Math 581d, Fall 2010, Homework 2

William Stein

October 6, 2010

Do the following, and turn them in by email (`wstein@gmail.com`) before the beginning of class on Wednesday, October 13, 2010.

1. For each C/C++ library or program in the 8-page list of C/C++ libraries that I handed out on Oct 4 (or see my blog post `http://sagemath.blogspot.com/2010/10/standalone-cc-libraries-that-are.html`), find some input to Sage that causes code in that C/C++ library (or program) to be executed. The more interesting the better. This is a potentially difficult assignment, if you start from nothing, so I'll walk you through a systematic way of doing one of these, and if you get stuck you can use this approach to all of the others. Let's find an example that uses the `polybori` library... whatever that is.

   (a) I search for polybori appearing in the Sage library source code:

```
flat:hw wstein$ sage
--------------------------------------------------------------------------------
| Sage Version 4.5.2, Release Date: 2010-08-05                                 |
| Type notebook() for the GUI, and license() for information.                  |
--------------------------------------------------------------------------------
sage: search_src('polybori')
crypto/mq/mpolynomialsystem.py:26:   sage: sr = mq.SR(2,1,2,4,gf2=True,polybori=Tru
crypto/mq/mpolynomialsystem.py:115:    sage: sr = mq.SR(1,1,1,4,gf2=True,polybori=T
crypto/mq/mpolynomialsystem.py:1236:            sage: sr = mq.SR(2,4,4,8,gf2=True,p
crypto/mq/mpolynomialsystem.py:1322:        from polybori.ll import ll_encode
crypto/mq/mpolynomialsystem.py:1323:        from polybori.ll import ll_red_nf_redsb
crypto/mq/sr.py:92:    sage: sr = mq.SR(1,1,1,4, gf2=True, polybori=True)
```

   (b) I look at one of the files that is mentioned:

```
sage: SAGE_ROOT
'/Users/wstein/sage/build/sage-4.5.2'
sage: quit
$ cd /Users/wstein/sage/build/sage-4.5.2/devel/sage/sage/
$ vi crypto/mq/mpolynomialsystem.py
$ sage
sage: sr = mq.SR(2,1,2,4,gf2=True,polybori=True)
```

```
SR(2,1,2,4)
sage: sr.polynomial_system()
...
```

(c) I'm honestly not sure whether the polybori library really got used at all above. Being nervous, I'll try harder, again using `search_src('polybori')` I find the file `rings/polynomial/pbori.pyx`. This one looks more promising, and I paste in this example:

```
sage: from polybori import BooleSet
sage: B.<a,b,c,d> = BooleanPolynomialRing(4)
sage: BS = BooleSet(a.set())
sage: BS
{{a}}
sage: type(B)
<type 'sage.rings.polynomial.pbori.BooleanPolynomialRing'>
```

That's got to be using polybori. I'll take my chances with the grader :-)

2. Track down the source code for the function in the mpfi (interval arithmetic) library for computing the cosine of an interval. How many lines of code is it? Is it well documented (in your opinion)?