Jupyter Notebook and Music Generation

COSC 450 Programming Languages

By Noah Campbell, Erica Mbong

Objective

The objective of our project was to make melodies fast through the power of the Jupyter Notebook. To do this, group members set out to understand how to generate melodies automatically using python.

Background for this project

Group members had little experience with Jupyter notebook and its uses for music, so much research and note-taking improved their understanding. Research materials led to the discovery of Music21, a library used to create melodies in Jupyter notebook. Music generation through machine learning was the original goal of the project. However, complexities with machine learning proved too difficult to overcome in the given timeframe. Focusing on something simpler, group members decided to generate music iteratively.

Once the group decided on an idea, members got to work on the code. Group members first hard-coded simple melodies (placing notes of a melody one at a time) to get a basic grasp of Music21. This process was tedious, placing each playable note or chord in a list and looping through that list was easier to work with and yielded surprisingly nice-sounding results.

Team Member Introduction

Noah Campbell was responsible for documentation, source control, and the coding of our project. Not a lot of documentation was "beginner-friendly," so it helped to create our own documentation on unfamiliar programming concepts during research. A repository was also created on GitHub, which allowed group members to commit changes to the latest version of the project.

Erica Mbong managed presentations, research, and coding. By creating presentations that group members can publicly access, it was easier to outline the different parts of what we wanted to present. Research on Muisc21 was helpful in hardcoding melodies during the earlier stages of our melody-generating process.

Testbed setup

Some of the testing materials used by group members included the interface of Jupyter Notebook, midi files, and music21. The next-generation interface of Jupyter Notebook was helpful during the debugging process. When the group ran into an issue, they could run "chunks" of working code, which helped pinpoint problems and reveal solutions. The group tested the sound quality of melodies through Midi files. If something sounded off in the midi file, the generation pattern underwent minor tweaks to create a nicer melody. Music21 is the base of our project as it allows us to create notes, chords, and other musical data for melody creation.

Technologies used

Music21:

As stated, the group used Music21 to define notes, chords, and other musical data. Music21 is an object-oriented approach to analyzing music through symbols or (scores). The toolkit is useful

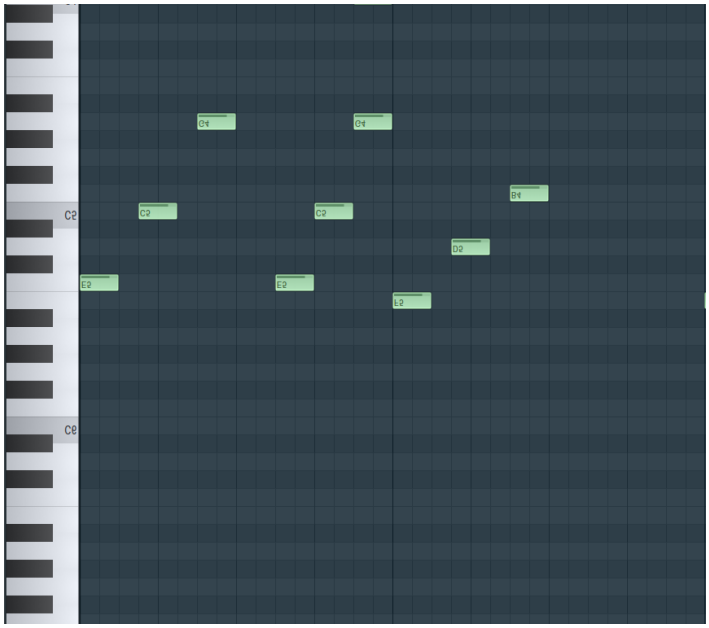
for musicologists with little programming experience or programmers with limited music theory knowledge and access to sophisticated software tools. In an age where symbolic music data is primarily written by hand, music 21 provides reusable tools built from the ground up with object-oriented design throughout.

NumPy:

Earlier in development, NumPy proved useful for making random melodies to test the capabilities of music21. NumPy is a package known for its usefulness in scientific computing with python. There are many uses for NumPy such as arranging elements in an array, but the team was most interested in the random feature of NumPy. Simple melodies were constructed from random notes, and this was a quick way to get musical data up and running.

Fl Studio:

Fl studio was used to visualize music output generated from code. Fl studio is a music creation software used to create songs through virtual instruments that simulate the sound of real pianos, trumpets, etc. By dragging and dropping our midi files into FL studio, team members were able to get a better sense of where each note exists on the keyboard.

