Sonic Pi Generative Music Unit Plan
Lesson # 1 - Using Random Events to Write A Song

Lesson Objectives

Students will be able to write a program which simulates using random events to choose sequences of notes and rhythms to be used in a song

Suggested Duration

1 period (45 minutes)

NYS Computer Science and Digital Fluency Learning Standards

7-8.IC.1 Compare and contrast tradeoffs associated with computing technologies that affect individuals and society.

7-8.CT.6 Design, compare and refine algorithms for a specific task or within a program.

Vocabulary

Indeterminate - Not definitely or precisely determined or fixed; not known in advance.
Aleotoric - Depending on the throw of a dice or on chance; random.

Assessments

- Assess ____. Check for the ability to:
  - Create data structures which hold a specific number of values
  - Use methods that randomly select values from data structures
  - Use a loop to repeat the same functions with different arguments

Do Now

Play students excerpt of “Music of Changes” by John Cage.
Link: John Cage: Music of Changes (1951)

While listening, have students write down:
“I notice…” and “I wonder…”

Get student responses.

Address student wonders about how the music was written.
Ask students how they think this music was written. Get responses.

Inform students this piece was written using a series of coin flips to choose each part of the piece including notes values, volume, duration, tempo etc.
This type of music is called Indeterminate or Aleatoric. John Cage felt that this type of composing was a way to remove human choice and ego from the musical composition process by having all decisions made by random events instead of the composer making those decisions themselves.

Lesson

Part 1 - Making a song based on random events

1. Inform students they are going to make their own composition based on random events.

2. Hand out ***Randomized Song Instructions and Template sheets.***
   Go over directions for the assignment.
   Show students Random.org webpages for dice rolls and coin flips.
   Do a couple of examples with students first.
   - Have students choose 6 random number
   - Have students choose 2 random sleep values
   - Go through dice roll and coin flips to choose values

3. Have students complete the song template.

4. Once students have finished the song template, they can go into Sonic Pi and enter the data for their song and listen to how it sounds.

   Note: Tell students the most efficient way to enter their code into Sonic Pi is to copy and paste the words ***play*** and ***sleep*** 16 times first. Then go through and enter the values for each function.

Part 2 - Refactoring Randomized Song

1. Tell students we want to recreate the process of writing this song in Sonic Pi without needing to roll dice or flip a coin.

   Things for students to consider:
   - How should we store our notes and sleep values
   - How can we randomly choose our notes and sleep values once we have picked the values we want to use.
   - How many times does our code play and sleep

2. Have students work on how to write a program to make a Randomized song.

Example Code solution:

```
notes = [40, 50, 60, 70 ,80, 60]
time = [0.5, 1.25]
16.times do
  play notes.choose
  sleep time.choose
end
```

Wrap Up/Assessment

Inform students that there are functions in Sonic Pi that are capable of replicating rolling a dice or flipping a coin.

dice - This will return a value between 1 and 6

rand_i - This function will return a value of 0 or 1

Ask students to consider what might be the pros/cons of using a computer program to automate the process of rolling dice and coin flips.
Also have them consider what are some real world examples of where we would need to have a number or sequences of numbers generated randomly.

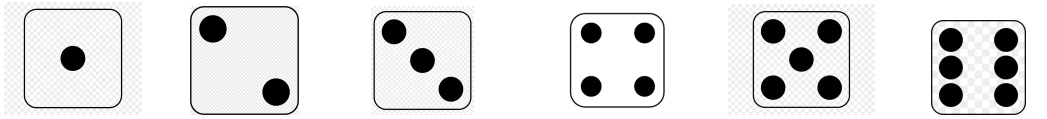
Students should submit their code and complete the wrap up questions in submission document posted on Google Classroom
Exit Slip: Lesson 1 - Random Event Simulation submission document

Randomized Song Instructions

Task: You will create a song that is composed by using random events to decide the notes and sleep values.

Choose 6 numbers between 40 and 100. These will be your play values.
Write each number above one of the sides of the die:

_____ _____ _____ _____ _____ _____



Choose two numbers between 0.125 and 2. These will be your sleep values.
Write each number above one of the sides of the coin

_____ _____

Directions

You are going to make a piece of music that is determined by the results of random events (dice rolls and coin flips)

Go here for dice roll simulation (choose ***roll 1 virtual dice***): https://www.random.org/dice/

Go here for coin flip simulation (choose ***flip 1 virtual coin***): https://www.random.org/coins/

After each roll of the die, write down the number which corresponds to that side of the die for the play value.

After each coin flip, write down the number which corresponds to that side of the coin for the sleep value.

Note: It is not specified whether you should alternate between a die roll and coin flip or if you should do all the die rolls and then all the coin flips. You can make this decision (or have a random event determine what you should do!)

Randomized Song Template

Write the results of your random events here.

play		**play**	
sleep		**sleep**	
play		**play**	
sleep		**sleep**	
play		**play**	
sleep		**sleep**	
play		**play**	
sleep		**sleep**	
play		**play**	
sleep		**sleep**	

Random Event Simulation submission document

1. Copy and paste the code for your randomized song in the box below

2. Consider 2 Pros and 2 Cons to using a computer simulation for random events like dice rolls and coin flips.
   Write your responses in the boxes below

Pros	Cons

3. Give at least two real world examples where there needs to be a random number or a random sequence of numbers chosen.

Example #1	Example #2

Sonic Pi Generative Music Unit Plan
Lesson # 2 - Pseudo vs True randomness

Lesson Objectives

Students will be able to
- To understand the difference between true random and pseudorandom
- Include a method to generate true random output in their code

Suggested Duration

1 period (45 minutes)

NYS Computer Science and Digital Fluency Learning Standards

7-8.CT.1 Compare the results of alternative models or simulations to determine and evaluate how the input data and assumptions change the results.

Vocabulary

Pseudorandom - A sequence that appears to be statistically random, despite having been produced by a completely deterministic and repeatable process

Assessments

- Assess ____. Check for the ability to:
  - Write a block of code to produce a random sequence of numbers
  - Use the console to obtain information about their program
  - Determine the difference between true random and pseudorandom
  - Use a method to generate True random sequences

Do Now

Have students write out a series of 8 numbers between 40 and 100
Have students compare their series of numbers with students around them.

Ask how many students had the same sequence of numbers.
Establish that it would be very rare for multiple people to come up with the same sequence of randomly chosen numbers.

Lesson

Part 1 - Generating random numbers

1. Introduce students to ***rrand_i( ).***
   This function will choose a random integer between a specified range of numbers. The function takes two parameters, the minimum and maximum value of the range
   Example: rrand_i(10, 20) will return a number between 10 and 20.

2. Have students write a short program in Sonic Pi that will recreate the Do Now which uses the ***rrand_i*** function. However, tell them this program should play each value and then sleep for a certain amount of time.

3. Allow students 3-5 minutes to complete.
   Remind students who are stuck that you need to choose a random number 8 times, so they should include some type of loop.

Example of completed code

```
2  8.times do
3    play rrand_i(40, 100)
4    sleep 1
5  end
```

Part 2 - Pseudorandom

1. Have students run their program and write down the notes that appear in the console when the program has finished.

   Students should then compare their results with the students around them. Students should observe that everyone got the same results.

   Example of console output

```
{run: 5, time: 0.0}
 └─ synth :beep, {note: 85.0}

{run: 5, time: 1.0}
 └─ synth :beep, {note: 84.0}

{run: 5, time: 2.0}
 └─ synth :beep, {note: 68.0}

{run: 5, time: 3.0}
 └─ synth :beep, {note: 54.0}

{run: 5, time: 4.0}
 └─ synth :beep, {note: 46.0}

{run: 5, time: 5.0}
 └─ synth :beep, {note: 72.0}

{run: 5, time: 6.0}
 └─ synth :beep, {note: 90.0}

{run: 5, time: 7.0}
 └─ synth :beep, {note: 73.0}
```

Note: Students who do not get the same results may have included *use_random_seed* which will be discussed next

2. Explain to students that Sonic Pi, along with many other random number generators, does not actually generate random sequences of numbers. (The purpose of this is so that if you create a random number sequence that sounds good, you will be able to repeat it)

3. Ask students how we could fix this program to get different sequences of numbers. Students should recall ***use_random_seed*** from a previous lesson.

