CoCoA-5 - Feature #1509

RingElems with empty input

14 Oct 2020 11:52 - John Abbott

Status:	Closed	**Start date:**	14 Oct 2020
Priority:	Normal	**Due date:**	
Assignee:	Anna Maria Bigatti	**% Done:**	100%
Category:	bug	**Estimated time:**	3.01 hours
Target version:	CoCoA-5.4.0	**Spent time:**	3.00 hours

Description

In CoCoA-5 we have RingElems and RingElemList.
The second function can produce an empty list (from input "[]").
What should RingElems do when passed an empty string (or one full of whitespace)?

Currently it gives an unhelpful error mesg:

```
--> ERROR: Unexpected '\377'
--> [CoCoALib] ReadFactor
--> RingElems(R,"  ");
--> ^^^^^^^^^^^^^^^^^
```

Related issues:

Related to CoCoALib - Design #1391: RingElems: syntax with [and] ?	**Closed**	**10 Jan 2020**
Related to CoCoALib - Design #1523: Input fns: action when when istream is in...	**Closed**	**26 Oct 2020**

History

#1 - 14 Oct 2020 11:53 - John Abbott

- Related to Design #1391: RingElems: syntax with [and] ? added

#2 - 14 Oct 2020 14:10 - John Abbott

- Description updated

#3 - 14 Oct 2020 14:12 - Anna Maria Bigatti

- Description updated

- Assignee set to Anna Maria Bigatti

- % Done changed from 0 to 10

- Estimated time set to 2.00 h

we want

```
/**/  RingElems("     ");
[]
```

PS should also work for an empty string

Defined in RingElemInput.C

#4 - 14 Oct 2020 14:13 - John Abbott

- Description updated

#5 - 21 Oct 2020 13:56 - John Abbott

- Status changed from New to In Progress

- % Done changed from 10 to 20

We could add the following at the start of RingElems (in RingElemInput:C:278)

```
    std::vector<RingElem> v;
    // Skip initial whitespace; if we hit EOF, return empty vector
    in >> std::ws;
    if (in.eof() || !in) return v;
```

I have just tried this, and it seems to work.

#6 - 23 Oct 2020 09:33 - Anna Maria Bigatti

- Tracker changed from Bug to Feature

Fixed in RingElemInput.C, therefore also for CoCoALib.

#7 - 23 Oct 2020 09:59 - John Abbott

- % Done changed from 20 to 50

Ah, me too. I have just checked in my code... oddly, there were no clashes. What does that mean?

#8 - 26 Oct 2020 14:32 - John Abbott

- Related to Design #1523: Input fns: action when when istream is in bad state? added

#9 - 26 Oct 2020 19:39 - John Abbott

- Status changed from In Progress to Resolved

- % Done changed from 50 to 80

I have improved (??) the code. It should give better err mesgs now. Here are some examples:

```
/**/ RingElems(ZZ,",");
--> ERROR: Unexpected ','

/**/ RingElems(ZZ,"1,");
--> ERROR: Unexpected EOF

/**/ RingElems(ZZ,"*");
--> ERROR: Unexpected '*'

/**/ RingElems(ZZ,")");
```

```
--> ERROR: Unexpected ')'

/**/ RingElems(ZZ,"x");
--> ERROR: symbol not in ring
/**/
```

#10 - 29 Oct 2020 15:57 - Anna Maria Bigatti

John Abbott wrote:

> We could add the following at the start of RingElems (in RingElemInput:C:278)
> if (in.eof() || !in) return v;

You are allowing (i.e. return empty vector) a problematic in. Why? Isn't this dangerous?

#11 - 29 Oct 2020 22:03 - John Abbott

I do not think it is dangerous.

We cannot easily compare with C++ input via the >> operator since that reads into a variable which already has a value (and in the input stream is bad, the variable is left unchanged, I believe).

Note sure how relevant this is: C++ getline will return an empty string if applied to a stream which is bad (or already at EOF).

Since we offer a function test if a stream is at EOF (or bad?), we could simply make input operations from bad istreams into no-ops.
An alternative might be to throw exceptions, but exactly when? For the caller, it is probably simpler to check IsAtEOF before reading, rather than having to put the read into a try...catch block.

#12 - 30 Oct 2020 19:58 - John Abbott

- Status changed from Resolved to Feedback

- % Done changed from 80 to 90

I have also implemented RingElemVec (even though Anna did not want it).

I'll check in so we can test it.

#13 - 30 Oct 2020 20:20 - John Abbott

Checked in.

#14 - 08 Jan 2021 11:35 - Anna Maria Bigatti

- Status changed from Feedback to Closed

- % Done changed from 90 to 100

- Estimated time changed from 2.00 h to 3.01 h