BINUS UNIVERSITY
BINUS INTERNATIONAL

Assignment Cover Letter

(Individual Work)

Student Information:		**Surname**	**Given Names**	**Student ID Number**
	1.	**Melvern**	**Marco**	**2201798351**

Course Code	**: COMP6502**	**Course Name**	**: Introduction to Programming**
Class	**: L1BC**	**Name of Lecturer(s)**	: 1. Monica Hidajat
Major	**: CS**		
Title of Assignment (if any)	: Pitch Range Checker		
Type of Assignment	**: Final Project**		
Submission Pattern			
Due Date	**: 21-11-2018**	**Submission Date**	**: 21-11-2018**

The assignment should meet the below requirements.

1. Assignment (hard copy) is required to be submitted on clean paper, and (soft copy) as per lecturer's instructions.
2. Soft copy assignment also requires the signed (hardcopy) submission of this form, which automatically validates the softcopy submission.
3. The above information is complete and legible.
4. Compiled pages are firmly stapled.
5. Assignment has been copied (soft copy and hard copy) for each student ahead of the submission.

Plagiarism/Cheating

BiNus International seriously regards all forms of plagiarism, cheating and collusion as academic offenses which may result in severe penalties, including loss/drop of marks, course/class discontinuity and other possible penalties executed by the university. Please refer to the related course syllabus for further information.

Declaration of Originality

By signing this assignment, I understand, accept and consent to BiNus International terms and policy on plagiarism. Herewith I declare that the work contained in this assignment is my own work and has not been submitted for the use of assessment in another course or class, except where this has been notified and accepted in advance.

Signature of Student: (Name of Student)

1. Marco Melvern

Pitch Range Checker

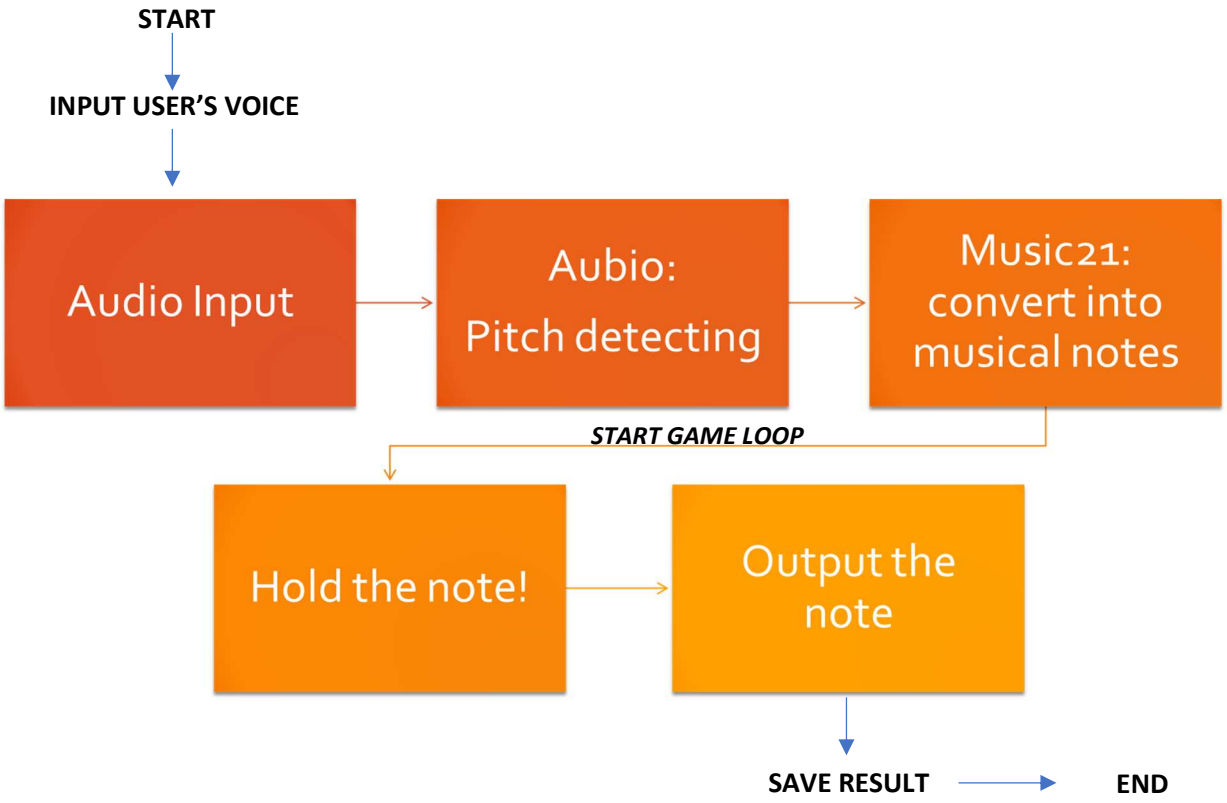
Name : Marco Melvern

ID : 2201798351

Description

Simply speaking, this program is meant to record the user's voice to capture its lowest vocal pitch and its highest vocal pitch, then records it afterwards. It uses an input device to record the incoming voice and what makes it unique is the fact that it uses Pygame module to function; one fact which is not frequently seen on other stereotype projects. Furthermore, everyone has its own vocal range, thus every result would be heterogenous according their own voice.

Design



Discussion

IMPLEMENTATION:

The modules implemented in this project are: Pygame, Aubio, Music21, as well as Cents.