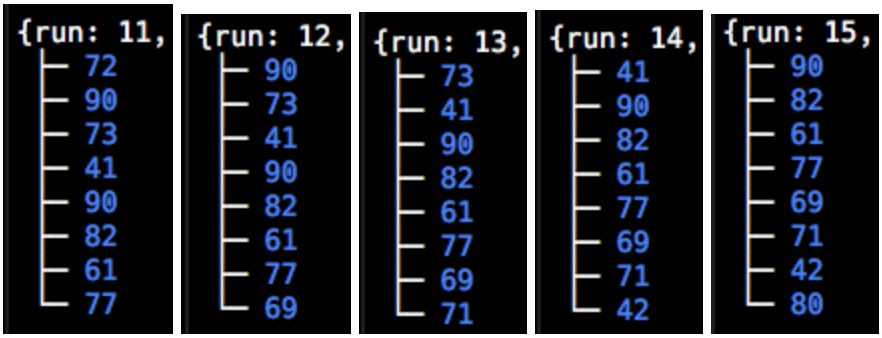
4. Have students add ***use_random_seed*** to the top of their code, but assign each student a specific number to use as an argument for that function. (1, 2, 3, 4, 5, 6, 7, 8 etc)

```
1  use_random_seed 5
2  8.times do
3    play rrand_i(40, 100)
4    sleep 1
5  end
```

Students should write down the sequence they get. Have them compare with each other. Now they should see that they are getting different sequences.

5. Ask for the results of 5-6 students who had sequential seed numbers (4-10, 11-16 etc) Write the results on the board

   Output examples (use_random_seed 5 - 9)



   Ask students if they see a pattern. They should notice that each sequence contains 7 of the same numbers as the previous sequence.

6. Explain that when Sonic Pi generates a number sequence using use_random_seed, it will always make the same number sequence (which is millions of numbers long) but it will displace the first number for each increment of the number passed as an argument in use_random_seed. This is what we call Pseudorandom. It appears random but it actually will produce the same results.

Part 3 - True Randomness

1. All random functions in Sonic Pi generate pseudorandom patterns, so even if we tried to do something like this:
   use_random_seed rrand_i(1, 100000)
   It will still always create the same pattern.

2. Introduce **Time.now**
   This function will output the current date and time
   Demo the following code for students

   puts Time.now
   Point out the output on in the console.
   2021-11-29 22:04:49.908957 -0500

3. We can convert this output into an integer by adding ***.to_i*** to the end of this function.

   Demo the following code for students

   puts Time.now.to_i
   Point out the output on in the console.
   1638241639

Run the code again and point out that the number output in the console is different

4. We can now use this function output as the argument to **use_random_seed** and will always get a different number which will change depending on the day and time we run our program. This is as close as we can get to true randomness in our code.

5. Have students use Time.now.to_i for use_random_seed and run their code. Have some students wait a few seconds before running their code to ensure they get different results.

   Ask students why they should wait instead of running their code at the same time
   (If they run their code at the same time, they will get the same output since the number is based on the date and time.)

   Example of code

```
use_random_seed Time.now.to_i
8.times do
  play rrand_i(40, 100)
  sleep 1
end
```

6. Explain that having random number sequences generated by using some outside influence to determine the seed is called True Random.

Wrap Up/Assessment

Have students read the following webpage which discusses the differences between PseudoRandom and True Random number generators.
When finished, students should complete the Lesson 2 - PseudoRandom Exit Slip

PseudoRandom Exit Slip

Answer the Following Questions

1. True / False (circle one) - The following code will always produce the same sequence of notes

```
use_random_seed rrand_i(1, 10000)
16.times do
  play rrand_i(40, 80)
  sleep 1
end
```

Explain why you chose your answer.

2. What is the difference between Pseudo Random and True Random?

3. Why does ***Time.now.to_i*** always provide us with a different number that we can use in the ***use_random_seed*** function?

Sonic Pi Generative Music Unit Plan
Lesson # 3 - Manipulating Data Structures

Lesson Objectives

Students will be able to use ring chain methods to change the size and values selected from data structures.

Suggested Duration

1 period (45 minutes)

NYS Computer Science and Digital Fluency Learning Standards

7-8.CT.1 Compare the results of alternative models or simulations to determine and evaluate how the input data and assumptions change the results.

Assessments

- Assess ____. Check for the ability to:
    - Use built in methods to select values from data structures
    - Use functions which return random values to select a random number of values from a data structure

Do Now

Students should complete ***Selecting Random Sequences*** worksheet

Lesson

Part 1 - Ring chain methods

1. After completing the worksheet, explain to students that each set of directions they did in the worksheet represent a method that can be applied to a sequence of numbers that is stored inside of a ring (a data structure specific to Sonic PI)

2. Provide the list of these methods to students.

- .drop(n)
- .take_last(n)
- pick(n)
- .shuffle
- .take(n)
- .reverse
- .drop_last(n)

3. Working in pairs, have students predict which method goes with which set of directions.
   They should write the method on the specified line of the worksheet.

Part 2 - Code examples in Sonic Pi

1. Demonstrate all methods in Sonic PI by printing the results on each method in the console.

2. When getting to responses that require students to choose a number between 1 and 6, have a student provide a chosen number for the first example.

   When moving on to the next examples, ask if students can think of a way that they could have Sonic Pi choose that number for them
   Possible response: dice, rrand_i
   Demonstrate each one

Code Examples

```
1
2   # Original sequence
3   seq = (ring, 50 ,52, 54, 55, 57, 59, 61, 62)
4
5   # store shuffled sequence in new variable
6   seqShuffled = seq.shuffle
7
8   # print new shuffled sequence
9   puts seqShuffled
10
11  # print reversed shuffled sequence
12  puts seqShuffled.reverse
13
14  # print first 4 numbers from shuffled sequence
15  puts seqShuffled.take(4)
16
17  # remove btwn 1-6 numbers from beginnning
18  puts seqShuffled.drop(dice)
19
20  # pick btwn 1-6 random numbers from sequence
21  puts seqShuffled.pick(rrand_i(1, 6))
22
23  # remove btwn 1-6 numbers from end
24  puts seqShuffled.drop_last(3)
25
26  # take last 2 numbers from sequence
27  puts seqShuffled.take_last(2)
```

Console Output

```
=> Starting run 3

{run: 3, time: 0.0}
  ├─ (ring 52, 50, 61, 55, 59, 62, 54, 57)
  ├─ (ring 57, 54, 62, 59, 55, 61, 50, 52)
  ├─ (ring 52, 50, 61, 55)
  ├─ (ring 62, 54, 57)
  ├─ (ring 50, 52, 59)
  ├─ (ring 52, 50, 61, 55, 59)
  └─ (ring 54, 57)
```

3. Call attention to the fact that the original sequence of notes is actually a major scale
   Replace original sequence ring with scale function - scale(50, :major) and show that the results will be the same

```
2   # Original sequence
3   seq = scale(50, :major)
```

Wrap Up/Assessment

Have students check their predictions of which methods go with which set of directions from the worksheet.

Students can choose remaining time to experiment with ring chain methods in Sonic Pi.

Challenge students to try and play through the notes in the rings. Consider issues that arise. Explain that these issues will be addressed in the next class.

Selecting Random Sequences worksheet

Original Number Sequence: 50, 52, 54, 55, 57, 59, 61, 62

Step 1: Rearrange the original number sequence. This will be your new number sequence that you will refer to in order to complete the following responses.

_____ _____ _____ _____ _____ _____ _____ _____

Ring method_____

Step 2: Complete the following responses using your new sequence each time.
DO NOT fill in the Ring method line yet.