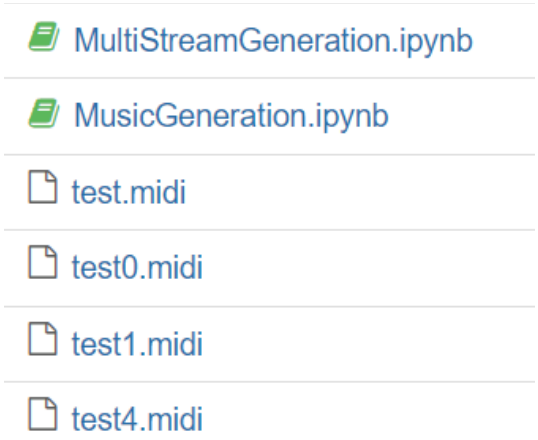


The image above displays the piano roll of FL studio. A piano roll is used for placing individual notes on a song. each green rectangle represents a note that can be played. For purposes of the group, the piano roll is used to display all notes in midi files generated by the music generation program for visualization. The group also used FL studio to test which notes sounded best together.

Midi files:

A MIDI file is a data file that stores the information necessary for music creation. Through MIDI data, musical instruments can play notes. Variables like duration, volume, and pitch determine

how notes sound. The program created by the group makes regular use of MIDI files to store created melodies.



Above is an image of the main programs we used along with their output (The midi files). When clicked each midi file plays back a sound depending on which instruments and other parameters (duration, pitch) are specified.

GitHub:

To make changes accessible to all group members, GitHub was used. The internet hosting service is a popular way to keep track of project versions through the power of Git. Git is a way to keep track of changes made to files and GitHub allows for better file management. Being a website and cloud-based service, GitHub is a reliable way to manage code. Teammates had an easier time managing the growing complexity of project code by making commits (individual changes to file(s)) and documenting why that change was made.

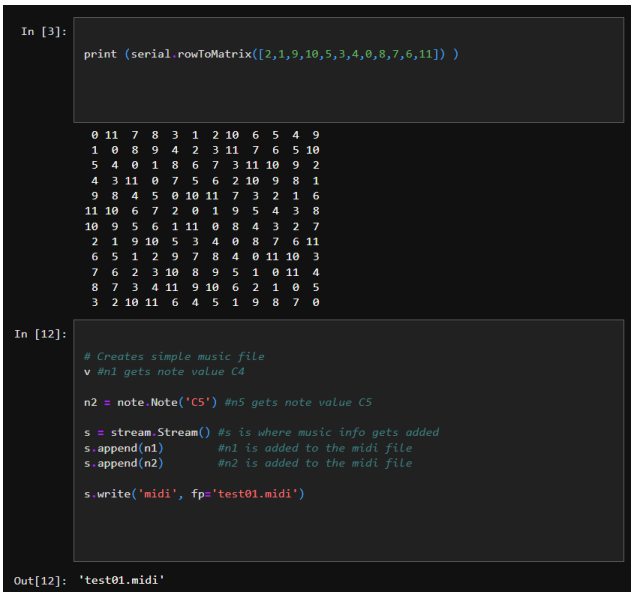
Jupyter Notebook:

Jupyter Notebook was the web application used for coding. Jupyter notebook is good for containing equations, narrative text, and helpful tools for visualizing data. In one instance, the team wanted to visualize MIDI files after the program creates them. Originally, once a file was created, members had to rummage through folders on their computer and locate the exact MIDI file to play. Now, playing melodies is quick and easier to visualize in Jupyter Notebook with the implementation of play sliders.

```
# Run streams concurrently
s3 += s1+s2

# Create file
fileName = 'HappyBirthday.midi'

s3.write('midi', fp=fileName)

# Display file name
print("Music file of "+fileName+":")

# Display create file in Jupyter notebook
EmbedSound(fileName)
```

```
Music file of HappyBirthday.midi:
```

The dark bar in the above image plays the melody created by the program called "HappyBirthday.midi,"

