Arnav Shirodkar CMSC201:Data Structures

Email: as9086@bard.edu

Date: 20/02/2020

Assignment Description: Create a class Ringbuffer that creates a circular queue. Using the ringbuffer, create 2 additional classes GuitarString and GuitarHeroLite that use the Ringbuffer as well as the Karplus-Strong update to create 2 octaves

a guitar audio output.

Collaboration Statement: I worked on this file with Mason and Tina, getting further insight into how to debug my Ringbuffer class and GuitarHeroLite class from Mason and Tina respectively.

**RingBuffer.java**

import java.util.NoSuchElementException;

public class Ringbuffer{

private int n = 0;

private int max= 0;

private int first = 0;

private int last = 0;

private double[] buffer;

Ringbuffer(int capacity){

buffer = new double[capacity];

max = capacity;

}

int size(){

if (n >= max);{

n = max;

}

return n;

}

boolean isEmpty(){

if (n == 0){

return true;

} else {

return false;

}

}

boolean isFull(){

if (n == buffer.length){

return true;

} else {

return false;

}

}

void enqueue(double x) {

if (n == max && first == last){ // if full, move first and last together

buffer[last++] = x;

first++;

if(last == buffer.length){ last = 0; }

if(first == buffer.length){ first = 0; }

} else {

buffer[last++] = x;

if (last == buffer.length) {

last = 0;

}

}

if(!isFull()) {

n++;

} else {

n = max;

}

}

double dequeue(){

if(isEmpty()){

throw new NoSuchElementException("Queue Underflow");

}

//buffer[first] = null cannot be written here – Loitering is an issue for an array filled with object

n--;

double exitValue = buffer[first];

first++;

if(first == buffer.length){

first=0;

}

return exitValue;

}

double peek(){

if(!isEmpty()){

return buffer[first];

} else {

throw new NoSuchElementException("Queue Underflow");

}

}

void print(){ // print method I used to check and debug the Ringbuffer

for(int i=0;i<max;i++){

System.out.println(buffer[i]);

}

}

}

**GuitarString.java**

public class GuitarString {

private int n;

private Ringbuffer joe;

private int count;

GuitarString(double Frequency){

n = (int)(44100/Frequency);

joe = new Ringbuffer (n);

for(int i =0; i<= n ; i++) {

joe.enqueue(0);

}

}

void pluck(){

for(int i =0; i<=n ; i++){

double white = (Math.random() - 0.5);

joe.enqueue(white);

}

count++;

}

void tic(){

double firstVal = joe.dequeue();

double secondVal = joe.peek();

double newValue = 0.994 \*((firstVal+secondVal)/2);

joe.enqueue(newValue);

count++;

}

double sample(){

return joe.peek();

}

int time(){

return count;

}

}

**GuitarHeroLite.java**

import edu.princeton.cs.algs4.\*;

public class GuitarHeroLite {

public static void main(String[] args) {

int capacity = 37;

String keyboard = "q2we4r5ty7u8i9op-[=zxdcfvgbnjmk,.;/' ";

GuitarString[] Guitar = new GuitarString[capacity];

for (int i = 0; i < capacity; i++) {

Guitar[i] = new GuitarString((440) \* Math.pow(1.05956, i - 24));

}

while (true) {

// check if the user has typed a key; if so, process it

if (StdDraw.hasNextKeyTyped()) {

char key = StdDraw.nextKeyTyped();

if(keyboard.indexOf(key)!= -1) {

// returns -1 if any other key not in the string is pressed

int i = keyboard.indexOf(key);

Guitar[i].pluck();

}

}

double sample=0;

for (int i = 0; i < capacity; i++) {

sample = sample + Guitar[i].sample();

}

StdAudio.play(sample);

for (int i = 0; i < capacity; i++) {

Guitar[i].tic();

}

}

}

}