1. Write your new sequence in reverse

Ring method_____

_____

2. Choose a number between 1 and 6.
   Take that many numbers from the front of the new sequence

Ring method_____

_____

3. Choose a number between 1 and 6.
   Take that many numbers from the back of the new sequence

Ring method_____

_____

4. Choose a number between 1 and 6.
   Randomly choose that many numbers from the new sequence

Ring method_____

_____

5. Choose a number between 1 and 6.
   Remove that number of numbers from the end of the new sequence.
   Write down the remaining number in the sequence

Ring method_____

_____

6. Choose a number between 1 and 6.
   Remove that number of numbers from the beginning of the new sequence.
   Write down the remaining number in the sequence

Ring method_____

Sonic Pi Generative Music Unit Plan
Lesson # 4 - Looping through data structures of different lengths

Lesson Objectives
Students will be able to iterate through data structures of different lengths in a loop structure.
Suggested Duration
1 period (45 minutes)
NYS Computer Science and Digital Fluency Learning Standards
7-8.CT.9 Read and interpret code to predict the outcome of various programs that involve conditionals and repetition for the purposes of debugging.
Vocabulary
N/A
Assessments
• Assess ____. Check for the ability to: - accurately explain what each line is doing in a block of code - use the **.length** method to accurately iterate through all indices of a data structure

Do Now
Have students log into Peardeck: Lesson 4 - Different Length Data Structures

```
1
2  seq = (ring, 50, 52, 54, 55, 57, 59, 61, 62).shuffle
3
4  use_random_seed Time.now.to_i
5  4.times do
6    new_seq = seq.take(dice)
7    puts new_seq
8    sleep 4
9  end
```

In peardeck slide, Students should explain what is happening in each line of this block of code.

Lesson

Part 1 - Making Predictions

1. After completing Do Now, have students write down predictions about what is going to happen in this code in peardeck slide

2. Have students run this code in Sonic Pi. After the code has run, they should write down each sequence of notes that appears in the console.

Example of Note Sequences in Console

```
=> Starting run 82

{run: 82, time: 0.0}
 └─ (ring 52, 50, 61)

{run: 82, time: 4.0}
 └─ (ring 52, 50, 61, 55, 59, 62)

{run: 82, time: 8.0}
 └─ (ring 52)

{run: 82, time: 12.0}
 └─ (ring 52, 50, 61, 55, 59)
```

3. Have students write observations in peardeck.
   Share comments which explain that you get a different amount of numbers each time.

4. In peardeck, have students choose which part of the code is causing us to get a different amount of numbers each time

   Answer: ***new_seq = seq.take(dice)***
   .take will choose a certain number from the front of the sequence and dice will alway return a random number between 1 and 6

Part 2 - Identifying Issues with loops #1

1. Have students copy and paste this code from the Peardeck slide into Sonic Pi

```
2  seq = (ring, 50, 52, 54, 55, 57, 59, 61, 62).shuffle
3
4  use_random_seed Time.now.to_i
5  new_seq = seq.take(dice)
6  puts new_seq
7  4.times do
8    play new_seq.tick
9    sleep 1
10 end
```

2. If we wanted to play through each of these sequences of different lengths using a .times do/end block, there would be a problem.

   Give students 1-2 minutes to discuss and predict what a potential problem would be. Have students share out responses. Asks students to clarify by explaining which part of the code might cause the problem.

3. Have students make observations in the Peardeck about what happens.
   Remind them to listen and watch the output in the console.
   Students should write their observations in the Peardeck

4. Share out responses.
   Possible responses: *When the sequence has less than 4 numbers, it repeats some of the numbers.*
   When the sequence has more than 4 numbers, not all the numbers are played

5. Explain that since the loop block happens 4 times, it will always play 4 notes, but this is a problem because each time there is a different amount of notes and there is only a 1 out of 6 chance it will be exactly 4 note.

   We want to find a way that it will always play all the numbers in the sequence, no matter how long the sequence is.

Part 3 - .length - Finding the length of a data structure

1. Introduce ***.length***
   If we put ***.length*** at the end of a variable containing a data structure (like new_seq), it will return the number of notes in the sequence.

   Demonstrate this by adding ***puts new_seq.length*** to the current code in Sonic Pi

```
5  new_seq = seq.take(dice)
6  puts new_seq
7  puts new_seq.length
```

   Have students look a the console results (Examples will vary)

```
{run: 90, time: 0.0}
 ├─ (ring 52, 50, 61, 55, 59, 62)
 └─ 6
```

   We see the actual number sequence which has 6 number and below we just see the number six which is what is returned when we use the .length method with a data structure.

2. We can then treat new_seq.length as a variable which holds the value of the length of the sequence of notes and add it to our .times do/end block

```
 9  new_seq.length.times do
10    play new_seq.tick
11    sleep 1
12  end
```

   Tell students that if this feels too complex for them, they can create a new

variable to store the .length of the sequence and use that with the .times do/end block

```
 9  len = new_seq.length
10
11  len.times do
12    play new_seq.tick
13    sleep 1
14  end
```

3. Have students run the code multiple times to see that each time it plays the exact number of times as are in the sequence

Part 4 - Identifying Issues with loops #1

1. If we want to create a loop of different length sequences, we can put everything into a live loop and each time through the loop it will choose a new sequence length.

   Have students wrap their code inside of a live loop

   Code Example

```
 2  seq = (ring, 50, 52, 54, 55, 57, 59, 61, 62).shuffle
 3
 4  live_loop :randomLoop do
 5    use_random_seed Time.now.to_i
 6    new_seq = seq.take(dice)
 7    puts new_seq
 8    puts new_seq.length
 9
10    len = new_seq.length
11    len.times do
12      play new_seq.tick
13      sleep 1
14    end
15  end
```

2. Have students run this code and observe the output in the console.

They should look at what is printed in the console as well as the actual notes that are being played.
There is a problem with what we want to happen and what is actually happening.
Give students time to see if they can identify the problem.
Write their prediction in the Peardeck slide

3. If students are struggling to find the issues, provide them with the following hints

Hints:

- Every time the console prints out the note sequence and sequence length, we are at the beginning of the .times do/end block which means we should be at the beginning of the sequence again.
- Before we added the live loop, every time we ran the code, it started with the same number.

4. **Problem:** The sequence doesn't always start on the first note in the sequence.

5. Because we are using .tick, the count of tick continues to count up and will not always line up with the first note in the ring.

To resolve this, we can just add a ***tick_reset*** before the .times do/end loop
This way the new sequence will always start at the beginning of the sequence

```
10    len = new_seq.length
11    tick_reset
12    len.times do
13      play new_seq.tick
14      sleep 1
15    end
```

Wrap Up/Assessment

Assignment: Create a live loop which generates different lengths of note sequences that uses a ring method to change the length of the sequence

Students should copy and paste code into Peardeck slide

Checklist: Lesson 4 - Data Structure Assignment Checklist Assessement Tool

Aim: Students will be able to iterate through data structures of different lengths in a loop structure.

```
1
2  seq = (ring, 50, 52, 54, 55, 57, 59, 61, 62).shuffle
3
4  use_random_seed Time.now.to_i
5  4.times do
6    new_seq = seq.take(dice)
7    puts new_seq
8    sleep 4
9  end
```

Students, write your response!

Write out what each line of code is doing.
Format your response like this:
Line 2: *explanation of what this line is doing*
Line 4: *explanation of what this line is doing*

```
1
2  seq = (ring, 50, 52, 54, 55, 57, 59, 61, 62).shuffle
3
4  use_random_seed Time.now.to_i
5  4.times do
6    new_seq = seq.take(dice)
7    puts new_seq
8    sleep 4
9  end
```

Students, write your response!

Make a prediction about what this code is going to do

___

```
1
2  seq = (ring, 50, 52, 54, 55, 57, 59, 61, 62).shuffle
3
4  use_random_seed Time.now.to_i
5  4.times do
6    new_seq = seq.take(dice)
7    puts new_seq
8    sleep 4
9  end
```

Students, write your response!