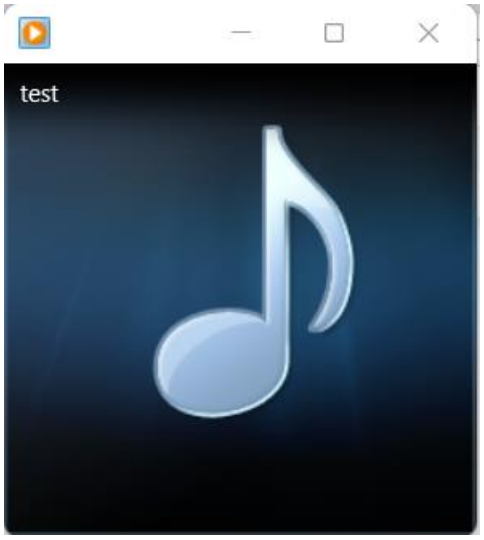


```
In [3]:
print (serial.rowToMatrix([2,1,9,10,5,3,4,0,8,7,6,11]) )

 0 11  7  8  3  1  2 10  6  5  4  9
 1  0  8  9  4  2  3 11  7  6  5 10
 5  4  0  1  8  6  7  3 11 10  9  2
 4  3 11  0  7  5  6  2 10  9  8  1
 9  8  4  5  0 10 11  7  3  2  1  6
11 10  6  7  2  0  1  9  5  4  3  8
10  9  5  6  1 11  0  8  4  3  2  7
 2  1  9 10  5  3  4  0  8  7  6 11
 6  5  1  2  9  7  8  4  0 11 10  3
 7  6  2  3 10  8  9  5  1  0 11  4
 8  7  3  4 11  9 10  6  2  1  0  5
 3  2 10 11  6  4  5  1  9  8  7  0

In [12]:
# Creates simple music file
v #n1 gets note value C4

n2 = note.Note('C5') #n5 gets note value C5

s = stream.Stream() #s is where music info gets added
s.append(n1)        #n1 is added to the midi file
s.append(n2)        #n2 is added to the midi file

s.write('midi', fp='test01.midi')

Out[12]: 'test01.midi'
```

In addition, Jupyter Notebook is great for running code in "chunks." The chunks of code are separated by individual input fields. These fields can run separately from each other. If there were an error in one field, the other fields can still run under the right circumstances. Team members regularly encountered errors when first generating melodies, so running code in chunks allowed working code to run despite other chunks throwing error messages.

Media Player:

Since all members use windows, the windows media player helped play midi files once a midi file was clicked. The media player application was made by Microsoft and can play audio, and video, and allows for image viewing on Windows computers.



Above is a depiction of a media player playing a generated melody “test.”

System Design

**the (.ipynb) file extension in “Program Files” section is for python*

Variables

Location Used	Type	Name	Relationship
MultiStreamGeneration	Stream	s	Used for creating streams.
	Stream	s1	Used for creating streams.
	Stream	s2	Used for storing all streams created so they can run simultaneously.
	Note[]	n11	Contains a list of all possible notes the program can play.
	MidiFile	mf	Variable used to display Midi file in EmbedSound().
	String	fileName	Allows user to put a specific name of the midi file.
MusicGeneration	Stream	s	Used for creating streams.
	Note	n1	Plays notes individually
	Note[]	n2	Contains a list of all possible
	Note	n3	Plays notes individually
	Midi	mf	Variable used to display Midi file in EmbedSound().
	Int[]	Traversal	Determines what notes to play. Stores two values “[x,y]” x represents the first half of the song traversal, y represents the second half
	Int[]	Duration	Stores two values “[x,y]” x represents the length of notes

			in the first half of the song and y represents the length of notes in the second half of the song.
	Int[]	LowPitch	Specifying the lowest range on the keyboard for the melody to play.
	Int[]	MedPitch	Specifying the Med range on the keyboard for the melody to play.
	Int[]	HighPitch	Specifying the High range on the keyboard for the melody to play.
	String	fileName	Allows user to put a specific name of the midi file.

