# CoCoA-5 - Bug #669

## NUL char in input terminates CoCoA-5

06 Mar 2015 17:26 - John Abbott

| | | | |
|---|---|---|---|
| **Status:** | New | **Start date:** | 06 Mar 2015 |
| **Priority:** | Low | **Due date:** | |
| **Assignee:** | | **% Done:** | 0% |
| **Category:** | Parser/Interpreter | **Estimated time:** | 0.00 hour |
| **Target version:** | CoCoA-5.?.? | **Spent time:** | 0.55 hour |

**Description**

I tried a "fuzzing" test with CoCoA-5 (feeding the executable as input).
CoCoA-5 treats a NUL (ASCII code 0) as end-of-input; do we want this?

Note that CoCoA-5 treats EOT (ASCII 04) as end-of-input.

### History

**#1 - 02 Mar 2020 20:37 - John Abbott**

Maybe parser::tryToRecover is where one needs to look?

**#2 - 04 Mar 2020 21:19 - John Abbott**

A couple of days ago I fixed a similar bug in the reading of string literals.
The problem was that the code checked that the character read was NUL, and took that to mean end-of-input, but it was also possible to check the value of a CharPointer instead -- cleaner, and it allows NULs inside string literals (OK, perhaps not so useful).

**#3 - 04 Mar 2020 21:28 - John Abbott**

The relevant source code is mostly likely in Lexer.C around lines 136--137.
Inside Lexer::getToken there is a big switch statement which explicitly tests for **'\0'**, and returns Token::EndOfFile in that case.

I could just try commenting it out to see what happens... could be risky!

If it is commented out then NUL would simply trigger an "Unknown symbol" exception (if nothing worse occurs).