CoCoA-5 - Bug #443

lambda keyword

19 Feb 2014 21:31 - John Abbott

Status:	Closed	**Start date:**	19 Feb 2014
Priority:	High	**Due date:**	
Assignee:	John Abbott	**% Done:**	100%
Category:	Renaming	**Estimated time:**	5.50 hours
Target version:	CoCoA-5.0.9	**Spent time:**	5.50 hours

Description

I've just resuscitated the eigenvector code.

Having lambda as a keyword is a nuisance, *e.g.* it cannot be used as the name of an indeterminate!

I suggest changing the keyword into LambdaFn

Comments?

2014-03: final decision was for **Func...EndFunc**

Related issues:

Related to CoCoA-5 - Support #488: CoCoAManual: Help page for porting old C4 ...	**Closed**	**21 Mar 2014**

History

#1 - 20 Feb 2014 10:36 - John Abbott

Other suggestions for the keyword: **func...EndFunc**, **AnonFn...EndAnonFn**, **AnonFunc...EndAnonFunc**

My first suggestion was **LambdaFn...EndLambdaFn**

Why "lambda"? For those who don't know lambda calculus it's not obvious what "lambda" could mean.

#2 - 20 Feb 2014 15:34 - John Abbott

- Category changed from CoCoA-5 function: new to Renaming

- Target version set to CoCoA-5.0.9

- % Done changed from 0 to 10

Here is an example taken from NotBuiltin.cpkg5

Current form:

```
SortBy(ref L, Lambda(A,B) Return A.C>B.C; EndLambda);
```

One possibility (a bit longer than the current form)

```
SortBy(ref L, LambdaFn(A,B) Return A.C>B.C; EndLambdaFn);
```

Another possibility (a bit shorter than the current form)

```
SortBy(ref L, func(A,B) Return A.C>B.C; EndFunc);
```

Now I've seen an example, I like func a bit less because func(A,B) looks a lot like a fn call.

Other possibilities: DefFn, DefFunc, DefAnon, DefAnonFn, Defun...

```
SortBy(ref L, DefFn(A,B) Return A.C>B.C; EndDefFn);
```

#3 - 21 Feb 2014 13:16 - John Abbott

I've spoken to Anna and she has accepted that having lambda as a keyword in CoCoA-5 is inconvenient. The question is what to replace it with?

(A) a new keyword, maintaining the current syntax;
(B) a new syntax (*e.g.* some languages allow (x,y) |-> x^2+y^2).

Our current syntax fits the CoCoA "philosophy" of bracketing like this KeyXYZ...EndKeyXYZ. It is a bit cumbersome, but is conveniently general.

I quite like the "right arrow" syntax: I think it is fairly "natural" (and CoCoA does boast about its "natural" way of expressing maths). However it is not clear how the formula on the RHS is delineated (*i.e.* there is no explicit end marker), and it's not clear how it could be extended to multi-command fns.

Comments? Criticisms? Ideas?

#4 - 21 Feb 2014 15:20 - Anna Maria Bigatti

I prefer a name intstead of the arrow.
So I looked at synonyms: (mind you: I'm joking)
X --> definitely not good ;-)
nameless
SuchAndSuch
unnamed
whatsit
whatchamacallit --> a bit long
whatsitsname
YouKnowWho

Seriously I suggest (nice and short): NoName, DefFn

#5 - 21 Feb 2014 15:50 - John Abbott

Over lunch I made a survey (sample size = 1): Oscar had no idea what a "lambda function" might be (nor indeed what is meant by "anonymous function"). I feel this supports my feeling that there is no need to retain the word "lambda".

I looked at the Wikipedia page about anonymous functions. There are all sort of different syntaxes for anon fns (and many do use "lambda" in some way).

My favourite is still func...EndFunc; it is compact, and I believe reasonably clear. Slight variants are fun...EndFun (but the End of Fun doesn't sound much fun though), and function...EndFunction (very clear, but also quite long).

Here's how the last one might look in use:

```
SortBy(ref L, function(A,B) Return A.C>B.C; EndFunction);
```

#6 - 21 Feb 2014 16:01 - Anna Maria Bigatti

John Abbott wrote:

> Over lunch I made a survey (sample size = 1): ~~Oscar-Anonymous had no idea ~~;)
>
> My favourite is still func...EndFunc; it is compact, and I believe reasonably clear.