Pygame is a widely-used module in Python and is known by most programmers both experienced and first-timers as a beneficial tool to create executables as well as games. Its sheer simplicity and diverse functionality makes this specific module chosen as the main framework in this project.

Aubio is a beneficial pitch detection framework used to capture voice recorded by the input device then convert it into a frequency in Hertz. It is an influential yet important module as it relieves the need to convert complicated codes into frequencies.

Finally, Music21 is a musical framework usually used by musicians to read frequencies and converting it into a specific musical note. Collaborating with Cents, it handles equations being used to convert frequencies obtained into musical notes. Cents helps to measure in between half steps by using 1/100 of a half step as a counting formula. Using cents, it could tell how far away a note if from the “perfect pitch”, or the ideal sound for our note.

HOW IT WORKS:

The simplified version of how it works are explained from the simple flow chart in section two. This is the detailed version.

Firstly, the user must execute the file via Python in the command prompt. The reason behind this is the fact that the user must choose an input device to be used as the microphone to record voices, since every computer has its own input devices. If the user did not input an input device, the program will automatically list all the available input devices built in the computer and asks the user to choose and input the listed devices.

Afterwards, the main screen would appear with a simple button. After the button is being clicked by the mouse, the main program starts. The program starts to open the microphone stream enabling voice to be recorded. Then, it calibrates the microphone with the environment. At the same time, the user is asked to sing the lowest note he/she could sing. Bear in mind that to achieve perfect result, consistency is a major factor in this scenario. Hence, stable voice is mandatory to be recorded.

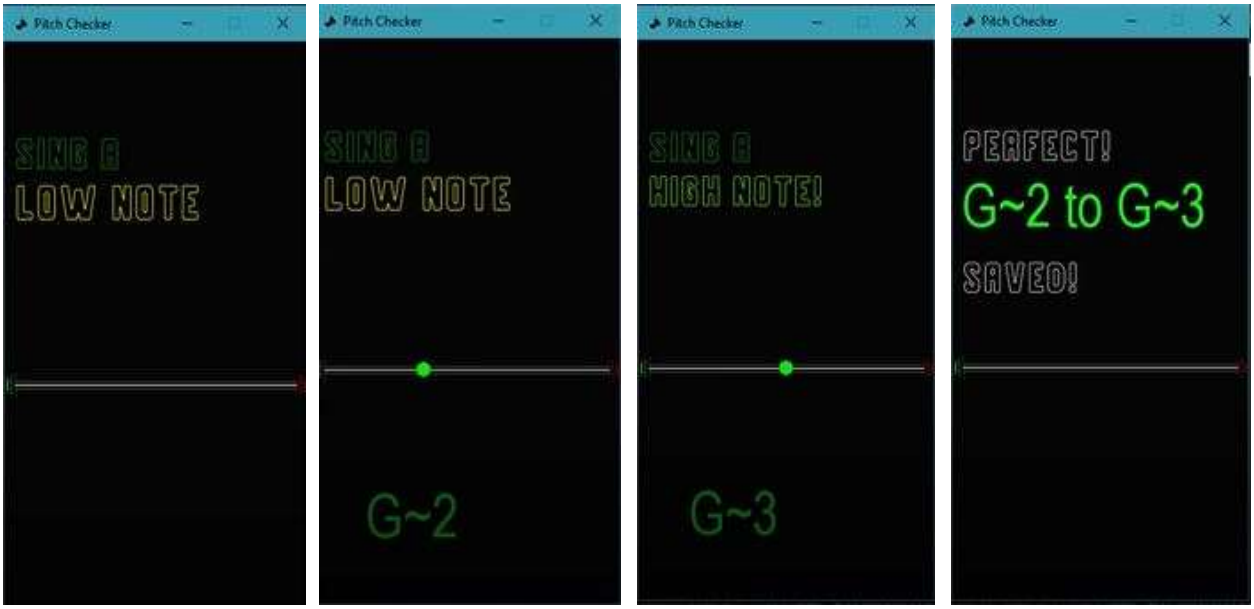
Whilst the user is singing its lowest voice, the program quickly recorded the voice and converting it into a frequency in Hertz then converting it again into a musical note. The program will also animate the result of the voice into the screen. Green marks will appear if the notes being sung are accurate and sharp, red marks will appear when the notes are choppy or too loud and inaccurate. The program will eventually compare the samples previously recorded into the data with the samples being recorded at the time being. Hereafter, if the samples obtained have the same frequency and musical note for ten times in a row, the program would record the specified musical note.

After the program obtained the user's lowest note, it will eventually ask the user to sing its highest note. Whilst the user sings its highest note the program would perform the same execution as it did on the lowest note, though this time it would check whether the user's high note is not lower than its low note. The program would not close and stop if it does not find an accurate note. Ten same samples in a row will also be taken by the program to validate the user's highest note. After both low note and high note are obtained, the program will record the highest musical note being sung and compiles them into a text file. Ultimately saving the data of the vocal pitch range.

CLASS EXPLANATION:

The settings class inside the program made sure of the program to run privately and without intervention from third-parties. Meaning that the variables within the class would be the same throughout the program execution and could only be read and not overwritten.

Evidence



Resources

Kaiser, K. (2017, December 7). *Building a Vocal Range Detection Program in Pygame and Music21*. Retrieved on November 19, 2018, from https://www.makeartwithpython.com/blog/vocal-range-python-music21/