Methods

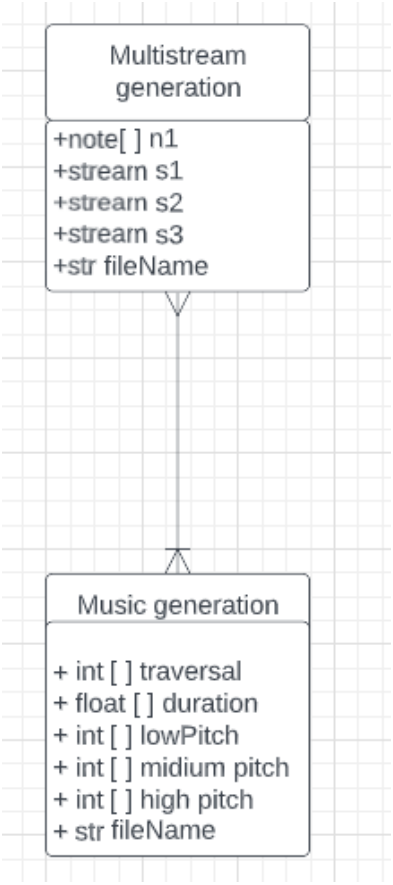
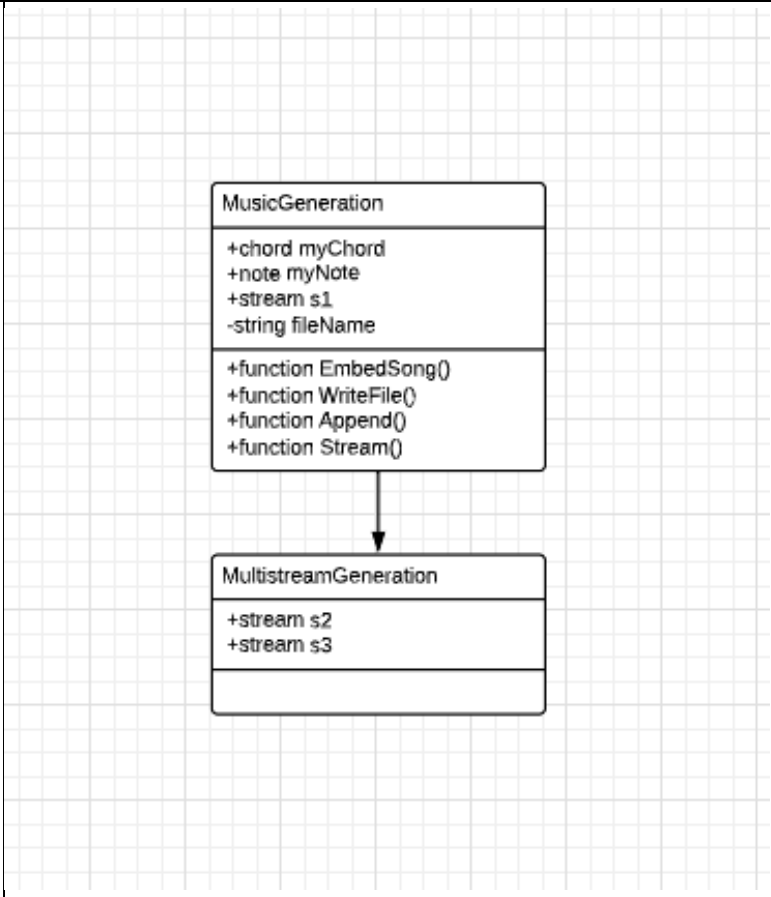
Location Used	Name	Relationship
MultiStreamGeneration & MusicGeneration	EmbedSound()	Presents a play button inside of Jupyter notebook that plays sound when pressed.
	Write()	Writes stream of notes or chords to a Midi file
	Show()	Shows midi file to play in Jupyter notebook

Program Files (.ipynb)

Name	Relationship
MultiStreamGeneration	Tests if multiple streams of musical data can play concurrently
MusicGeneration	Allows for the user to generate their own music through tweaking variables like the pitch and duration of notes.

Flowcharts (UML & E-R Diagrams)

**Our UML and E-R diagrams are simple because we reuse variables in our code.*

E-R Diagram	UML Diagram
	

Future project work

Machine Learning: could be an interesting direction to take the project in the future. To restate, the group originally wanted to create a project related to machine learning but ended up simplifying the project instead.

Visualization: There exists a software capable of visualizing music in Jupyter Notebook, but teammates had a difficult time using it properly. If there was more time, showing the sheet music in Jupyter notebook would be nice to add to the project. Being able to see the sheet music would also be very helpful in helping users see what they are using. This could also be help in creating music faster as you would be able to see what you are making and to locate problems in sounds easily.

Instruments: Music21 has many instruments to choose from when it comes to music generation. Team members experimented with a few instruments but did not delve too deeply. If given more time, members would use a larger variety of instruments to generate music. Implementing more instruments could also evolve the song. This would allow us to play songs that have multiple instruments in them.

Note Duration: Teammates experimented primarily with the lengths of quarter notes, but if given more time, half notes and whole notes would be added as well. Being able to implement songs would also allow for a wide span of to be created at multiple levels of difficulty. This would allow for many different genres to be created.

Conclusion (What we have learned)

Throughout this project, teammates were introduced to the vast capabilities of the Jupyter notebook for music generation. Members enjoyed recreating popular songs like the "happy birthday" song in Jupyter Notebook and gained a deeper understanding of python in the process. This project exposed us to deeper concepts of python which we will continue to learn. This project also gave us a better understanding of machine learning. Group members will continue their research on music generation outside of the classroom to further their understanding.

Sources:

http://midi.teragonaudio.com/tutr/midiform.htm#:~:text=What's%20a%20MIDI%20file%3F,MIDI%20file%20contains%20musical%20information.

https://archives.ismir.net/ismir2010/paper/000108.pdf

https://kinsta.com/knowledgebase/what-is-github/