PYTHON PROGRAMMING AND ITS ENGINEERING APPLICATIONS

Student Name : ABID ALI

Student ID : 2023280099

Student Email : abiduu354@gmail.com / abidali@mail.nwpu.edu.cn

Professor’s Email: leetao@nwpu.edu.cn

Date of Submission : 2023-4-01

Assignment 1 : Calculator

Description

A simple and scientific Calculator written in PyQt5. It has also scientific mode functions (sin, cos, log, ln). View the answer in LCD display like widget.

Installation

First Install dependencies...

- python3
- python3-pyqt5

Libraries

- import os, sys, re
- from math import *
- from PyQt5.QtCore import QEventLoop, QTimer
- from PyQt5.QtWidgets import QApplication, QMainWindow, QAction, QActionGroup, QButtonGroup

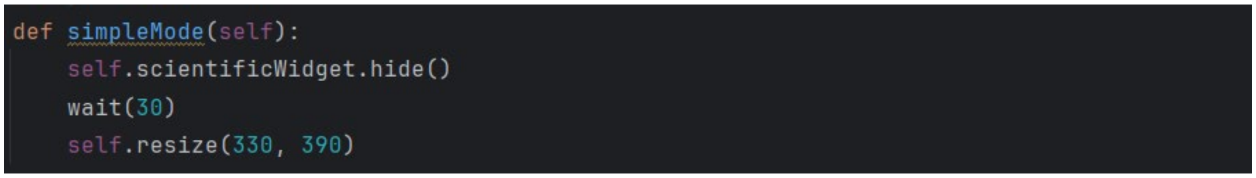
Code Snippets

The provided code is a simple calculator application implemented using PyQt5 in Python. It includes basic arithmetic operations as well as scientific functions such as square root, cube root, trigonometric functions, logarithms, and constants like pi and e.

Here's a summary of the functionalities:

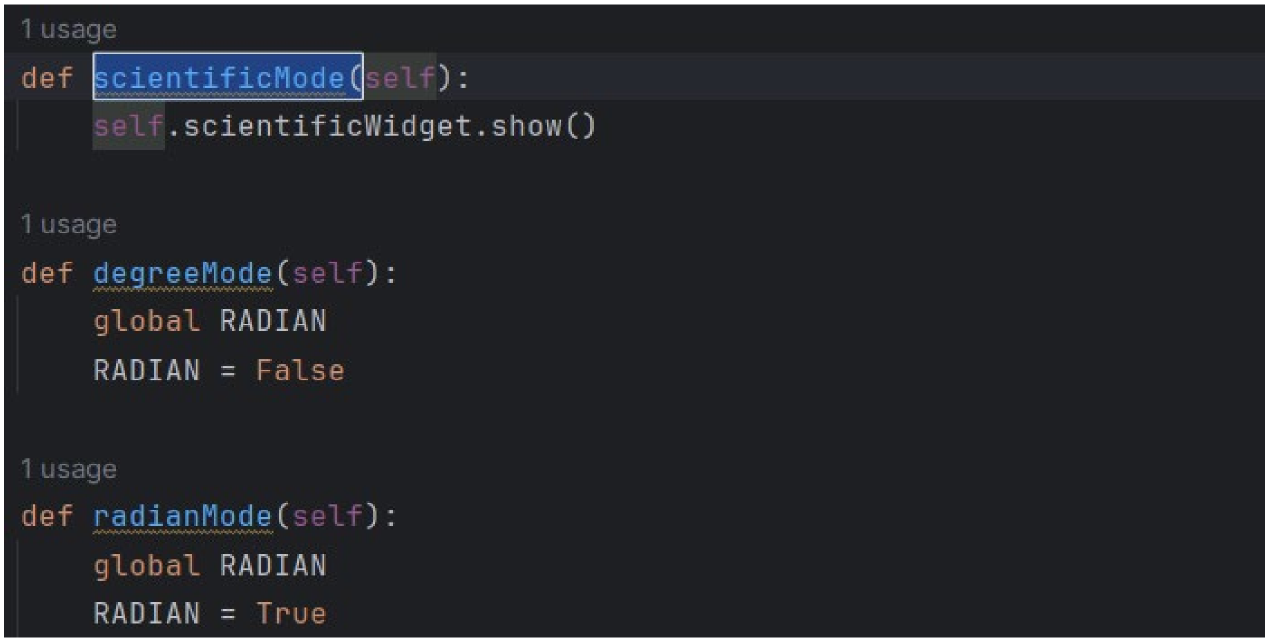
- The calculator supports both simple and scientific modes.

