The Last One"?

It's quite an abstraction! I do agree FP is another required knowledge, but I don't think it's for novices (mathematicians, might take to it--most people, not so much).

> *...and of course OOP. For the latter, I wouldn't know what to recommend. Never got the hang of it myself.*

I love OOP. Smalltalk might be a good first language. Java is another good choice. Actually Java is probably better just in terms of coverage and available support (lots!). Smalltalk is one of those "better mousetrap" languages the world didn't beat a path to.

> *I think most C programmers that have this experience will approximately 'see' the assembly that will be generated for the C code, which can be valuable.*

And amusing! I've almost doubled over seeing the assembly that the compiler spit out sometimes (e.g. a jump to the very next instruction!). (Having written a couple (crude) compilers, I understand why this happens, but it's still pretty funny.)

>> *I've been mentoring a friend of mine through an adult education CS degree. His university uses Java as their teaching language, and after watching over his shoulder these few years, I think that is not a bad choice at all.*  
>  
> *They switched to Java at my university as well, but I wasn't too happy about it. The teaching was not very good, for one thing,...*

THAT'S a shame.

> *...but the serious trouble began when students lost sight of performance. They would just think of any Vector operation as  $O(1)$ , or have inner loops with object creation.*

I wonder, though. Learn the basics first and THEN you have the "language" to understand why and what to improve. It's like the old programming adage: 1. Make it Work. 2. Make it work Right. 3. Make it work Fast.

>> *If I had to teach a programming course again, I'd take a very close look at some of the newer languages: Dylan, Scheme, Ruby or Python. I've been hearing some interesting things about them...*  
>  
> *I've been doing some Python lately, and it's mostly marvelous.*  
> *The weird thing is the indenting;*

That gets a lot of comments. People tend to hate it or love it! I can see its attraction, but I can also see its dangers, and I think I lean towards more clear scope "definers" (although, please,

Re: Any experience with "The Last One"?

comp.lang.c: Re: Any experience with "The Last One"?

no "begin" and "end" :-).

- > *One minor drawback as a teaching aid (IMO, at least) is that it has*
- > *implicit variable typing (or, to be more precise, a variable is*
- > *just a binding of a name to a typed expression):*
- >
- > *a = 10*
- > *a = "hello"*
- >
- > *...is OK in Python.*

Python inherits a lot of concepts from Lisp, which is a language that also just binds a name to an expression. The above (in a different syntax, of course) would also work in Lisp.

There's been a raging debate in comp.lang.lisp about strong typing verse weak or no typing, and it's been interesting reading.

One argument is that type safety only eliminates one class of errors and this can lull the unwary into thinking a program is more correct than it is. I don't quite buy this, as I believe most experienced programmers know that a clean compile is no proof against all sorts of runtime errors. A clean compile ONLY guarantees static type safety.

Which can be nice, but can also prevent certain modes of work. One fellow reported a case where a large graphics library needed modifications that changed the types of some data. As the library was large and the changes widespread, there was a long time (days) when the library **COULD NOT BE COMPILED**. Thus could not be tested incrementally. When the compile was finally possible, it turned out that a large chunk of the work—the motivation for the change was "3. Make it work Fast"—was unnecessary. Incremental testing would have shown that.

Having dabbled in Lisp, the whole incremental testing thing really is rather nice. After listening to both sides of the debate, I find I have no real opinion wrt typed languages. Either are fine! (For that matter, I don't think I even have a preferred language anymore!)

```
--  
|_ CJSonack <Chris@Sonack.com> _____| How's my programming? |  
|_ http://www.Sonack.com/ _____| Call: 1-800-DEV-NULL |  
|_____|_____|
```