PS2A

LINEAR FEEDBACK SHIFT REGISTER (PART A)

We will be completing the linear feedback shift register assignment described at http://www.cs.princeton.edu/courses/archive/fall13/cos126/assignments/lfsr.html.

For this portion of the assignment, you will:

- implement the LFSR class
- implement unit tests using the Boost test framework

DETAILS

https://www.ibm.com/developerworks/aix/library/au-ctools1_boost/index.html

https://theboostcpplibraries.com/

If you're working on Mac, you may install boost using homebrew.

- Per the Princeton assignment, implement the LFSR class, with the following methods:
  - constructor which accepts a C++ `String` of `1` and `0` characters, and a tap position;
  - `int step()` function of zero args which returns an `int` that will be a zero or a one;
  - `int generate(int k)` function that returns a *k*–bit integer;
  - instead of implementing the `toString` method, overload the `<<` stream insertion operator to display its current register value in printable form (see these instructions

http://www.learncpp.com/cpp-tutorial/93-overloading-the-io-operators/)

The implementation must be contained in files named `LFSR.cpp` and `LFSR.hpp`.

A note to give guidance on your internal representation: Your code must work with seed strings up to 32 bits long.

- Two additional unit tests in Boost, in a file `test.cpp`. Here is a starter file for your tests:

Your Makefile should have the targets `all`, `LFSR.o`, `test.o`, `ps2a`, and `clean`, and make sure all prerequisites are correct (e.g., `LFSR.o` should have `LFSR.cpp` and `LFSR.hpp` as prerequisites).

- Submit a `ps2a-readme.txt` file that includes:
- (1) your name,
- (2) an explanation of the representation you used for the register bits (how it works, and why you selected it), and
- (3) a discussion of what's being tested in your two additional Boost unit tests.
- Make sure all your files are in a directory named `ps2a`

SUBMIT INSTRUCTIONS

Archive and submit your source code files `test.cpp`, `LFSR.cpp`, and `LFSR.hpp` plus your `Makefile` and your `ps2a-readme.txt`. The executable that the Makefile builds must be called `ps2a`. If you additionally have a `main.cpp` file with some printf-style tests, you may include that too.

Submit using the submit utility as follows:

submit schakrab ps2a ps2a

GRADING RUBRIC

Core implementation: 4

(full & correct implementation=4 pts; nearly complete=3pts; part way=2 pts;

(included and covers material required)

Total: 10

JUST FOR FUN

Maybe you want to test that your LFSR actually goes through 2^k - 1 steps before recycling (where k is the length of the seed)?

You can use the `std::streamstring` class to get your current register value into a string; e.g.:

```
#include <sstream>
...
LFSR l("001", 1);
std::stringstream buffer;
buffer << l;

if (buffer.str().compare("001") == 0)
  std::cout << "yeah!\n";
else
  std::cout << "argh!!\n";
```

You can use this ability to keep stepping your LFSR and count how many steps it takes for the register to recycle back to the initial seed.