- In simple mode, basic arithmetic operations (+, -, *, /) can be performed along with parentheses for grouping.



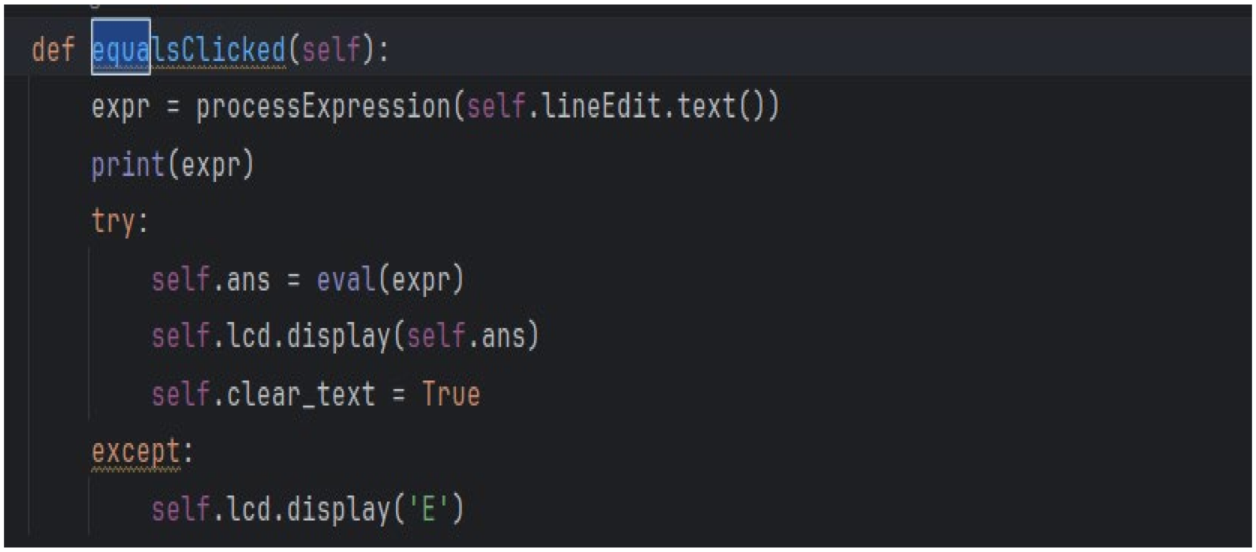
```
def simpleMode(self):
    self.scientificWidget.hide()
    wait(30)
    self.resize(330, 390)
```

- Scientific mode includes additional functions such as square root, cube root, trigonometric functions (sin, cos, tan), logarithms (log and ln), and constants (pi and e).



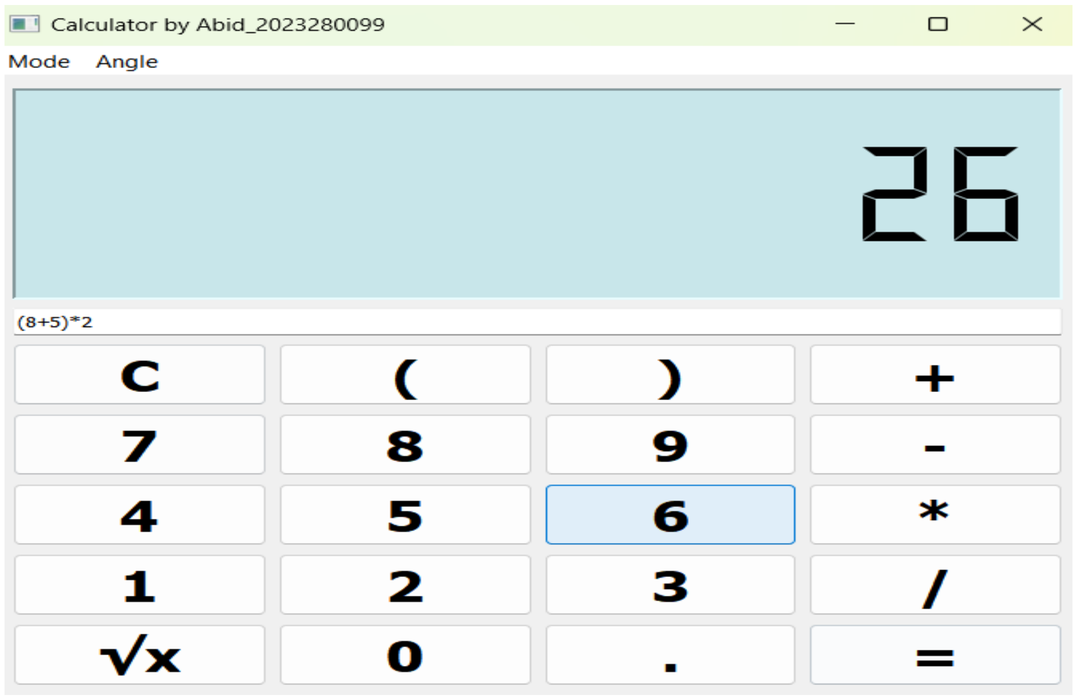
```
1 usage
def scientificMode(self):
    self.scientificWidget.show()

1 usage
def degreeMode(self):
    global RADIAN
    RADIAN = False

1 usage
def radianMode(self):
    global RADIAN
    RADIAN = True
```

