FInal Project Creative Work:

For my creative work, I decided to focus on a looping music piece that implements the golden ratio, randomness, and random walks.

The first idea that came to mind for a looping piece was to make a melody that would sound nice no matter what when looped. The most popular 4 chord songs use C F A G, so I decided to loop these. The melody goes CACAFGGsAs, but instead of chords I play the I, II, III, V in an ascending scale and then descending scale. For the sharps, I play only the major chords I, III, V because they aren't in the key.

For the beginning of the piece, I decided to use a drum cymbal combo to create a fast, upbeat start. I set a random seed as well because Sonic Pi doesn't have true randomness so that creators can always retrieve random generated sounds if they like them. Then, I set the pause between loops to be 4 times the golden ratio. Next, I set my sample to "guit_e_fifths" which plays notes on an e guitar but at intervals of perfect fifths. Here, I put the rate to be random which essentially makes the frequency random. The amount of time between each perfect fifth is governed by the rhythm set at the top. The line "rhythm = spread(5,8)" indicates a spread of 5 notes within 8 beats.

Since I've already explained the melody, I will move on to the last section which is the random walks. I start with a random walk array which will determine the frequency to play my notes at. Then I have a probability array for each index of the random walk array. This probability array indicates the probability of moving from the current index to any other index. I decided to make the probability array imitate an absorbing boundary. Wherever the current point is in the random walk array, the probability array will be calibrated to favor going to closer indices in the array. For example, if the random walk array was in the middle, it would have a

higher chance of going to adjacent indices than the far indices. If the random walk was at the end, then it would have a hard time jumping to the other end of the array. The probability array is made up of numbers 0-49. Every time the loop is repeated, there will be a number X selected from 0-49. An example array would be [6, 12, 25, 30, 36, 40, 49]. I then run a for loop from left to right and if X is less than the current index, then the current index corresponding to the random walk arr value will be the next point. If X was 35, then index 5 in the random walk array would be the new frequency for notes in the loop. There are three samples of sounds that I use for this random walk. A snare, a hum, and a glass rub.