Which line of code is causing is causing us to get a different amount of numbers each time?

```
1
2  seq = (ring, 50, 52, 54, 55, 57, 59, 61, 62).shuffle
3
4  use_random_seed Time.now.to_i
5  4.times do
6    new_seq = seq.take(dice)
7    puts new_seq
8    sleep 4
9  end
```

Students choose an option

Copy and paste this code into Sonic Pi and run it.

```
seq = (ring, 50, 52, 54, 55, 57, 59, 61, 62)

use_random_seed Time.now.to_i

new_seq = seq.take(dice)

puts new_seq

4.times do

  play seq.tick

  sleep 1

end
```

Write an observation about what you saw/heard happening in the code.

Students, write your response!

What is the problem we are having with our code?

Students, write your response!

Hints

– – –

Everytime the console prints out the note sequence and sequence length, we are at the beginning of the .times do/end block which means we should be at the beginning of the sequence again.

Before we added the live loop, every time we ran the code, it started with the same number.

Assignment:

Create a live loop which generates different lengths of note sequences that uses a ring method to change the length of the sequence.

Checklist:
https://docs.google.com/document/d/10YDjOf7TBPSt37JdO2xDpmEIxQgOSvNat9XRLXPEIfc/edit?usp=sharing

Copy and Paste your code to this slide

Data Structure Assignment Checklist

Each part is worth a total of 2 points.
2 - Accurately completes requirement
1 - Requirement is attempted but not completed accurately
0 - Does not include requirement

_____ Uses true randomness

_____ Uses data structure method to select values

_____ Uses method to select random number of values from data structure

_____ Stores manipulated data structure in variable

_____ Uses loop to iterate through all values of data structure

_____ = Total Score out of 10

Sonic Pi Generative Music Unit Plan
Lesson # 5 - Conditional Statements

Lesson Objectives

Students will be able to use conditional statements to provide different possible musical outcomes based on randomly generated conditions stored in variables

Suggested Duration

1 period (45 minutes)

NYS Computer Science and Digital Fluency Learning Standards

7-8.CT.6 Design, compare and refine algorithms for a specific task or within a program.

7-8.CT.7
Design or remix a program that uses a variable to maintain the current value of a key piece of information.

7-8.CT.9 Read and interpret code to predict the outcome of various programs that involve conditionals and repetition for the purposes of debugging.

Vocabulary

Assessments

- Assess ____. Check for the ability to:
    - Make if/else conditional statements
    - Store a randomly generated value in a variable
    - Accurately use single line conditional statements
    - Create nested conditional statements

Do Now

Have students log into Peardeck
Have students iterate through code to explain what is happening

```
1
2  live_loop :conditional do
3    use_random_seed Time.now.to_i
4    if rand_i == 0
5      play 60
6      sleep 1
7    else
8      play 80
9      sleep 1
10   end
11 end
```

Get students responses to explain what this code is doing

Lesson

Part 1 - Single Line conditional statements

1. In Sonic Pi, there is a way we can simplify a conditional statement into a single line of code:

   This:

```
2  live_loop :conditional do
3    use_random_seed Time.now.to_i
4    if rand_i == 0
5      play 60
6    end
7    sleep 1
8  end
```

Is the same as this:

```
2  live_loop :conditional do
3    use_random_seed Time.now.to_i
4    play 60 if rand_i == 0
5    sleep 1
6  end
```

This can be a good way to simplify our code to make it more readable and not have to add extra 'end's which are hard to keep track of and can make our code more confusing to read

2. Ask students how we might refactor the code in the Do Now using this method.

   Possible student solution:

```
2  live_loop :conditional do
3    use_random_seed Time.now.to_i
4    play 60 if rand_i == 0
5    play 80 if rand_i == 1
6    sleep 1
7  end
```

3. Have students copy and paste code sample from Peardeack into an empty buffer and observe what happens.

```
1  live_loop :conditional do
2    use_random_seed Time.now.to_i
3    puts "flip!"
4    play 60 if rand_i == 0
5    play 80 if rand_i == 1
6    sleep 1
7  end
```

   Write observation in Peardeck slide

Possible observations: Sometimes nothing plays, sometimes two notes play

4. Have students turn and talk to discuss predictions as for why this may be.
   Have students write predictions in Peardeck

5. Remind students that rand_i is a function that will return either 0 or 1
   Just like flipping a coin

   Explain code: If rand_i is a coin that we are flipping, when we have rand_i written twice in our code, it is like we are flipping two different coins. Sometimes they will both be heads, sometimes they will both be tails, sometimes one will be heads and one will be tails.

   Have students change the second if statement to rand_i == 0
   They will see that the program behaves the same

   Explain to students that there isn't necessarily anything wrong with this approach and it could be useful in their program if they wanted to have a different number of notes play each time

6. If they want to just have one note play each time, they need to store the result of the coin flip (rand_i) in a variable and use that variable name in the condition.

```
2  live_loop :conditional do
3    use_random_seed Time.now.to_i
4    flip = rand_i #variable to store result of ran
5    play 60 if flip == 0
6    play 80 if flip == 1
7    sleep 1
8  end
```

Part 2 - Exercise

Create a live loop.
Store the results of the dice roll in a variable.
Have a different note play for each number on the dice.
Use single line conditional statements.

Part 3 - Homework assignment

1. Present students with Choose Your Own Adventure Template
   Lesson 5 - Out of Class Assignment
2. Explain directions.
3. Go through Example assignment.
   Lesson 5 - Out of Class Assignment - Example
4. Assignment should be done on Google Classroom

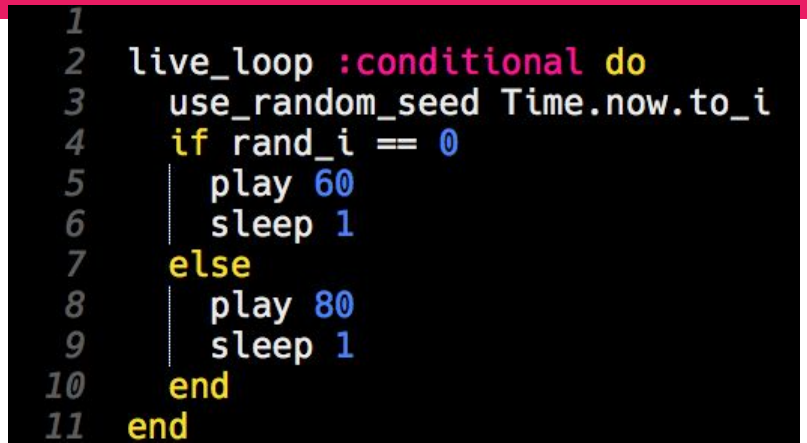
Wrap Up/Assessment

Present example code of solution for Dice roll exercise.

Example of end result

```
live_loop :conditional do
  use_random_seed Time.now.to_i
  roll = dice
  play 50 if roll == 1
  play 55 if roll == 2
  play 60 if roll == 3
  play 65 if roll == 4
  play 70 if roll == 5
  play 75 if roll == 6
  sleep 1
end
```

Aim: To use conditional statements to provide different output based on randomly generated conditions stored in variables



```
1
2  live_loop :conditional do
3    use_random_seed Time.now.to_i
4    if rand_i == 0
5      play 60
6      sleep 1
7    else
8      play 80
9      sleep 1
10   end
11 end
```

Write out what each line of code is doing.
Format your response like this:
Line 2: *explanation of what this line is doing*
Line 4: *explanation of what this line is doing*

```
1
2  live_loop :conditional do
3    use_random_seed Time.now.to_i
4    if rand_i == 0
5      play 60
6      sleep 1
7    else
8      play 80
9      sleep 1
10   end
11 end
```

Students, write your response!

