Functional testing:

Tests with positive result:

- Positive testing:
  - Arduino plays the previously declared sequence correctly. Declared sequence is the same format as sequence generated with Python script.
  - Python script created in order to send processed MIDI notes to Arduino works properly on a declared data.
  - Uploading files with the wrong format results in “Enter valid file!” comment in Python console
  - Uploading silent file generates no files/arrays with midi notes and gain

- Negative Testing:
  - Python script analyzes frequencies between 0 - 22 [kHz] but for spectrogram it shows freq scale between 0 - 1k [kHz] for the clarity
  - Audio volume is neutral for program performance
  - Audio files can be relatively long, e.g 6 min
  - The script works properly for multiple clean computer-generated frequencies and clean digital instruments

Tests with negative result:

- Positive testing:
  - Arduino script fails at processing received data with MIDI notes, therefore, it cannot be played instantly after executing Python code.
  - Blurred audio files aren’t analyzed properly (Python script pull out whole range frequencies with highest gain
- Negative testing:
  - Script doesn’t work properly on pitch-bended frequencies and recorded audio files containing larger amounts of noise.

Non-functional testing:

- Long audio files don't slow down Python script in any noticeable way. Time required for creating a complete analysis is short. For 6 min piano audiofile it takes about 1.5 min to analyze.
- Frequency resolution is 5.38 [Hz] so it is satisfying for all notes above 82 [Hz] E2.
  Time resolution is 0.05 [s] and it is enough.