Subject: What does an optimal scientific programming language/environment need?
Posted by grunes on Fri, 19 Sep 2003 18:29:12 GMT
View Forum Message <> Reply to Message

I'm working on creating an optimal scientific programming language and environment. My hope is that people who use current environments have specific things they love about it, that need to be included. For now I'm trying to combine the best concepts from FORTRAN, BASIC, C, APL, IDL, PV-WAVE, and possibly MATLAB.

I have been an applications programmer in support of scientific research for 23 years, in radar and optical remote sensing, data compression, and statistical image processing, and have also produced operational software for ship-borne, airborne, and orbital platforms, so I have some ideas of my own. But I want other people's input before I begin, because I expect this to be a lot of work.

What I have in mind so far is:

1. Free or cheap software. Moderate price documentation ($20-50?).

2. Deliverable free and non-free executable products without special permission or royalty. Unfortunately, this seems to be the single most important criteria of modern scientific/engineering applications development, from the sponsor's point of view.

3. Very rapid developement and testing. Requires extreme conciseness, support of arrays, complex number, linear algebra, finite element and numerical integration of functions and differential equations, and little need for type and shape declarations. Can easily switch on automatic detection of subscript checking, memory reference checking, argument mismatches, fixed and floating point errors. FORTRAN style adjustable array bounds (e.g., a(-3:4, 5:7)).

4. Very rapid learning. Most of the language and environment must be summarizeable in a very few pages. Design must be consistent. Documentation unambiguous. For those who like lots of words, examples or homework problems, I could supply a larger tutorial manual.

5. 100% upwards compatibility with earlier versions of the same language/environment, and between platforms.

6. Close enough to standard mathematical notation to be mostly debuggable by inspection. But must have ASCII transliteration so people can use there own editors if they don't like mine.

7. Can add the declaration statments that make efficiency possible.

8. A compiled mode that really is as fast as FORTRAN or C, if you add those declarations. Compiler would produce 2nd level code for compilation by g77 and gcc.

9. The interpreter and Executables must be rock solid stable. That means very few heap dynamic memory allocation operations (malloc, calloc, alloc, new, allocate). Support dyanmic stack allocation, and a garbage collected heap area.

10. Screen and file graphics (plots, diagrams, images) must be very easy to produce. Default style products must be publication quality.

11. Can call and be called from FORTRAN and C programs.

12. Must support two primary development platforms at this time: Lintel and Wintel (PCs under Linux and Windows). (Nothing else is economically viable any more.) For the moment, I will ignore multiple CPU support, networking, and signal processing chips.

13. Be able to associate an area of virtual memory with an entire file. Should handle raw bytesream files, as well as TIFF scientific data sets. Since the indicated platforms only support 1-2 GB of user virtual memory space, I will have to create a special FILE type that allows 64 bit address manipulation.

14. Multi-precision calculations (e.g., 2, 4, 8, 16 byte and greater floating point numbers), and exact (rational) calculations must be easy to do.

15. No bugs. (I'll try.)

Please add your own criteria!

Subject: Re: What does an optimal scientific programming language/environment need?
Posted by Ken Plotkin on Sun, 21 Sep 2003 17:16:31 GMT

View Forum Message <> Reply to Message

On Sun, 21 Sep 2003 17:29:12 +0100, John Sullivan
<spamtrap@yddraiggoch.demon.co.uk> wrote:

> If you follow the link to the 920 programming languages on page
> http://wombat.doc.ic.ac.uk/foldoc/contents/language.html
> you will see that at least 5 languages are missing:
>   A, A+, J, K, Q-Nial

Make that at least six. I don't see COMAL listed, either.

Subject: Re: What does an optimal scientific programming language/environment need?
Posted by R.G. Stockwell on Sun, 21 Sep 2003 19:16:57 GMT

View Forum Message <> Reply to Message

"John Sullivan" <spamtrap@yddraiggoch.demon.co.uk> wrote in message
news:G8SUNoRYHdb$EAf2@yddraiggoch.demon.co.uk...

> In message <Dbjbb.20886$Od.764849@twister.tampabay.rr.com>, David Frank
> If you follow the link to the 920 programming languages on page
> http://wombat.doc.ic.ac.uk/foldoc/contents/language.html
> you will see that at least 5 languages are missing:
>   A, A+, J, K, Q-Nial
> --
> John Sullivan
> Please note that any disclaimer on email to me does not
> apply, because I have not agreed with it. If your lawyers
> disagree, please ask them to study the law of contract.

Also missing are IDL, PVWAVE, and MATLAB, which lack of mention
seems odd since those are the newsgroups to which this message is
crossposted.

bob

Subject: Re: What does an optimal scientific programming language/environment need?
Posted by phil chastney on Sun, 21 Sep 2003 19:56:01 GMT

View Forum Message <> Reply to Message

"R.G. Stockwell" <noemail@please.com> wrote in message
news:hQmbb.33$Rk2.40033@news.uswest.net...