Copy and paste this code into Sonic Pi and run it.

```
live_loop :conditional do
  use_random_seed Time.now.to_i
  puts "flip!"
  play 60 if rand_i == 0
  play 80 if rand_i == 1
  sleep 1
end
```

Write an observation about what you saw/heard happening in the code.

Why do you think our code is behaving this way?

Students, write your response!

Task:

Create a live loop.
Store the results of the dice roll in a variable.
Have a different note play for each number on the dice.
Use single line conditional statements.

Copy and paste your code here

Choose Your Own Adventure Story

Line 1 - Create a scenario where a character is presented with a choice (Give two options)

Option 1:
Option 2:

If your character chooses the first option from line 1:
Write the outcome of the character's choice on line 2. This outcome should result in having to make another choice.

Line 2

Option 1:
Option 2:

If your character chooses the first option from line 2:
Write the outcome of the character's choice on line 3. This should end the story.

Line 3

If your character chooses the second option from line 2:
Write the outcome of the character's choice on line 4. This should end the story.

Line 4

If your character chooses the second option from line 1:
Write the outcome of the character's choice on line 5. This outcome should result in having to make another choice.

Line 5

Option 1:
Option 2:

If your character chooses the first option from line 5:
Write the outcome of the character's choice on line 6. This should end the story.

Line 6

If your character chooses the second option from line 5:
Write the outcome of the character's choice on line 7. This should end the story.

Line 7

Choose Your Own Adventure Story - Example

Line 1 - Create a scenario where a character is presented with a choice (Give two options)

You enter a room. On the table in this room, there is a plate of cookies and a single cupcake.
Option 1: You eat the cookie
Option 2: You eat the cupcake

If your character chooses the first option from line 1:
Write the outcome of the character's choice on line 2.
This outcome should result in having to make another choice.

Line 2

Eating the cookie has given you the power to read people's minds.
Option 1: You see a doctor to try and figure out what has happened
Option 2: You use this power to your own benefit

If your character chooses the first option from line 2:
Write the outcome of the character's choice on line 3. This should end the story.

Line 3

The doctor discovers that the effects are only temporary. You spend the day resting at home and are back to normal the next day.

If your character chooses the second option from line 2:
Write the outcome of the character's choice on line 4. This should end the story.

Line 4

You use this power to find out the answers to your math test. You get in trouble for cheating, fail the test and can no longer read minds.

If your character chooses the second option from line 1:
Write the outcome of the character's choice on line 5.
This outcome should result in having to make another choice.

Line 5

The cupcake causes all your hair to fall out.
Option 1: You get a wig
Option 2: You wear a hat.

If your character chooses the first option from line 5:
Write the outcome of the character's choice on line 6. This should end the story.

Line 6

The adhesive from the wig causes an allergic reaction to your scalp. Your hair never grows back and your head becomes too sensitive to cover up with anything else.

If your character chooses the second option from line 5:
Write the outcome of the character's choice on line 7. This should end the story.

Line 7

While wearing the hat, Someone on the street mistakes you for someone they owe $1000 to. They give you the money and you use it to buy the latest iPhone. Your hair grows back two days later.

Sonic Pi Generative Music Unit Plan
Lesson # 6 - Nested Conditional Statements

Lesson Objectives

Students will be able to use nested conditional statements with specified probability to determine which outcome will be chosen

Suggested Duration

1 period (45 minutes)

NYS Computer Science and Digital Fluency Learning Standards

7-8.CT.8 Develop or remix a program that effectively combines one or more control structures for creative expression or to solve a problem.

Vocabulary

Probability - The extent to which something is probable; the likelihood of something happening or being the case.

Assessments

- Assess ____. Check for the ability to:
  - Use function to affect probability of outcome for conditional statements
  - Create program which uses nested conditional statements

Do Now

Have students log into Peardeck: Lesson 6 - Nested Conditional Lesson Slides

Have students partner up to read their Choose Your Own Adventure stories which should have been completed as homework.

Have the writer of the story read line 1. When given the choice, the other students will flip a coin (Use https://www.random.org/coins/)
Heads - PIck the first choice (line 2)
Tails - Pick the second choice (line 5)

Writer reads the next line. When given the choice, the other student will flip a coin.

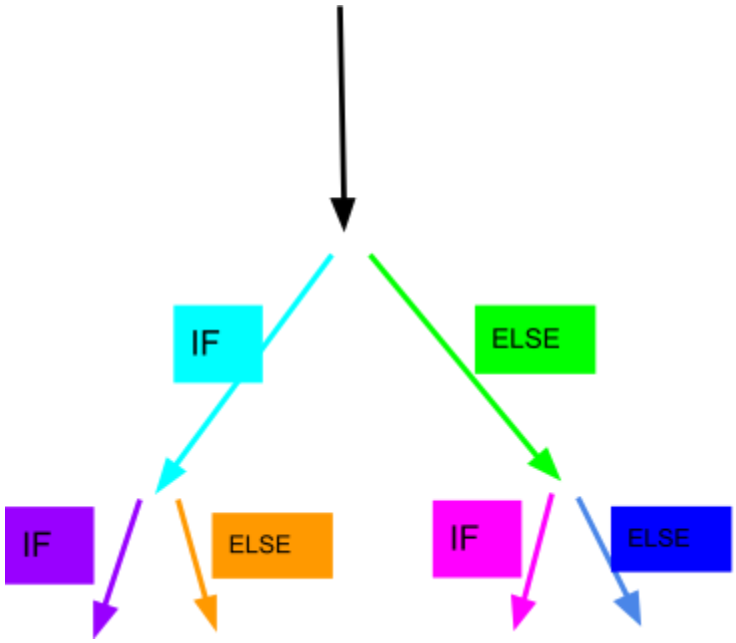
Heads - Pick the first choice
Tails - Pick the second choice.

After they have finished, have students switch rolls and do the other story.
This time, they should use a dice roll to choose which option to pick. (Use https://www.random.org/dice/)
If they roll a 6, pick the first option. Otherwise, pick the second option.

Lesson

Part 1 - Nested conditionals

1. Conditionals are like approaching a fork in the road and having to choose a path. However once we have chosen one path we may wind up with another choice later on.
   This is called a nested condition.
   Present the following image:



Part 2 - Probability

1. Ask students the following question:

When reading your stories with the coin flip, what was the probability of you choosing the first choice?

Students should answer in Peardeck slide
Possible responses: 50%, 50/50, 1 out of 2, 1 in 2

2. When we flip a coin there are 2 possible out comes - heads or tails. Percentage-wise, this would be a 50% chance of each.

   We could also phrase it as a one in two chance.

3. Ask students the following question:
   When reading your stories with the dice roll, what was the probability of you choosing the first choice?

   Students should answer in Peardeck slide
   Possible responses: 1 out of 6, 1 in 6

4. Explain to students that the probability of getting the first choice was less with the dice roll than the coin flip.

Part 3 - one_in(n) function

1. In Sonic Pi, there is a function we can use as a condition to determine a specific probability of which path we will take.

2. Introduce ***one_in(n)***

   This function returns a boolean value (true or false). The number provided as an argument determines the probability that the function will return the value of true

   Example: ***one_in(2)*** - 1 in 2 chance the return value will be true
   one_in(6) - 1 in 6 chance the return value will be true.

3. Show students Choose your Own Adventure Code Template

   Iterate through each line of the code. Point out the nested conditional statements and probability.

```
use_random_seed Time.now.to_i
puts # Enter line 1
if one_in(2)
  puts # Enter line 2
  if one_in(2)
    puts # Enter line 3
  else
    puts # Enter line 4
  end
else
  puts # Enter line 5
  if one_in(2)
    puts # Enter line 6
  else
    puts # Enter line 7
  end
end
```

4. Present students with example code

```
use_random_seed Time.now.to_i
puts "You enter a room. On the table in this room,
there is a plate of cookies and a single cupcake."
if one_in(2)
  puts "Eating the cookie has given you the power to read people's minds."
  if one_in(2)
    puts "The doctor discovers that the effects are only temporary.
You spend the day resting at home and are back to normal the next day."
  else
    puts "You use this power to find out the answers to your math test.
You get in trouble for cheating, fail the test and can no longer read minds."
  end
else
  puts "The cupcake causes all your hair to fall out."
  if one_in(2)
    puts "The adhesive from the wig causes an allergic reaction to your scalp.
Your hair never grows back and your head becomes too sensitive to cover up
with anything else."
  else
    puts "While wearing the hat, Someone on the street mistakes you
for someone they owe $1000 to. They give you the money and you use it
to buy the latest iPhone. Your hair grows back two days later."
  end
end
```

5. Have students copy and paste the lines from their story from the assignment document into the Sonic Pi template code. Run the code a few times to view the results.

