CoCoALib - Design #1538

RingElem from string (ReadExpr)

13 Nov 2020 14:59 - John Abbott

Status:	Closed	**Start date:**	13 Nov 2020
Priority:	Normal	**Due date:**	
Assignee:	John Abbott	**% Done:**	100%
Category:	Various	**Estimated time:**	1.61 hour
Target version:	CoCoALib-0.99800	**Spent time:**	1.70 hour

Description

What should the following do

```
RingElem(QQ, "1/-2");
```

Currently it gives error "unexpected minus".

Related issues:

Related to CoCoALib - Design #1529: INPUT questions	**Closed**	**31 Oct 2020**
Related to CoCoA-5 - Bug #1537: EmacUI: strange colours, sometimes	**Closed**	**13 Nov 2020**
Related to CoCoA-5 - Design #1540: Double power	**Closed**	**16 Nov 2020**
Related to CoCoALib - Bug #1579: Readexpr/RingElem: unhelpful error message w...	**Closed**	**23 Feb 2021**
Related to CoCoALib - Slug #1518: SLUG: Printing PPs with many indets	**In Progress**	**23 Oct 2020**

History

#1 - 13 Nov 2020 14:59 - John Abbott

- Related to Design #1529: INPUT questions added

#2 - 13 Nov 2020 15:04 - John Abbott

Similar problem for **RingElem(QQ, "2*-3");**

#3 - 13 Nov 2020 15:20 - John Abbott

My personal opinion is that I do not like **1/-2** and **2*-3**; they both look like typos to me. Perhaps the user meant **1/x-2** or **2*x-3**?
In contrast, with brackets I find it easier to read, and less likely that the user made a typo.

#4 - 17 Nov 2020 20:28 - John Abbott

- Related to Bug #1537: EmacUI: strange colours, sometimes added

#5 - 17 Nov 2020 22:53 - John Abbott

- Status changed from New to In Progress

- % Done changed from 0 to 10

The input functions are quite messy: not really a coherent design philosophy.

It seems that **ReadExpr** gives error if there are extra chars after the end of the expr, **but ReadExprSemicolon** does not complain if there are extra chars after the semicolon.
At the moment my expectation is that a fn which accepts a stream would read the next item on the stream, and leave the input pointer at the first char after the read value.
If the caller wants to ensure that there is nothing else in the stream then this must be written explicitly in the caller's code.

With this design **ReadExprSemicolon** becomes even simpler to implement.

Maybe there could be **ReadJust1Expr** which behaves as ReadExpr currently does?

#6 - 20 Nov 2020 11:39 - John Abbott

- Related to Design #1540: Double power added

#7 - 17 Feb 2021 11:16 - John Abbott

- Status changed from In Progress to Resolved

- Assignee set to John Abbott

- % Done changed from 10 to 70

I have just checked what the code does (or seems like it should do).
The current impl is:

- reading from istream keeps going until an "impossible" char is found; error if expr up to that point is incomplete, o/w no error & remaining chars are ready for further input operations
- reading from string expects the string to contain exactly the right input: trailing non-whitespace chars will trigger an error

#8 - 23 Feb 2021 10:23 - John Abbott

- % Done changed from 70 to 80

I have changed the CoCoA-5 parser so that it produces a warning when given input such as **2*-1** or **3/-1**. The computation still works, but the warning recomments putting the 2nd arg inside brackets. Also **2*+1** and **3/+1** produce the same warning.

An impromptu test suggests that all works as desired.

#9 - 23 Feb 2021 10:25 - John Abbott

Should we attempt to produce a better error message for the example given in the original description?
Is it worth it? Is that a good use of our limited resources?

#10 - 23 Feb 2021 10:49 - John Abbott

- Related to Bug #1579: Readexpr/RingElem: unhelpful error message when input is wrong added

#11 - 16 Sep 2021 12:54 - John Abbott

- Status changed from Resolved to Closed

- % Done changed from 80 to 100

- Estimated time set to 1.61 h

#12 - 12 May 2023 07:21 - Anna Maria Bigatti

- Related to Slug #1518: SLUG: Printing PPs with many indets added