- The calculator interface consists of buttons for numeric input, operations, and scientific functions.

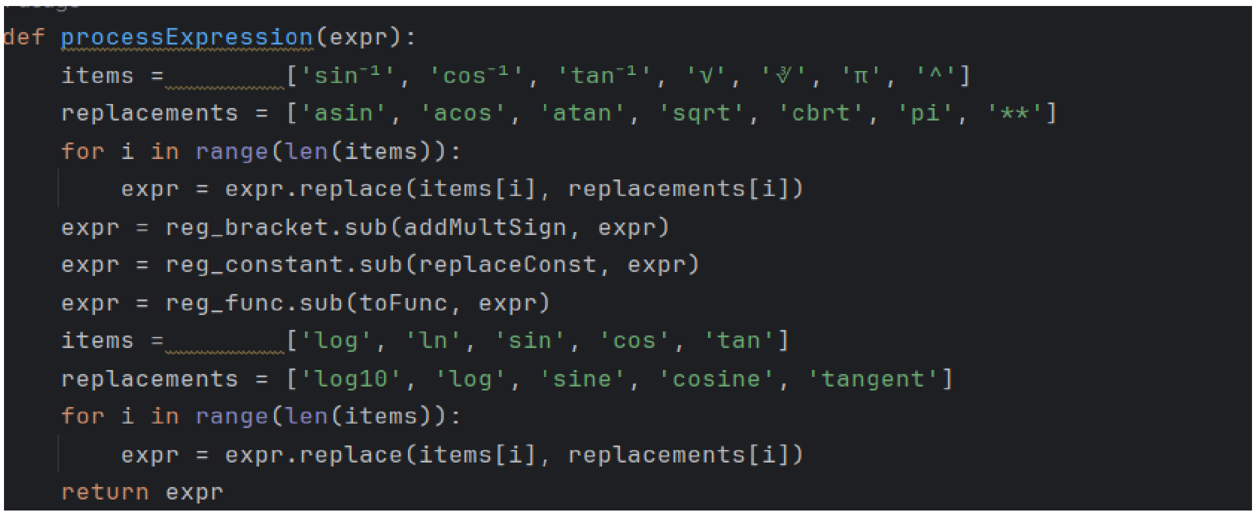


```
def equalsClicked(self):
    expr = processExpression(self.lineEdit.text())
    print(expr)
    try:
        self.ans = eval(expr)
        self.lcd.display(self.ans)
        self.clear_text = True
    except:
        self.lcd.display('E')
```

- The equalsClicked method evaluates the expression entered in the line edit widget and displays the result in the LCD display.



- Various keyboard shortcuts are provided for ease of use, such as backspace for deleting characters, Enter for evaluating expressions, and 'A' for inserting the previous answer.