>
> "John Sullivan" <spamtrap@yddraiggoch.demon.co.uk> wrote in message
> news:G8SUNoRYHdb$EAf2@yddraiggoch.demon.co.uk...
>> In message <Dbjbb.20886$Od.764849@twister.tampabay.rr.com>, David Frank
>> If you follow the link to the 920 programming languages on page
>> http://wombat.doc.ic.ac.uk/foldoc/contents/language.html
>> you will see that at least 5 languages are missing:
>>   A, A+, J, K, Q-Nial
>
> Also missing are IDL, PVWAVE, and MATLAB, which lack of mention
> seems odd since those are the newsgroups to which this message is
> crossposted.

SIMPL is also missing, which is a favourite of mine because it has been used 5
or 6 times, for completely different languages (which are not to be confused
with the current Linux initiative)

actually, I think I could name a couple of dozen other omissions . . . /phil

Subject: Re: What does an optimal scientific programming language/environment need?
Posted by Gary L. Scott on Sun, 21 Sep 2003 21:46:30 GMT
View Forum Message <> Reply to Message

Ken Plotkin wrote:
>
> On Sun, 21 Sep 2003 17:29:12 +0100, John Sullivan
> <spamtrap@yddraiggoch.demon.co.uk> wrote:
>
>> If you follow the link to the 920 programming languages on page
>> http://wombat.doc.ic.ac.uk/foldoc/contents/language.html
>> you will see that at least 5 languages are missing:
>>   A, A+, J, K, Q-Nial
>
> Make that at least six. I don't see COMAL listed, either.

Well, I've written several "macro processors". That's a pseudo
interpreter I would guess. Does that count or only those that were
borne of academia? Including those, I would guess there may be many
thousands of "languages".
--

Gary Scott
mailto:garyscott@ev1.net

Fortran Library
http://www.fortranlib.com

Support the GNU Fortran G95 Project: http://g95.sourceforge.net

Subject: Re: What does an optimal scientific programming language/environment need?
Posted by bjorn on Mon, 22 Sep 2003 10:26:20 GMT
View Forum Message <> Reply to Message

"phil chastney" <phil@KEEP.amadeus-info.SPAMFREE.freeserve.co.uk> wrote in message
news:<bkkvp6$bfh$1@news7.svr.pol.co.uk>...

>>> you will see that at least 5 languages are missing:
>>>   A, A+, J, K, Q-Nial
>>
>> Also missing are IDL, PVWAVE, and MATLAB, which lack of mention
>> seems odd since those are the newsgroups to which this message is
>> crossposted.
>
> SIMPL is also missing, which is a favourite of mine because it has been used 5

It would probably be more accurate to list progamming languages in
groups and subgroups rather than individually.

Quite aï¿½ few are just dialects of some major brand and do not
neccesarily need to be listed separately.

I am not sure you would end up with many groups, there would be a lot
of subgroups and then various numerous dialects within the subgroups.

It would of course be interesting to see the number of followers who
believe in different groups, subgroups and dialects.

It will take years to get people to agree on the groupings and where
individual programming dialects really belong.

Subject: Re: What does an optimal scientific programming language/environment need?
Posted by Paul Chapman on Mon, 22 Sep 2003 12:03:18 GMT

View Forum Message <> Reply to Message

>>> http://wombat.doc.ic.ac.uk/foldoc/contents/language.html

Also missing:
   CLOS
   CPL (although it is referenced under BCPL)
   Lua (http://www.lua.org/ - interesting!)
   Mathematica
   ML

Of course none of my languages appear, ;) including:

- A Minsky-register-machine macro-assembler (2001)
- Proprietary language Taxcel (1999)
- DE, which I designed for implementing I-APL (1986)
- Proprietary market-research language IRL (1976)
- Unnamed OO language (work in progress)

My guess is that historically there are thousands, if not tens of thousands,
of proprietary languages, probably many of which are still used in
production code years after their first implementation.

Additionally there must be thousands, if not tens of thousands, of languages
developed by hobby programming-language designers like me, which only got to
prototype stage.

Cheers, Paul

Subject: Re: What does an optimal scientific programming language/environment need?

Posted by Duane Bozarth on Mon, 22 Sep 2003 13:53:17 GMT

View Forum Message <> Reply to Message

grunes wrote:
>

After reading your wish list and replies to date, sounds like you're
describing Matlab redesigned/reimplemented with a seamless integration
to Mathematica based on present knowledge as opposed to its evolution
requiring it to keep compatability.

Subject: Re: What does an optimal scientific programming language/environment need?

