```
    1: #include "GuitarString.hpp"
    2:
    3: /*create a guitar string of the given frequency using a
    4: sampling rate of 44,100.*/
    5: GuitarString::GuitarString(double frequency){
    6:         G_cap = (ceil(SAMPLING_RATE/frequency));
    7:
    8:         this->_rb = new RingBuffer(G_cap);
    9:
   10:         //fill with 0s to represent the guitar string at rest
   11:         for(int i = 0; i < G_cap; i++){
   12:                 this->_rb->enqueue(0);
   13:         }
   14: }
   15:
   16: /*create a guitar string with size and initial values are given
   17: by the vector*/
   18: GuitarString::GuitarString(std::vector <sf::Int16> init){
   19:
   20:         this->_rb = new RingBuffer(init.size());
   21:
   22:         for (std::vector<sf::Int16>::iterator i = init.begin(); i != init.en
d(); ++i){
   23:       _rb->enqueue(*i);
   24:
   25:         }
   26: }
   27:
   28: GuitarString::GuitarString(){
   29:
   30: }
   31:
   32: /*Pluck the guitar string by replacing the buffer with random
   33: values, representing white noise*/
   34: void GuitarString::pluck(){
   35:                 //empty the buffer
   36:                 delete _rb;
   37:                 _rb = new RingBuffer(G_cap);
   38:
   39:                 //replace with random noise (white noise)
   40:                 for(int i = 0; i < G_cap; ++i){
   41:
   42:                         _rb->enqueue((int16_t)(rand() & 0xffff));
   43:                 }
   44:
   45: }
   46:
   47: /*advance the simulation one time step */
   48: void GuitarString::tic(){
   49:         _ticNum++;
   50:
   51:         double first = _rb->dequeue();
   52:         double front = sample();
   53:
   54:         //std::cout << "Decay: " << DECAY << std::endl;
   55:
   56:         double value = DECAY * ((first + front)/2);
   57:
   58:         //std::cout << "value:   " << value << std::endl;c;
   59:         _rb->enqueue(value);
   60: }
```

```
61: /*return the current sample*/
62: sf::Int16 GuitarString::sample(){
63:         //May be returning this incorrectly
64:         return _rb->peek();
65: }
66:
67: /*return number of times tic was called so far*/
68: int GuitarString::time(){
69:
70:         return _ticNum;
71: }
72:
73: GuitarString::~GuitarString(){
74:         //need to use delete function
75:         delete _rb;
76:
77: }
```