Project Proposal

Project Description

PyKeys
PyKeys is a game with preloaded levels of difficulty where bubbles fall onto their corresponding keys on a piano. The user must hit the keys on time for the score to increment. The game has two additional features, a music composition mode wherein the user can input notes saved into a file loaded as a level for the user to play, and an algorithm that analyzes where the user makes the most errors, allowing the user to practice the section again.

Competitive Analysis

Similar projects include 'Synthesia,' a computer program that allows users to play on a MIDI Keyboard. This project is identical to 'Synthesia' because it has the composition feature and tracks the score. One difference is that while 'Synthesia' pauses the piece when a wrong note is pressed (until corrected), PyKeys provides a performance analysis after the music is played and allows the user to practice the section of the piece with the most errors.

Structural Plan

Parts of the project: Base Piano, Falling Bubbles, Music Composition, Analyze Mistakes

Base Piano
There exist two functions that help draw the piano: drawKey, and drawFlatsandSharps. To illustrate each of the 16 keys, we use a for loop getting the key bounds for each key. Upon drawing the keys, we call the drawFlatsandSharps function, which receives the bounds and draws the flats/sharps in their appropriate areas. In the mousePressed function, we label the key and check if the x and y coordinates of the mouse press are within the range of the keys or sharps and play sounds accordingly.

Falling Bubbles
This is done by utilizing timer fired. In this function, we call a function called move falling bubbles which increments the y coordinates. Meanwhile, redrawAll draws the bubble using the new y coordinates each time. To ensure that the bubble falls on the right key, we gather the x coordinates from indexing into the base piano notes and get the x cell bounds accordingly.

Music Composition
This has several features: naming the composition, listening to the composition, saving the composition, and playing it in the game mode. This is done through a series of file handling procedures. First, we store the composition into a new file in the appropriate format, and then we append the name of the composition to the songs text file. Upon returning to the home screen and selecting the game mode, the composition shows up as a selection for the game.

Analysis of Mistakes
This function is supposed to check the part of the song where the user makes the most mistakes and accordingly allow the user to practice that section again.

Algorithmic Plan

Complex Ideas in code: Music Composition, Analysis of Mistakes

Music Composition
This will require a function that converts the composition into a file, so when the user chooses to save their song, it gets saved accurately. The function will need to open a new file under a name provided by the user and then loop through the user's composition, appending the notes in the correct format to a list. Upon doing this, the function needs to write these notes to the file.

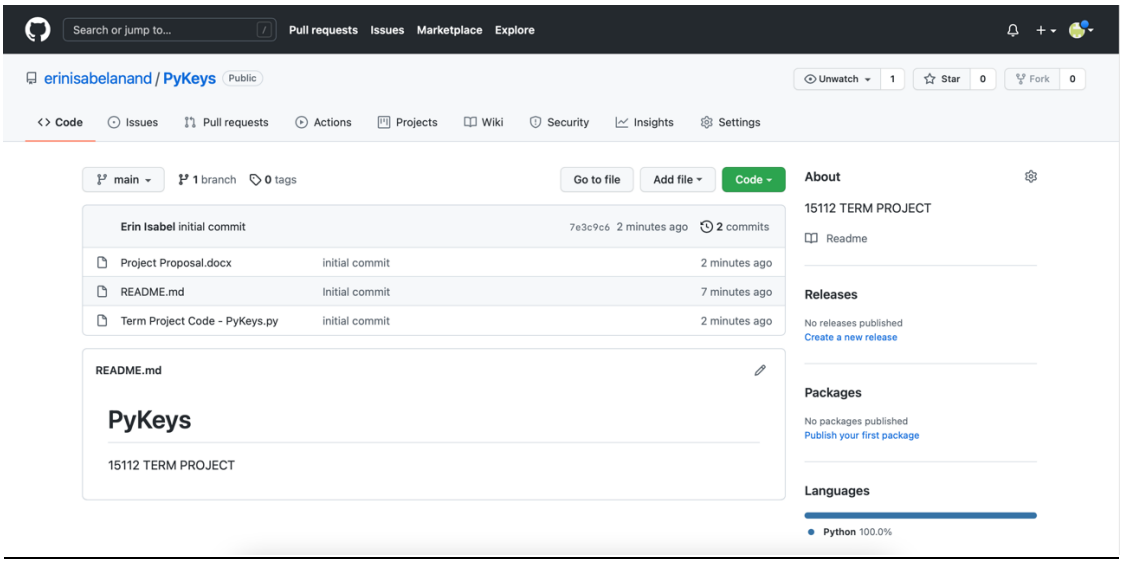
Analysis of Mistakes
To do this, we need to keep sectioning the song into pieces. While the game is being played, we will also have to keep a tab on where the user is making the most mistakes. This could be done by accessing a 2D list, storing whether a note was played accurately or not, and finding which section had the most cumulative wrong notes. Then, we need to create a feature that allows the user to practice that section repeatedly until the number of mistakes reduces to a minimum.

Project Proposal

Timeline Plan

Date	Plan
16-Nov to 17-Nov	- Make the sharps and flats functional - Add a scoring system - Work on making a music composition mode
18-Nov	- Continue to work on making the music composition mode
19-Nov to 20-Nov	- Transform user input from the music composition game and implement it in the code
21-Nov to 22-Nov	- Start working on the analysis of the user's mistakes
23-Nov	- Improve UX
24-Nov to 28-Nov	- Improve UX
29-Nov to 1-Dec	- Maybe add a 2.5D effect / improve UX

Version Control Plan



1.0 Screenshot from GitHub of the repository

I created a repository on GitHub, which I plan to update daily, or whenever I make a significant change/addition to my program.

```
(base) eanand:TERM PROJECT erinisabel$ cd PyKeys/
(base) eanand:PyKeys erinisabel$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(base) eanand:PyKeys erinisabel$ git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Project Proposal.docx
        Term Project Code - PyKeys.py

nothing added to commit but untracked files present (use "git add" to track)
(base) eanand:PyKeys erinisabel$ git add -A
(base) eanand:PyKeys erinisabel$ git status
On branch main
Your branch is up to date with 'origin/main'.
```

1.1 Method of updating repository

Project Proposal

Module List

Musical Beeps – This allows for the generation of piano tones