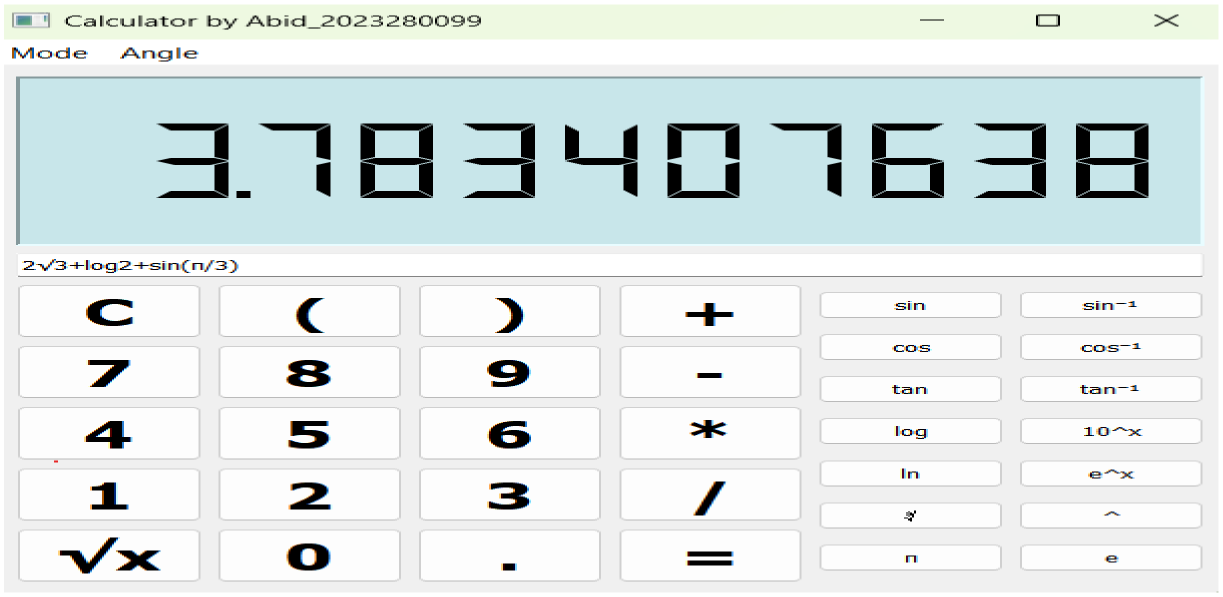


```
def processExpression(expr):
    items = ['sin⁻¹', 'cos⁻¹', 'tan⁻¹', '√', '∛', 'π', '^']
    replacements = ['asin', 'acos', 'atan', 'sqrt', 'cbrt', 'pi', '**']
    for i in range(len(items)):
        expr = expr.replace(items[i], replacements[i])
    expr = reg_bracket.sub(addMultSign, expr)
    expr = reg_constant.sub(replaceConst, expr)
    expr = reg_func.sub(toFunc, expr)
    items = ['log', 'ln', 'sin', 'cos', 'tan']
    replacements = ['log10', 'log', 'sine', 'cosine', 'tangent']
    for i in range(len(items)):
        expr = expr.replace(items[i], replacements[i])
    return expr
```

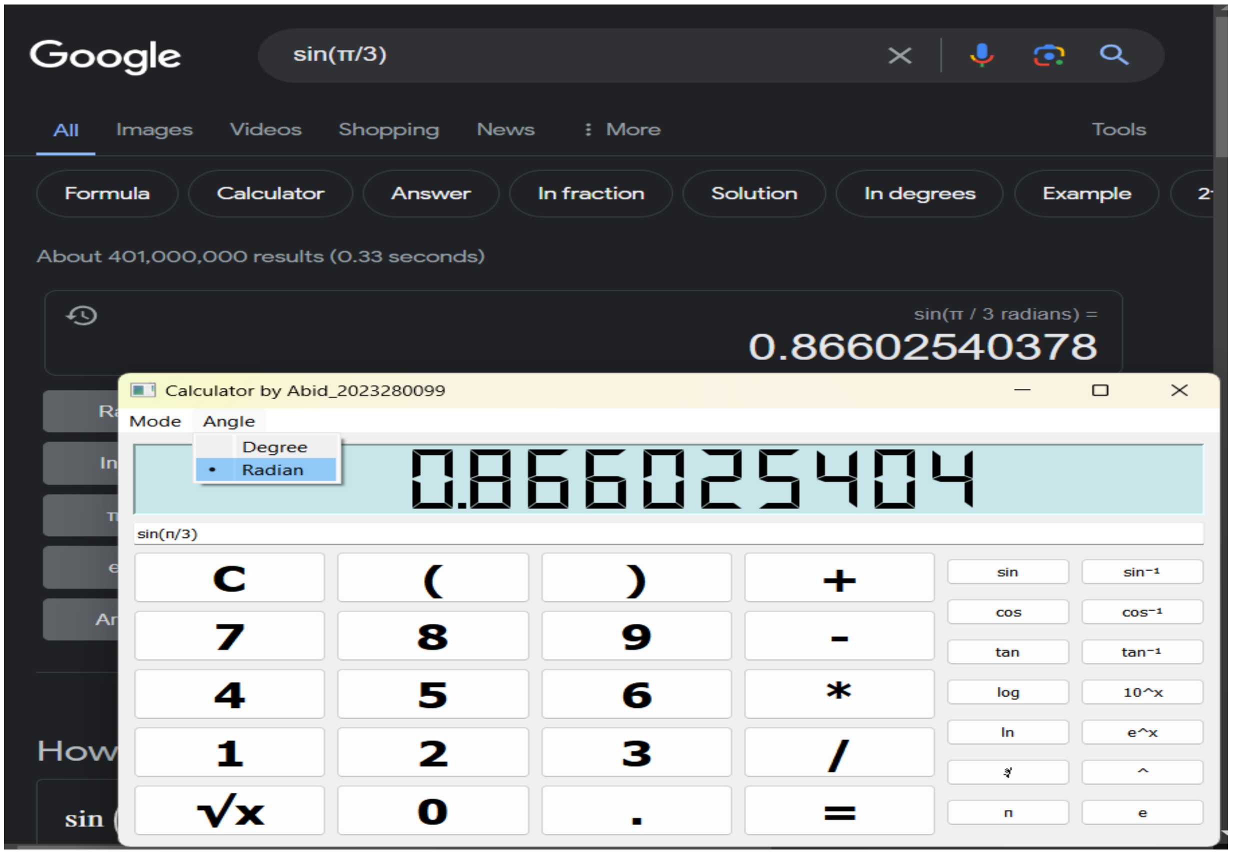
Overall, this code provides a functional calculator application with both basic and scientific functionalities, implemented using PyQt5 for the graphical user interface.

