

```
1: // Copyright 2022
2: // By Dr. Rykalova
3: // Edited by Dr. Daly
4: // test.cpp for PS1a
5: // updated 5/12/2022
6:
7: #include "FibLFSR.hpp"
8:
9: #define BOOST_TEST_DYN_LINK
10: #define BOOST_TEST_MODULE Main
11: #include <boost/test/unit_test.hpp>
12:
13: BOOST_AUTO_TEST_CASE(testStepInstr1) {
14:     FibLFSR l("1011011000110110");
15:     BOOST_REQUIRE_EQUAL(l.step(), 0);
16:     BOOST_REQUIRE_EQUAL(l.step(), 0);
17:     BOOST_REQUIRE_EQUAL(l.step(), 0);
18:     BOOST_REQUIRE_EQUAL(l.step(), 1);
19:     BOOST_REQUIRE_EQUAL(l.step(), 1);
20:     BOOST_REQUIRE_EQUAL(l.step(), 0);
21:     BOOST_REQUIRE_EQUAL(l.step(), 0);
22:     BOOST_REQUIRE_EQUAL(l.step(), 1);
23: }
24:
25: BOOST_AUTO_TEST_CASE(testStepInstr2) {
26:     FibLFSR l2("1011011000110110");
27:     BOOST_REQUIRE_EQUAL(l2.generate(9), 51);
28: }
29:
30: BOOST_AUTO_TEST_CASE(testStepInstr3) { // length_error - parameter length
!= 16
31:     BOOST_CHECK_THROW(FibLFSR l3(""), length_error);
32:     BOOST_CHECK_THROW(FibLFSR l3("1111"), length_error);
33:     BOOST_CHECK_THROW(FibLFSR l3("11111111111111111111"), length_error);
34: }
35:
36: BOOST_AUTO_TEST_CASE(testStepInstr4) { // invalid_argument - parameter co
ntains !1 && !0
37:     BOOST_REQUIRE_THROW(FibLFSR l4("1011011000110112"), invalid_argument);
38:     BOOST_REQUIRE_THROW(FibLFSR l4("101101100011011c"), invalid_argument);
39: }
40:
41: BOOST_AUTO_TEST_CASE(testStepInstr5) { // Both length_error and invalid_a
rument
42:     BOOST_REQUIRE_THROW(FibLFSR l5("abcd"), length_error);
43:     BOOST_REQUIRE_THROW(FibLFSR l5("abcdabcdabcdabcdabcd"), length_error);
44: }
```