Posted by phil chastney on Mon, 22 Sep 2003 15:19:13 GMT

View Forum Message <> Reply to Message

"grunes" <grunes@yahoo.com> wrote in message
news:2c0d6c85.0309191029.3efe3a99@posting.google.com...

>
> <heavy-handed snip>
>
> 6. Close enough to standard mathematical notation to be mostly
> debuggable by inspection. But must have ASCII transliteration so
> people can use there own editors if they don't like mine.

mmm -- I'm not at all sure I see math notation as fitting my idea of
"debuggable by inspection" -- in fact, math notation frequently requires
excessively many rules to disambiguate -- the one that blighted my early
introduction to statistics is "sigma x-squared", which is either the sum of the
squares, or the square of the sum, but to this day, I'm not sure which -- and
what is the value of 3**3**3 (or, if you prefer, 3^3^3)? -- there appears to
be no consensus on this one

> 7. Can add the declaration statements that make efficiency possible.

oh yes, three times yes !!!

declarations can show what type the arguments are expected to be -- if possible, the language should be constructed so as to allow static type-checking, as seen in functional languages -- and/or the routine should check the types of the actual arguments at runtime -- and while we're at it, it might as well check the type of the result

secondly, it may be possible to state the expected size and shape of the actual arguments and/or the result, which would permit more efficient allocation of memory -- even just knowing the _maximum_ size could still be useful

and finally, what about the invariant conditions, as defined in Z? -- testing these conditions can be a real pain (think about the invariants involved in inserting an element into a sorted list), but if such tests could be provided by the developer (or, better yet, generated automatically), then they could, if necessary, be turned off after testing is deemed complete

this is a rather broader view of declarations than that found in most languages, but it would surely be a significant advance?

oh well, dream on, I guess -- regards . . . /phil

Subject: Re: What does an optimal scientific programming language/environment need?
Posted by phil chastney on Mon, 22 Sep 2003 15:26:38 GMT

View Forum Message <> Reply to Message

"grunes" <grunes@yahoo.com> wrote in message
news:2c0d6c85.0309191029.3efe3a99@posting.google.com...

>
> 5. 100% upwards compatibility with earlier versions of the same
> language/environment, and between platforms.

I don't see this as either necessary or desirable -- Fortran has survived so well precisely because of its ability to move the language definition sideways, without being shackled to earlier imperfections

regards . . . /phil

Subject: Re: What does an optimal scientific programming language/environment need?
Posted by on Mon, 22 Sep 2003 17:37:13 GMT

View Forum Message <> Reply to Message

"Bjï¿½rn Helgason" <bjorn@hansaworld.is> wrote in message
> It would probably be more accurate to list progamming languages in
> groups and subgroups rather than individually.
>

There is a nice work at:

http://www.levenez.com/lang/

Best regards
Josï¿½ Rui

--
== ============
"Shrines! Shrines! Surely you don't believe in the gods. What's your
argument? Where's your proof?". Aristophanes
== ============
mailto://jrfsousaODOIBD@esoterica.pt/ Crawler baith. UnCaps me to reply.
Contact information: http://homepage.esoterica.pt/~jrfsousa/contact.html
== ============

Subject: Re: What does an optimal scientific programming language/environment need?

Posted by bjorn on Tue, 23 Sep 2003 11:11:48 GMT

View Forum Message <> Reply to Message

Josï¿½ Rui Faustino de Sousa <jrfsousaODOIBD@esoterica.pt> wrote in message
news:<bknc07$3m125$1@ID-134488.news.uni-berlin.de>...
> "Bjï¿½rn Helgason" <bjorn@hansaworld.is> wrote in message
>> It would probably be more accurate to list progamming languages in
>> groups and subgroups rather than individually.
>>
>
> There is a nice work at:
>
> http://www.levenez.com/lang/

That is pretty neat

http://www.levenez.com/lang/history.html#01

Thanks

Subject: Re: What does an optimal scientific programming language/environment need?

Posted by phil chastney on Thu, 25 Sep 2003 12:45:41 GMT

View Forum Message <> Reply to Message

"grunes" <grunes@yahoo.com> wrote in message
news:2c0d6c85.0309191029.3efe3a99@posting.google.com...

> I'm working on creating an optimal scientific programming language and
> environment. My hope is that people who use current environments have
> specific things they love about it, that need to be included. For now
> I'm trying to combine the best concepts from FORTRAN, BASIC, C, APL,
> IDL, PV-WAVE, and possibly MATLAB.
>
> I have been an applications programmer in support of scientific
> research for 23 years, in radar and optical remote sensing, data
> compression, and statistical image processing, and have also produced
> operational software for ship-borne, airborne, and orbital platforms,
> so I have some ideas of my own. But I want other people's input before
> I begin, because I expect this to be a lot of work.

signal processing and no FFT? some mistake, surely? . . . /phil

Subject: Re: What does an optimal scientific programming language/environment need?

