N1 O2 T3 E4 S5

Musicalizador

TRABAJO FINAL

ABIGAIL JUSTINIANO - DJUSTINIANO@UDESA.EDU.AR

CAMILA GUERRERO - CGUERRERO@UDESA.EDU.AR

TEO KAUCHER - TKAUCHER@UDESA.EDU.AR

FLORENCIA LERDA - FLERDA@UDESA.EDU.AR

Objetivos del trabajo

El objetivo del trabajo fue realizar un Musicalizador, que a partir de una partitura pueda sintetizar notas predefinidas en archivos txt.

Y, además, establecer conexión con un instrumento real llamado metalófono.

Desarrollo del trabajo

El trabajo fue enriquecedor porque logramos poner a prueba todo lo aprendido en la materia.

Dentro del trabajo se declararon los siguientes módulos:

- ModulatorFunctions: En este módulo se encuentran declaradas las funciones base para darle forma a las notas musicales.
- Main: se encarga de interpretar lo ingresado por consola y ejecutar el programa
- Mod: Es un módulo que contienen las funciones de modulación.
- Instrument: Se encarga de leer e interpretar el file de los instrumentos disponibles.
- Note: Se encarga de leer e interpretar los archivos de notas y obtener la frecuencia, duración, etc.
- Synthesizer: Es un módulo que se encarga de generar las señales y el archivo wav

Errores inesperados con su solucion

- Cuando generábamos el archivo .wave la canción sonaba distorsionada. El error se debía que nuestra función que se encargaba de generarlo recibía por parámetro una cuenta que estaba mal, por lo que se volvió a calcular y funcionó.
- La canción no respetaba los tiempos que se pasaban por parámetro debido a que sumábamos los Arrays. Para solucionarlo, implementamos un Array de ceros.
- Surgió un nuevo error a raíz del anterior, el cual cuando pasamos canciones, por ejemplo: ‘queen’. Surge el error de que no puede sumar las señales porque tienen tamaños diferentes.
- Tuvimos error en argparse dentro del main, porque al querer ejecutarlo declaraba que no permite documentos con ‘.wav’ en el nombre. La solución fue cambiar el nombre.
- Cuando el programa se ejecuta, el usuario puede ingresar el nombre del archivo sin el ‘.wav’ ya que fue una entrada inesperada y solucionada.

No logramos realizar la modulación por falta de tiempo.

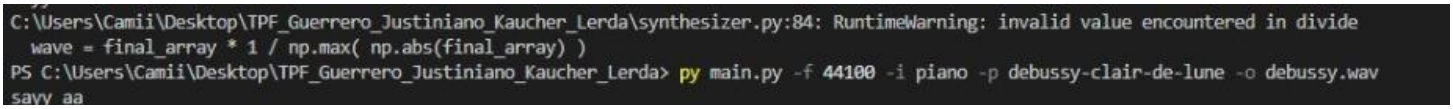
Pasos de ejecucion

Para la ejecución del programa se debe instalar lo siguiente:
Pip install mido
Pip install scipy

Y por último hay que ingresar los datos necesarios con el formato que está en la consigna

```
$ ./sintetizador [-f <frecuencia>] -i <instrumento> -p <partitura> -o <audio.wav>
> sintetizador [-f <frecuencia>] [-i <instrumento>] -p <partitura> -o <audio.wav>
```

En la imagen se representa la frecuencia que sería la frecuencia de muestreo, instrumento que son aquellas disponibles carpetas instruments, partitura son las disponibles en la carpeta scores, y audio.wav es el nombre con el que se va a guardar el archivo.wav de la canción.



Bibliografia

- https://www.youtube.com/watch?v=cdblJqEUDNo&t=2s
- https://thispointer.com/numpy-where-tutorial-examples-python/
- https://www.youtube.com/watch?v=iNmtSnb7TlQ
- http://elclubdelautodidacta.es/wp/2012/08/calculo-de-la-frecuencia-de-nuestras-notas-musicales/
- http://work.thaslwanter.at/sksound/html/
- https://www.youtube.com/results?search_query=test+python+visual+estudio+code
- https://www.geeksforgeeks.org/how-to-plot-a-smooth-curve-in-matplotlib/
- http://work.thaslwanter.at/sksound/html/