```
 1: #include "LFSR.hpp"
 2:
 3: #define BOOST_TEST_DYN_LINK
 4: #define BOOST_TEST_MODULE Main
 5: #include <boost/test/unit_test.hpp>
 6:
 7: BOOST_AUTO_TEST_CASE(fiveBitsTapAtTwo)
 8: {
 9:
10:   LFSR l("00111", 2);
11:   BOOST_REQUIRE(l.step() == 1);
12:   BOOST_REQUIRE(l.step() == 1);
13:   BOOST_REQUIRE(l.step() == 0);
14:   BOOST_REQUIRE(l.step() == 0);
15:   BOOST_REQUIRE(l.step() == 0);
16:   BOOST_REQUIRE(l.step() == 1);
17:   BOOST_REQUIRE(l.step() == 1);
18:   BOOST_REQUIRE(l.step() == 0);
19:
20:   LFSR l2("00111", 2);
21:   BOOST_REQUIRE(l2.generate(8) == 198);
22: }
```

```
 1: #ifndef LFSR_HPP
 2: #define LFSR_HPP
 3:
 4: #include <iostream>
 5: #include <string>
 6:
 7: class LFSR
 8: {
 9:
10: public:
11:   LFSR(std::string user_seed, int user_tap);
12:   ~LFSR();
13:   int step();
14:   int generate(int k);
15:
16:   friend std::ostream& operator<< (std::ostream &out, LFSR &lfsr);
17:
18: private:
19:   std::string seed;
20:   int tap;
21:
22: };
23:
24:
25: #endif
```

```
 1: #include "LFSR.hpp"
 2:
 3: LFSR::LFSR(std::string user_seed, int user_tap)
 4: {
 5:   tap = user_tap;
 6:   seed = user_seed;
 7: }
 8:
 9: LFSR::~LFSR()
10: {
11:
12: }
13:
14: int LFSR::step()
15: {
16:
17:   int bit;
18:   char c_bit;
19:
20:   if (seed.at(0) == seed.at((seed.size()-1) - tap))
21:     {
22:       bit = 0;
23:       c_bit = '0';
24:     }
25:   else
26:     {
27:       bit = 1;
28:       c_bit = '1';
29:     }
30:
31:   for(unsigned int i = 0; i < (seed.size()-1); i++)
32:     {
33:       seed.at(i) = seed.at(i+1);
34:     }
35:   seed.at(seed.size()-1) = c_bit;
36:
37:   return bit;
38: }
39:
40: int LFSR::generate(int k)
41: {
42:   int gen = 0;
43:
44:   for (int i =0; i < k; i++)
45:     {
46:       gen = gen*2 + LFSR::step();
47:     }
48:
49:   return gen;
50: }
51:
52: std::ostream& operator<<(std::ostream &out,LFSR &lfsr)
53: {
54:   out << lfsr.seed;
55:
56:   return out;
57: }
```

```
 1: C=g++ -g -Wall --std=c++98 -Werror
 2: E=.cpp
 3: O=test.o LFSR.o
 4: P=ps2a
 5: BOOST= -lboost_unit_test_framework
 6: #SFML= -lsfml-graphics -lsfml-window -lsfml-system
 7: all: $(P)
 8: $(P):$(O)
 9:         $(C) -o $(P) $(O) $(BOOST)
10:
11: $(E).o:
12:         $(C) -c $<  -o $@
13:
14: clean:
15:         rm $(O) $(P)
```