# CoCoA-5 - Support #692

## Local variables hiding out ones at top level

04 May 2015 15:11 - Anna Maria Bigatti

| | | | |
|---|---|---|---|
| **Status:** | Rejected | **Start date:** | 04 May 2015 |
| **Priority:** | Normal | **Due date:** | |
| **Assignee:** | John Abbott | **% Done:** | 100% |
| **Category:** | Parser/Interpreter | **Estimated time:** | 0.99 hour |
| **Target version:** | CoCoA-5.4.0 | **Spent time:** | 1.00 hour |

**Description**

In function definitions local variables hiding outer ones are signalled by a Warning.
At top level not. This might be confusing!
(... when someone has the bad habit of calling all variables x ;-)

```
/**/ Use QQ[x,y,z];

/**/ for x:=1 to 6 do print x+1 ; endfor;
234567

/**/ [ x+5 | x in 1..6];
[6, 7, 8, 9, 10, 11]
```

**Related issues:**

| | | |
|---|---|---|
| Related to CoCoA-5 - Bug #486: Warning about variable hiding another -- too z... | **New** | **20 Mar 2014** |

### History

#### #1 - 04 May 2015 15:43 - John Abbott

In the example given I note that x is actually a **protected** variable -- perhaps even more reason for a warning?

#### #2 - 02 Mar 2020 22:32 - John Abbott

*- Target version changed from CoCoA-5.?.? to CoCoA-5.4.0*

I'm now quite undecided about this.

Every so often I do get a warning about one variable hiding another, but it is almost always a nuisance rather than a help. To eliminate the warning I must then change the name of a variable, but that can be "annoying".

Here is an artificial example:

```
define fn(L)
  for j:=1 to len(L) do something; endfor;
  j := 1; while not(IsGood(L[j]) do incr(ref j); endwhile;
  // now use index j ...
enddefine;
```

#### #3 - 23 Sep 2020 11:06 - John Abbott

*- Status changed from New to In Progress*

*- % Done changed from 0 to 10*

I think I now prefer not to issue a warning in the examples given here.
If CoCoA issues too many warnings then the user may end up simply ignoring them.
As hinted in the comment above, I might even be tempted to remove the warnings which are currently given inside a user defined function when a local variable is also used as a loop variable. Here is another example of an annoying warning:

```
define john(n)
  L := [i in 1..n | IsPrime(i)];
  i := 1;
  while i <= len(L) do
    if IsPrime(L[i]+2) then return i; endif;
  endwhile;
  return 0;
enddefine;
```

The main point here is that he loop variable i in the first line of the function appears before the "scope" of the local variable i begins. However, CoCoA extends the "scope" of the local variable to the whole function... so a warning is issued.
They do say that using the same variable for different purposes inside a single function is poor style... :-/

#### #4 - 02 Oct 2020 11:43 - John Abbott

*- Status changed from In Progress to Rejected*

*- Assignee set to John Abbott*

*- % Done changed from 10 to 100*

*- Estimated time set to 0.99 h*

After a Skype discussion we have decided to leave things as they are: while not perfect (and sometimes annoying) the warnings may be a useful guide to programmers with not so much experience.