

# Re: Computer Algebra Algorithms

---

*Source:* <http://sci.tech-archive.net/Archive/sci.math.symbolic/2005-04/msg00024.html>

---

- *From:* Bernard Parisse <[parisse@xxxxxxxxxxxxxxxxxxxx](mailto:parisse@xxxxxxxxxxxxxxxxxxxx)>
  - *Date:* Fri, 08 Apr 2005 14:52:52 +0200
- 

I've been trying to follow this thread somewhat but there is many more posts here then I want to toughly dissect. I don't really think in the above posts enough motivation was given behind what you want to do. If you want to learn CAS, learn lisp because that is what the open source CAS are programmed in and a large part of the academic work in AI uses lisp.

Actually, there are C++ programmed CAS, like my `giac/xcas`  
[www-fourier.ujf-grenoble.fr/~parisse/giac.html](http://www-fourier.ujf-grenoble.fr/~parisse/giac.html)  
They are more flexible for programmers because they are available as C++ libraries, not only as a binary.

If you want to use C dynamically you are going to need some kind of parser. The parser could be written in C or any other language. It should return to you a tree structure representing the expression the user typed at the command line. You could then perform some algebra on it. In this case you may not care if you can overload the plus operator. You could just call it as a C function `pluss(a,b)`, since everything would be done by your custom interpreter (would this really be C?).

But the code will be more readable if you can write `a+b` in your source.

If you want to use C and you want to do something useful with CAS I would suggest learning the external function interface of MAPLE. This will allow you to Manipulate maple data structures in the Maple workspace with C functions. I havn't got into it but perhaps you could even use it to prevent the automatic simplification in Maple. Apparently the best reference for this is the Maple advanced programming guide. It is about 54 dollars.

You can do that with `giac` for free.

Re: Computer Algebra Algorithms