   Encourage them to change the argument to the ***one_in*** functions to change the probability of each outcome.

Wrap Up/Assessment

Have students put a copy of the Nested Conditional template in another buffer.

Expectations

- Add play, sample and sleep functions to make different possible outcomes in each if/else statement.
- Change the probabilities of the one_in functions

Extensions -

- Include single line conditionals within an if or else statement to add more possible outcomes.
- Have probabilities in ***one_in*** functions be chosen randomly

Code should be submitted via Google Doc in Google Classroom

Aim: To use nested conditional statements with specified probability to determine which outcome will be chosen

Open your Choose Your Own Adventure story assignment in Google Classroom

Story 1

– – –

Have the writer of the story read line 1. When given the choice, the other student will flip a coin (Use RANDOM.ORG - Coin Flipper)

Heads - PIck the first choice (line 2)

Tails - Pick the second choice (line 5)

Writer reads the next line. When given the choice, the other student will flip a coin.

Heads - Pick the first choice

Tails - Pick the second choice.

Story 2

– – –

Have the writer of the story read line 1. When given the choice, the other student will roll a die (Use RANDOM.ORG - Dice Roller)

Roll 6 - PIck the first choice (line 2)

Roll not a 6- Pick the second choice (line 5)

Writer reads the next line. When given the choice, the other student will flip a coin.

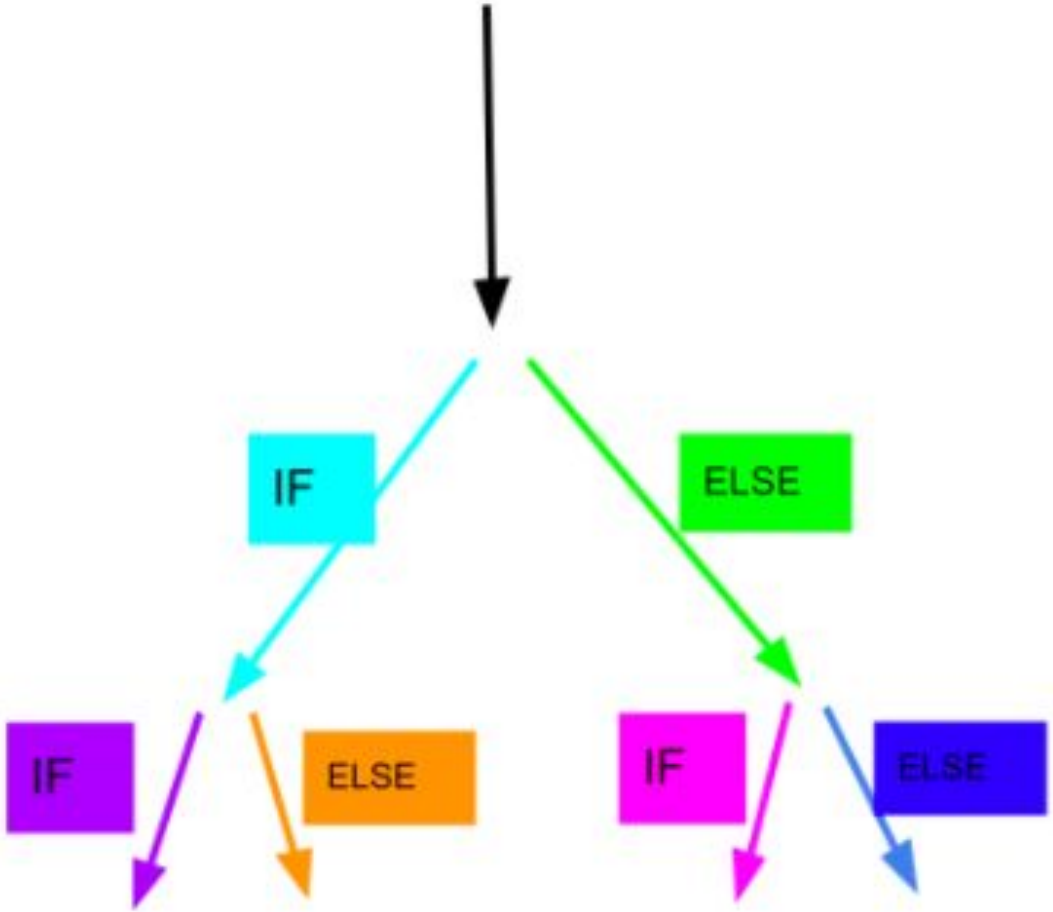
Roll 6 - Pick the first choice

Roll not a 6 - Pick the second choice.

Nested conditionals

Conditionals are like approaching a fork in the road and having to choose a path. However once we have chosen one path we may wind up with another choice later on.

In code, this is called a nested condition.


IF
ELSE
IF
ELSE
IF
ELSE

Probability:

The extent to which something is probable;
the likelihood of something happening or being the case.

When reading your stories with the coin flip, what was the probability of your choosing the first choice?

Students, write your response!

When reading your stories with the dice roll, what was the probability of you choosing the first choice?

Students, write your response!

Copy and paste this code into Sonic Pi and run it.

Part 1

```
 use_random_seed Time.now.to_i
puts # Enter line 1
if one_in(2)
  puts # Enter line 2
  if one_in(2)
    puts # Enter line 3
  else
    puts # Enter line 4
  end
```

Part 2

```
else
  puts # Enter line 5
  if one_in(2)
    puts # Enter line 6
  else
    puts # Enter line 7
  end
end
```

```
 1  use_random_seed Time.now.to_i
 2  puts # Enter line 1
 3  if one_in(2)
 4    puts # Enter line 2
 5    if one_in(2)
 6      puts # Enter line 3
 7    else
 8      puts # Enter line 4
 9    end
10  else
11    puts # Enter line 5
12    if one_in(2)
13      puts # Enter line 6
14    else
15      puts # Enter line 7
16    end
17  end
```

```ruby
use_random_seed Time.now.to_i
puts "You enter a room. On the table in this room,
there is a plate of cookies and a single cupcake."
if one_in(2)
  puts "Eating the cookie has given you the power to read people's minds."
  if one_in(2)
    puts "The doctor discovers that the effects are only temporary.
You spend the day resting at home and are back to normal the next day."
  else
    puts "You use this power to find out the answers to your math test.
You get in trouble for cheating, fail the test and can no longer read minds."
  end
else
  puts "The cupcake causes all your hair to fall out."
  if one_in(2)
    puts "The adhesive from the wig causes an allergic reaction to your scalp.
Your hair never grows back and your head becomes too sensitive to cover up
with anything else."
  else
    puts "While wearing the hat, Someone on the street mistakes you
for someone they owe $1000 to. They give you the money and you use it
to buy the latest iPhone. Your hair grows back two days later."
  end
end
```