Posted by grunes on Sun, 28 Sep 2003 02:11:18 GMT

View Forum Message <> Reply to Message

There are way too many responses to my original post to adequately
address them all. But a many of them are rather interesting.

===============

For example:

> and what is the value of 3**3**3 (or, if you prefer, 3^3^3)?
> -- there appears to be no consensus on this one

I assume you are using ** (FORTRAN) or ^ (BASIC) to represent
exponentiation (though ^ is more or less the standard math/logic
symbol for logical AND).

I thought there was consesus on x superscript y superscript z. It
means x raised to the (y raised to the z) power, just like FORTRAN
x**y**z means x**(y**z). The other meaning is already compactly
represented as x superscript yz, at least for positive reals.

I agree that a computer language notation must remove the ambiguities present in standard mathematical notation.

==============

I agree that signal processing should proably include FFT, and a few other commonly used transforms.

======================

At one point, I wanted to create my own IDL interpreter/compiler clone, and spent quite a lot of time figuring out how, but gave up after researching the complex legal status of "reverse engineering" in the USA.

In the mean time, I am following up on the suggestion one person made that "J", perhaps combined with calls to FORTRAN and C for what needs to be efficient, already answers the major needs. If that works well enough, fine.

Subject: Re: What does an optimal scientific programming language/environment need?
Posted by bv on Thu, 02 Oct 2003 01:01:07 GMT

View Forum Message <> Reply to Message

grunes wrote:
>
> I'm working on creating an optimal scientific programming language and
> environment. My hope is that people who use current environments have
> specific things they love about it, that need to be included. For now
> I'm trying to combine the best concepts from FORTRAN, BASIC, C, APL,
> IDL, PV-WAVE, and possibly MATLAB.

Before you embark on what is bound to be a long and winding road you might want to consider a recent quote by "DB" from sci.math.num-analysis ng which would invariably apply to whatever you might come up with.

"To get any chance of succeeding new programming languages should from the beginning provide a huge advantage compensating the loss of decades of expertises contained in the already available libraries, in the trained people, as well as in the compiler technology. Now to make the situation worse, the many functional languages compete with each others."

--
Dr.B.Voh

Applied Algorithms   http://sdynamix.com

Subject: Re: What does an optimal scientific programming language/environment need?
Posted by donotreply on Fri, 03 Oct 2003 00:08:57 GMT
View Forum Message <> Reply to Message

In article <k6Meb.443$ye2.217564282@newssvr11.news.prodigy.com>,
unixmonster@hotmail.com says...

```
>
>
> bv wrote:
>
>>  grunes wrote:
>>>
>>>  I'm working on creating an optimal scientific programming language and
>>>  environment. My hope is that people who use current environments have
>>>  specific things they love about it, that need to be included. For now
>>>  I'm trying to combine the best concepts from FORTRAN, BASIC, C, APL,
>>>  IDL, PV-WAVE, and possibly MATLAB.
>>
>>  Before you embark on what is bound to be a long and winding road you
>>  might want to consider a recent quote by "DB" from sci.math.num-analysis
>>  ng which would invariably apply to whatever you might come up with.
>>
>>  "To get any chance of succeeding new programming languages should
>>  from the beginning provide a huge advantage compensating the loss of
>>  decades of expertises contained in the already available libraries, in
>>  the trained people, as well as in the compiler technology.  Now to make
>>  the situation worse, the many functional languages compete with each
>>  others."
>>
>>  --
>>  Dr.B.Voh
>>  -----------------------------------------------------
>>  Applied Algorithms    http://sdynamix.com
>
> I would prefer to see APL extended with operator overloading and with
> defined primitive numeric types - so that one could model things like
> Grassmann algebras, moving frames etc. and maintain the concise syntax.
>
> I see little point in inventing another syntax.
>
> The most useful math machine I have is my TI-92+, because I can take it
> anywhere and it has a "good enough" symbol manipulation capability. I use
> it mainly for doing calculations in 6-d space. The syntax is based on
> "Derive" and I find it quite acceptable.
```

DERIVE has been my favorite computer algebra and ad-hoc calculation
language for a long time. The fact that it is now sold by Texas Instruments
through their education department belies its power. It's LISP-based

(although the LISP is almost entirely hidden) and, in its current incarnation, quite programmable. However, there's little that's procedural about its programming (not unexpected, given its LISP roots); instead, one writes a number of functions that reference each other.

DERIVE and its ancestor, MuMath, has actually been around for a LONG time -- IIRC, since the late 1970s. By the standards of most anything found in the computer world, it's remarkably bug-free. It also allows symbolic results to be output in Fortran syntax.

Highly recommended.