```
    1: /*
    2: Name: ALbara Mehene
    3: Date: 9/25/2016
    4: Computing IV
    5:
    6: */
    7:
    8:
    9: #include <iostream>
   10: #include <string>
   11: #include <cmath>
   12: #include "LFSR.hpp"
   13:
   14: //constructor
   15: LFSR::LFSR(std::string seed_, int tap_){
   16:
   17:         seed = seed_;
   18:         tap = tap_;
   19:
   20: }
   21:
   22:
   23: int LFSR::step(){
   24:         int bit;
   25:         int size;
   26:
   27:         size = seed.length();//stored the amount of elements
   28:
   29:         bit =  seed.at(0) ^ seed [size - tap - 1]; //Took the total elements
 and subtracted by the tap and by 1
   30:
   31:
   32:         seed.erase(0, 1); // erased the front element
   33:
   34:         if(bit == 1){//condition if its 1, it would return the chracter 1
   35:                 seed.push_back('1');
   36:         }
   37:         else{//returns 0 if its anything else
   38:                 seed.push_back('0');
   39:         }
   40:         //returns bit to test the test.cpp
   41:         return bit;
   42: }
   43:
   44: int LFSR::generate(int k){
   45:         int temp = 0;
   46:
   47:         //Condition to test the generate function in test.cpp
   48:         for(int i = k - 1; i >= 0; i--){
   49:                 if(step() == 1){
   50:                         temp += pow(2,i);
   51:                 }
   52:         }
   53:         return temp;
   54:
   55: }
   56: //prints out the string if I were to use the a main. It was not required in
this assigment
   57: std::ostream& operator<< (std::ostream &out, LFSR &lfsr){
   58:         out << lfsr.seed;
   59:         return out;
```

```
60: }
61:
62:
63:
64: LFSR::~LFSR(){
65:
66: }
67:
68:
69:
70:
```