That's my favourite too. (I had understood you didn't like it)
Shorter still (too short?) is fn ... endfn

> function...EndFunction (very clear, but also quite long).

this could indeed be confusing: I think that's used in MatLab for normal function definition. Actually I'm assuming that because our **italian** students call function definitions "fare una /function/"

[20140222] it seems that Matlab uses the keyword function a bit like the keyword define in CoCoA-5; personally I do not regard "compatibility with Matlab" as a high priority (though perhaps we shouldn't be gratuitously incompatible?)

#7 - 21 Feb 2014 16:04 - John Abbott

Here's yet another idea (for simple anon fns), I suppose inspired by TeX.

An anon fn is any expression containing at least one identifier of the form **#n** where n is to be interpreted as being a positive integer literal (meaning the n-th argument. So the example I used above would become:

```
SortBy(ref L, (#1.C>#2.C));
```

I've added extra brackets (for clarity?).

This is undeniably compact. It is a bit opaque until you understand what #n means, but then I think it becomes quite readable.

I can think of 3 weak points:

- you cannot write an anon fn that does not use its last arg (assuming the arity is deduced from the highest numbered #n identifier -- LaTeX resolves this by requiring that the arity be specified at the start)
- you cannot give names to the args (this could be resolved by allowing the identifier to have a suffix which is ignored for semantic purposes: *e.g.* elim(#2indet, #1ideal) is equivalent to elim(#2,#1))
- it could be tricky to write an anon fn which defines another anon fn inside itself (two what does #1 refer inside the nested anon fn?)

[20140222] as presented here this idea is still rather "half baked" (*e.g.* because it is not clear how much "expression" belongs to the fn, how high up the expression tree should it go?) We need something like #2->#1+#2 where the lhs of the arrow indicates how many params (and also which part of the expr is in the fn defn)

#8 - 21 Feb 2014 16:07 - John Abbott

I forgot to add that if we have a very compact syntax for simple anon fns (single expr) then it is less important to have a compact syntax for longer anon fns.

#9 - 21 Feb 2014 16:18 - John Abbott

I found this neat little example in a file I was playing with:

```
fn:(x,y) |-> x+y
```

It is surely not as compact as #1+#2 or even #1x+#2y but is perhaps more immediately comprehensible.

[20140222] I wonder how hard it would be to impl this syntax, namely **fn:** *<arglist>* **|->** *<expr>* Any comments Giovanni?

#10 - 21 Feb 2014 16:27 - Anna Maria Bigatti

I still vote for (with the great advantage of being trivial to implement ;-)

```
SortBy(ref L, Fn(A,B) Return A.C>B.C; EndFn);
```

#11 - 22 Feb 2014 10:22 - John Abbott

Being "trivial to implement" is surely a **big advantage**!

I wonder whether fn...EndFn is too short; specifically do we really want fn (and Fn, FN & fN) all to be reserved words?

I think I might like to use fn (and perhaps Fn) as a variable name occasionally.

I have a slight preference for func...EndFunc simply because it seems less likely that I'd want to use func as a variable name.

#12 - 22 Feb 2014 11:40 - John Abbott

Here are the steps to make once we have decided what to do:
Step 1: modify interpreter (ought to be very easy, will break C5)
Step 2: modify packages and C5 test files (should work again)
Step 3: modify emacs macro defns to recognise new keyword
Step 4: update documentation

#13 - 24 Feb 2014 17:41 - John Abbott

- Status changed from New to Feedback

- Assignee set to John Abbott

- % Done changed from 10 to 90

We have chosen **Func...EndFunc** as the best compromise.

Implemented all steps (1)-(4).

Status is now **feedback**

#14 - 25 Mar 2014 14:54 - John Abbott

- Status changed from Feedback to Closed

- % Done changed from 90 to 100

I have left the name "lambda" in the code for lexer/parser/interpreter since anyone playing with that code should know what is meant by a "lambda fn".

In all other places the word "lambda" has been eliminated (except as a keyword for the manual entry).

Closing after 1 troublefree month in feedback.

#15 - 02 Apr 2014 18:35 - Anna Maria Bigatti

- Estimated time set to 5.50 h