Sonic Pi Code:

```
use_bpm 110
rhythm = spread(5, 8)
use_random_seed = 1235634

live_loop :drum do
  use_synth :fm
  sample :drum_bass_hard, amp: 0.8
  sleep 0.5
  play :e2, release: 0.2
  sample :elec_cymbal, rate: 12, amp: 0.6
  sleep 0.5
end
sleep 1.618 * 4

arr = [0.75, 0.8, 0.9, 1, 1.1, 1.2, 1.25]
random = 1

guitar_amp = 0.6
live_loop :harmonics do
  random = random * arr.choose
  if random < 0.7 or random > 1.6 then
    random = 1
  end
  if rhythm.tick then
```

```
    sample :guit_e_fifths, rate: random, amp: guitar_amp
  end
  sleep [1, 1.25, 1.5].choose
end

sleep 1.618  * 4
guitar_amp = 0
cnotes = (ring "c", "d", "e", "g")
anotes = (ring "a", "b", "c", "e")
fnotes = (ring "f", "g", "a", "c")
gnotes = (ring "g", "a", "b", "d")
gsnotes = (ring "gs", "c", "ds", "g")
asnotes = (ring "as", "d", "f", "a")
notes = (ring cnotes, anotes, cnotes, anotes, fnotes, gnotes, gsnotes, asnotes)

last_note = (ring :C, :A, :C, :A, :F, :G, :Gs, :As)
last_note_oct = (ring 7, 6, 7, 6, 7, 7, 7, 7)

coct = (ring 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 6, 6, 6, 6)
aoct = (ring 2, 2, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 6, 6)
foct = (ring 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 6, 6, 6, 6, 7)
gsoct = (ring 3, 4, 4, 4, 4, 5, 5, 5, 5, 6, 6, 6, 6, 7, 7, 7)
asoct = (ring 3, 3, 3, 3, 3, 4, 4, 5, 4, 5, 5, 6, 5, 6, 6, 7)
octaves = (ring coct, aoct, coct, aoct, foct, foct, gsoct, gsoct)

with_fx :reverb do
  live_loop :arp do
    use_synth :tri
    tick(:notes)
    16.times do
      random = random * arr.choose
      if random < 0.5 or random > 1 then
        random = 0.75
      end

      play note(notes.look(:notes).tick(:forward), octave: octaves.look(:notes).look(:forward)), amp:
random
      sleep 0.25
    end
    play note(last_note.tick, octave: last_note_oct.look)
    sleep 0.25
    15.times do
      play note(notes.look(:notes).reverse.tick(:rev), octave:
octaves.look(:notes).reverse.look(:rev)), amp: 0.5
```

```
    sleep 0.25
   end
   tick(:rev)

  end
end



sleep 1.618 * 4
arr = [0.85, 0.9, 1, 1.1, 1.15]
random = 0.3
use_synth :tb303
with_fx :reverb do |rev|
 loop do
   guitar_amp = 0.6
   random = random * arr.choose
   if random < 0.1 or random > 0.8 then
    random = 0.5
   end
   glass_amp = 1.5
   control rev, mix: random
   with_fx :slicer, phase: 0.25 do
    sample :ambi_glass_hum, sustain: 4, release: 4, amp: glass_amp
   end
   guitar_amp = 0
   glass_amp = 0

   #get new random seed
   32.times do
    use_random_seed = use_random_seed + 1
    x = rrand_i(0, 49)
    probability_array = [0, 0, 0, 0, 0, 0, 0]
    random_walk_arr = [0.4, 0.6, 0.8, 1, 1.2, 1.4, 1.6]
    random_walk = random_walk_arr.choose
    set :guitar_rate, 1

    if random_walk == 0.5 then
     set :probability_array, [11, 22, 33, 37, 41, 45, 49]
     for i in 0..(7)

      if (get[:probability_array])[i] > x then

       set :guitar_rate, random_walk_arr[i]
```

```
      break
    end
  end
end

if random_walk == 0.6 then
  probability_array = [9, 18, 27, 36, 42, 45, 49]
  for i in 0..(7)
    if probability_array[i] > x then

      set :guitar_rate, random_walk_arr[i]
      break
    end
  end
end

if random_walk == 0.8 then
  probability_array = [8, 16, 24, 32, 37, 43, 49]
  for i in 0..(7)
    if probability_array[i] > x then

      set :guitar_rate, random_walk_arr[i]
      break
    end
  end
end

if random_walk == 1 then
  probability_array = [7, 14, 21, 28, 35, 41, 49]
  for i in 0..(7)
    if probability_array[i] > x then

      set :guitar_rate, random_walk_arr[i]
      break
    end
  end
end

if random_walk == 1.2 then
  probability_array = [6, 13, 19, 25, 33, 41, 49]
  for i in 0..(7)
    if probability_array[i] > x then

      set :guitar_rate, random_walk_arr[i]
```

```
        break
      end
    end
  end

  if random_walk == 1.4 then
    probability_array = [5, 12, 17, 23, 28, 37, 49]
    for i in 0..(7)
      if probability_array[i] > x then

        set :guitar_rate, random_walk_arr[i]
        break
      end
    end
  end

  if random_walk == 1.6 then
    probability_array = [4, 11, 16, 22, 327, 36, 49]
    for i in 0..(7)
      if probability_array[i] > x then

        set :guitar_rate, random_walk_arr[i]
        break
      end
    end
  end

  use_synth :kalimba
  sample :ambi_haunted_hum, rate: get(:guitar_rate), sustain: 2, amp: 0.6

  use_synth :dpulse
  sample :ambi_glass_rub, rate: get(:guitar_rate), sustain: 2, amp: 0.6


  use_synth :pretty_bell
  sample :sn_zome, rate: get(:guitar_rate), sustain: 2, amp: 0.6
  sleep 2


 end
end
end
```