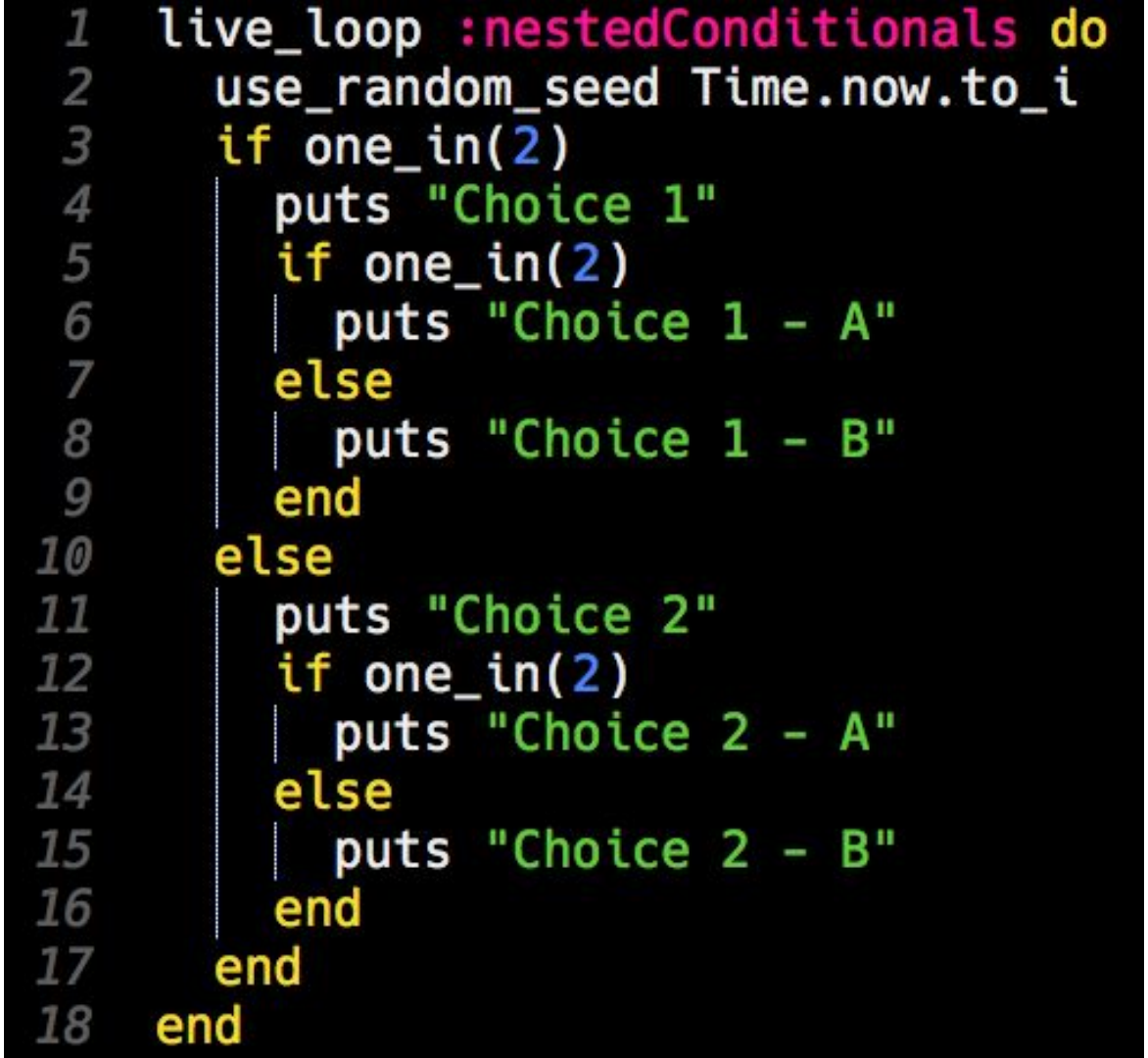
Copy and paste this code into Sonic Pi and run it.

Part 1

```
live_loop :nestedConditionals do
 use_random_seed Time.now.to_i
 if one_in(2)
   puts "Choice 1"
   if one_in(2)
     puts "Choice 1 - A"
   else
     puts "Choice 1 - B"
   end
```

Part 2

```
else
    puts "Choice 2"
    if one_in(2)
      puts "Choice 2 - A"
    else
      puts "Choice 2 - B"
    end
  end
end
```


live_loop :nestedConditionals do
use_random_seed Time.now.to_i
if one_in(2)
puts "Choice 1"
if one_in(2)
puts "Choice 1 - A"
else
puts "Choice 1 - B"
end
else
puts "Choice 2"
if one_in(2)
puts "Choice 2 - A"
else
puts "Choice 2 - B"
end
end
end

Expectations

- Add play, sample and sleep functions to make different possible outcomes in each if/else statement.
- Change the probabilities of the one_in functions

Extensions

- Include single line conditionals within an if or else statement to add more possible outcomes.
- Have probabilities in one_in functions be chosen randomly

Rubric	✅✅✅✅✅	✅✅✅	✅✅	✅
Meeting expectations	Project meets expectations and includes both extensions	Project meets expectations and includes one extension	Project meets expectations but includes no extensions	Project does not meet expectations

Sonic Pi Generative Music Unit Plan
Lesson # 7 - Random Durations of Notes and Samples

Lesson Objectives

Students will be able to use randomized output to affect the duration of notes and samples.

Suggested Duration

1 period (45 minutes)

NYS Computer Science and Digital Fluency Learning Standards

7-8.CT.7
Design or remix a program that uses a variable to maintain the current value of a key piece of information.

Vocabulary

Duration - The amount of time a note or sample will play for or the amount of time the sleep function will last before moving onto the next line of code

Assessments

- Assess ____. Check for the ability to:
    - Store randomized output into a variable to be used later in their code
    - Uses functions which affect the duration of notes and samples
    - Pass variables storing randomized values into different functions

Do Now

Write a line of code that plays a note for longer than one beat.
The length of the note should be chosen randomly.
The length of the randomly chosen note should be different each time.

Reminder: Use attack or release functions to change length of notes.

Possible Solutions

dice function:

```
use_random_seed Time.now.to_i
play 60, release: dice
```

rrand_i function

```
use_random_seed Time.now.to_i
play 60, release: rrand_i(2, 8)
```

Have students share out solutions. When one solution has been present, ask students to raise their hand if they have the same/similar solution. Look for different solutions, all of which are acceptable.

Lesson

Part 1 - Random Note Durations

1. Have students put this code into a live loop, so that each time through the loop the note will play for a different length of time.

   In order to do this, we need to make the sleep value the same as the value passed to the attack or release function.

   Have students turn and talk to discuss how they can do this.

2. Have students share out possible solutions.
   Common Misconception: Using the same function (dice, rrand_i) as a sleep value.
   Remind students that we saw this when working with conditional statements. Calling those functions more than once is like flipping two coins or rolling two dice, not getting different results from the same coin or die.

3. Solution: We need to store the output from these function in a variable which we can use for both the attack/release argument and the sleep value.

```
3  live_loop :duration do
4    use_random_seed Time.now.to_i
5    roll = dice
6    play 60, release: roll
7    sleep roll
8  end
```

Part 2 - Random Sample Durations

1. In a new buffer, have students create a live_loop that plays a sample and sleeps for the duration of that sample

```
1  live_loop :sampleLoop do
2    sample :ambi_choir
3    sleep sample_duration :ambi_choir
4  end
```

2. Have students include the rate function to change the length of the sample. Tell them have the rate of the sample be chosen randomly each time through the loop
   Hint: Use same strategy as for note duration (create a variable to store random value)

   <u>Common Misconception</u>
   Students forget that the sample_duration also needs to include the rate: function
   Because rate changes the duration of the sample

3. Have students share out solutions.

   Possible solution:

```
live_loop :sampleLoop do
  use_random_seed Time.now.to_i
  roll = dice
  sample :ambi_choir, rate: roll
  sleep sample_duration :ambi_choir, rate: roll
end
```

4. Ask students how does rate affect how the sample sounds
   Possible responses: It sounds higher in pitch, it plays faster

5. Point out that numbers passed to the rate: function that are above 1 will cause the duration to be faster (which also affects the pitch).

6. Introduce ***rrand***

   rrand will return a random value between specified range of numbers that includes a decimal point number. This can be used for any range of numbers, but is also useful for choosing random numbers that are less than 1.

