```
 1: #include "LFSR.hpp"
 2: #include <iostream>
 3: #include <vector>
 4: #include <string>
 5: #include <sstream>
 6: using namespace std;
 7:
 8: LFSR::LFSR(string seed, int tap){
 9:         for (int i = 0; i < seed.length(); ++i)
10:         {
11:                 if (seed[i] == '1')
12:                 {
13:                         arr.push_back(1);
14:                 }
15:                 if (seed[i] == '0')
16:                 {
17:                         arr.push_back(0);
18:                 }
19:         }
20:         num = tap;
21: }
22:
23: LFSR::~LFSR()
24: {
25:
26: }
27:
28: ostream& operator<<(ostream& out, const LFSR& right)
29: {
30:         string answer;
31:         std::stringstream ss;
32:         for (int i = 0; i < right.arr.size(); ++i)
33:         {
34:                 ss << right.arr[i];
35:         }
36:
37:         out << ss.str();
38:         return out;
39: }
40:
41:
42: int LFSR::step()
43: {
44:         int temp;
45:         int tp = (this->arr.size()-1) - this->num;
46:         if (this->arr[0] == 1 && this->arr[tp] == 0)
47:         {
48:                 temp = 1;
49:         }
50:         if (this->arr[0] == 0 && this->arr[tp] == 1)
51:         {
52:                 temp = 1;
53:         }
54:         if (this->arr[0] == this->arr[tp])
55:         {
56:                 temp = 0;
57:         }
58:         for (int i = 0; i < this->arr.size()-1; i++)
59:         {
60:                 this->arr[i] = this->arr[i+1];
61:         }
62:         this->arr[this->arr.size()-1] = temp;
63:         return temp;
64:
65: }
```

```
66:
67: int LFSR::generate(int k)
68: {
69:         int answer = 0;
70:         int temp;
71:         for (int i = 0; i < k; ++i)
72:         {
73:                 temp = this->step();
74:                 answer *= 2;
75:                 answer += temp;
76:         }
77:         return answer;
78: }
79:
```