   Example Code:

```
live_loop :sampleLoop do
  use_random_seed Time.now.to_i
  t = rrand(0.1, 0.9)
  sample :ambi_choir, rate: t
  sleep sample_duration :ambi_choir, rate: t
end
```

Remind students that when we use numbers that are less than 1 with the rate function, it will cause the duration of the sample to be slower.

7. ***rrand*** can also include negative numbers (remind students that using negative numbers for the rate function will cause the sample to play backwards)

   Example code:

```
live_loop :sampleLoop do
  use_random_seed Time.now.to_i
  dur = rrand(-1, 1)
  sample :ambi_choir, rate: dur
  sleep sample_duration :ambi_choir, rate: dur
end
```

Wrap Up/Assessment

Students may complete one of the two following programs.

1. Use both attack and release with the play function but have different attack and release values chosen randomly and sleep for the correct amount

Example of Finished Code:

```
live_loop :duration do
  use_random_seed Time.now.to_i
  attackRoll = dice
  releaseRoll = dice
  play 60, attack: attackRoll, release: releaseRoll
  sleep attackRoll + releaseRoll
end
```

2. Use a conditional statement to randomly choose between two different samples that play at a randomly chosen rate. The value used for the rate should also be used as the value in the conditional statement

Example of Finished Code:

```
live_loop :sampleLoop do
  use_random_seed Time.now.to_i
  dur = rrand(-1, 1)
  if dur < 0
    sample :ambi_choir, rate: dur
    sleep sample_duration :ambi_choir, rate: dur
  else
    sample :ambi_drone, rate: dur
    sleep sample_duration :ambi_drone, rate: dur
  end
end
```

Rubric: Lesson 7 - Random Durations Assignment Checklist Assessment Tool

Random Durations Assignment Checklist

Each part is worth a total of 2 points.
2 - Accurately completes requirement
1 - Requirement is attempted but not completed accurately
0 - Does not include requirement

Task completed (Circle one) - Note Duration | Sample Rate

_____Uses true randomness

_____Uses method to choose random value(s)

_____Stores random value(s) in variable(s)

_____Correctly uses variable(s) as argument for function which affects duration

_____Sleep function value is correct length based on random value used for duration

_____ = Total Score out of 10

Sonic Pi Generative Music Unit Plan
Lesson #8 - Final Project

Lesson Objectives

Students will be able to will understand requirements and expectations for Generative Music project

Suggested Duration

3 period (45 minutes) + Gallery Walk

NYS Computer Science and Digital Fluency Learning Standards

7-8.CT.4 Write a program using functions or procedures whose names or other documentation convey their purpose within the larger task.

7-8.CT.6 Design, compare and refine algorithms for a specific task or within a program.

7-8.CT.7 Design or remix a program that uses a variable to maintain the current value of a key piece of information.

7-8.CT.8 Develop or remix a program that effectively combines one or more control structures for creative expression or to solve a problem.

Vocabulary

Generative Music: Music that in whole or in part has been created with the use of an autonomous system

Assessments

- Assess ____. Check for the ability to:
  - Incorporate multiple methods of randomization into a program.
  - Explain different types of methods of randomization and how they affect the outcome of their program.

Do Now

Students should read Final Project Assignment Document

Lesson

Part 1 - Intro to Generative Music

1. Introduce concept of Generative Music -
   Generative music is a term popularized by Brian Eno to describe music that is ever-different and changing, and that is created by a system.

2. Play example from “Music for Airports” by Brian Eno
   Brian Eno - Ambient 1: Music for Airports [Full Album]

3. Explain that in this case, the system used was a series of tape machines each playing different loops running at different times and speeds, so that the the results would never repeat the same exact sequence.

Read the following quote “*“The particular piece I’m referring to was done by using a whole series of very long tape loops, like fifty, sixty, seventy feet long. There were twenty-two loops. One loop had just one piano note on it. Another one would have two piano notes. Another one would have a group of girls singing one note, sustaining it for ten seconds. There are eight loops of girls’ voices and about fourteen loops of piano. I just set all of these loops running and let them configure in whichever way they wanted to, and in fact the result is very, very nice. The interesting thing is that it doesn’t sound at all mechanical or mathematical as you would imagine. It sounds like some guy is sitting there playing the piano with quite intense feeling. The spacing and dynamics of “his” playing sound very well organized. That was an example of hardly interfering at all.“ from*
https://reverbmachine.com/blog/deconstructing-brian-eno-music-for-airports/

4. Explain that this method was used before computers were widely available and able to easily create musical output. As computers became more efficient and able to process audio more effectively, they were used to create generative music using programming and algorithms.

Part 2 - Assignment Roll out

1. Go over Assignment sheet.
   - Requirements
   - Random methods discussed in class
   - Grading expectations / Rubric

2. Show students an example project and play them the output.

Example Project Code

```
live_loop :loop1 do
  use_random_seed Time.now.to_i
  if one_in(3)
    n = rrand_i(0, 4)
    sample :elec_beep if n == 0
    sample :ambi_glass_rub if n == 1
    sample :ambi_soft_buzz if n == 2
    sample :elec_cymbal if n == 3
    sample :ambi_piano if n == 4
  else
    n = rrand_i(0, 4)
    sample :glitch_perc1 if n == 0
    sample :glitch_perc2 if n == 1
    sample :glitch_perc3 if n == 2
    sample :glitch_perc4 if n == 3
    sample :glitch_perc5 if n == 4
  end
  if one_in(4)
    sleep 0.75
  else
    sleep 1.25
  end
end

live_loop :loop2 do
  use_random_seed Time.now.to_i
  seq = scale(40, :minor).shuffle
  attackRoll = dice
  releaseRoll = dice
  new_seq = seq.take(dice)
  tick_reset
  new_seq.length.times do
    use_synth :tb303
    play new_seq.tick, attack: attackRoll, release: releaseRoll, amp: 0
    use_synth :sine
    play 28, attack: attackRoll, release: releaseRoll, amp: 0.7
    sleep attackRoll + releaseRoll
  end
end

rain = "/Users/admin/Downloads/mixkit-thunderstorm-in-the-forest-2396.wa

live_loop :loop3 do
  sample rain
  sleep sample_duration rain
end
```

Part 3 - Work Time

Students will be provided 3 class periods to work on this project.

Assignments will be submitted in a Google Doc on Google Classroom along with comments for written requirement of project

Wrap Up/Assessment

Students will have a gallery walk where they will leave their programs running and they can move from one computer to another and spend 2-3 minutes listening to the piece before moving onto the next one.

Task: You will write a program that creates a piece of generative music using different methods of randomization

Requirements for Generative music

- This piece should be ongoing (no definitive end)
- It should slightly change over time
- Changes should be influenced by some type of random events

Code Requirements

- Minimum of 3 Live Loops
- Use of randomization to create different outputs
  - .choose
  - Data structure manipulation
  - Single line conditional statements
  - Nested Conditional statements
  - One_in probability function
  - Random parameters
  - Coin flips/dice rolls (rand_i, dice)
- Use of play and sample functions
- One loop uses sounds from nature (wind, rain, ocean, forest, jungle etc)

Written Requirements

Code should be copied and pasted into Google Doc.

Use commenting in Google Classroom to identify the following:

- What methods of randomization are used
- How random methods affect the outcome of the program

Rubric

	🔥🔥🔥🔥	🔥🔥🔥	🔥🔥	🔥
Implementation of class concepts / Creativity	Code is significantly different from class examples	Code makes several alterations to the class example	Code copies class example with minimal alteration	Code does not include any attempt to use concepts presented in class
Use of different methods of randomization	Accurately uses 3 or more methods of randomization in code	Accurately uses 2 methods of randomization in code	Accurately uses 1 method of randomization in code	Code does not accurately use methods of randomization
Written Explanation of code	Clearly and logically explain how randomization in code is used	Adequately explain how randomization in code is used	Explanation about how randomization in code is used is unclear/requires more detail	Does not explain how randomization in code is used