Calculator Screenshots

1) Scientific Calculator GUI

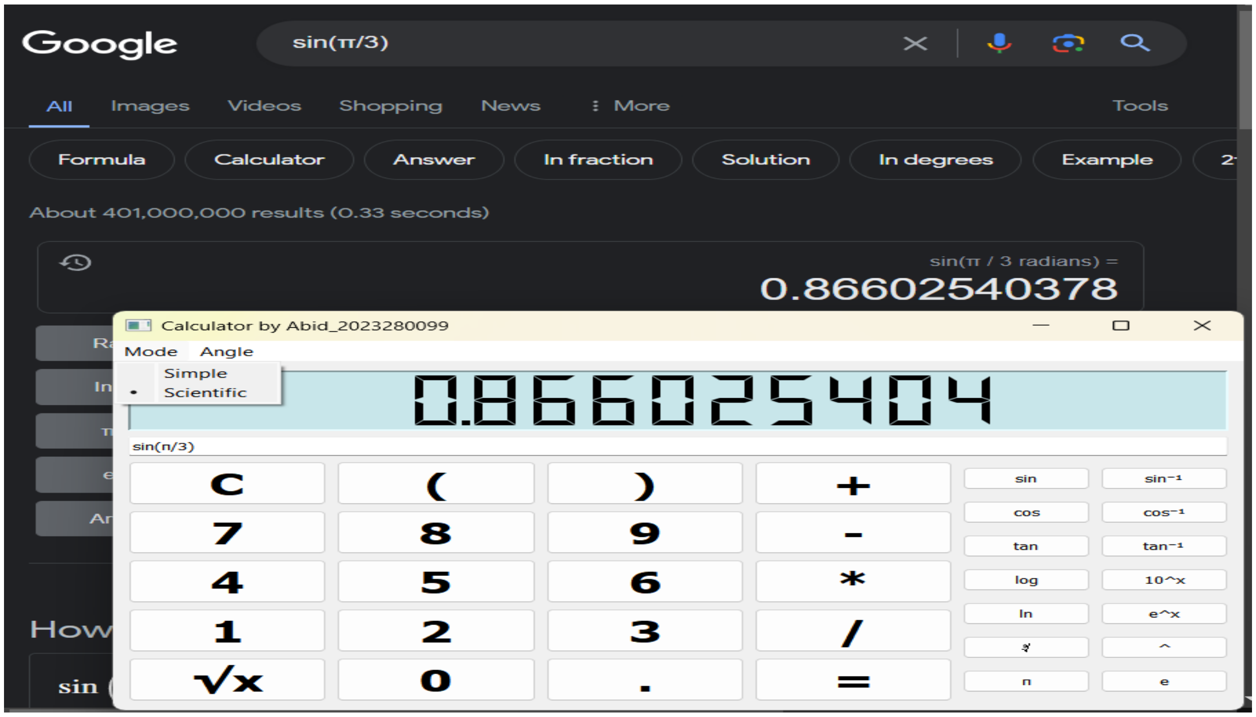


2) Scientific Calculator with Angle Degree/Radian

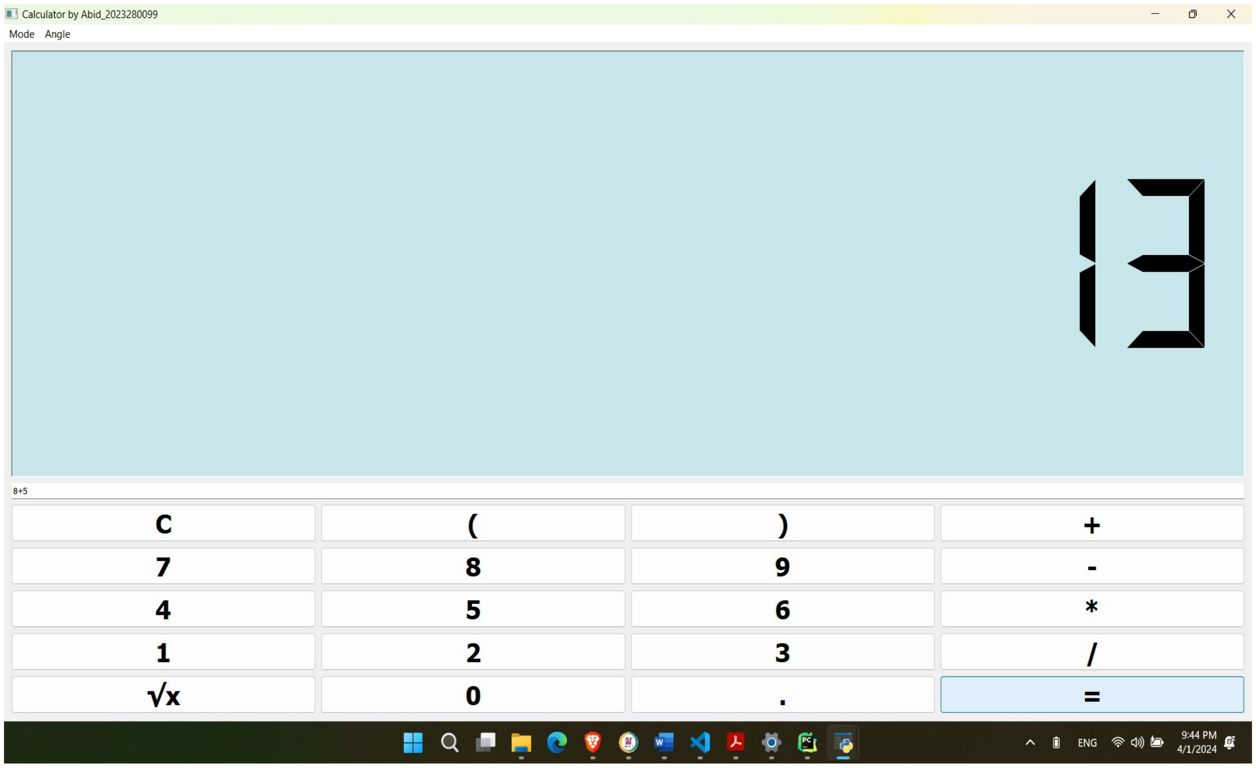


Google
sin(π/3)
All Images Videos Shopping News More Tools
Formula Calculator Answer In fraction Solution In degrees Example
About 401,000,000 results (0.33 seconds)
sin(π / 3 radians) =
0.86602540378
Calculator by Abid_2023280099
Mode
Angle
Degree
• Radian
0.866025404
sin(π/3)
C ( ) +
7 8 9 -
4 5 6 *
1 2 3 /
√x 0 . =
sin sin⁻¹
cos cos⁻¹
tan tan⁻¹
log 10^x
ln e^x
^
π e

3) Mode Switching



4) Simple Calculator GUI



Assignment 2 : Chatroom using pyqt5

Description

This code presents a basic graphical user interface (GUI) chat application using PyQt5, facilitating communication between a client and a server through sockets. The application window consists of a text edit widget to display chat messages, a line edit widget for the user to input messages, and a 'Send' button to transmit messages to the server. A thread is utilized to continually receive messages from the server and exhibit them in the text edit widget.

Code Explanation

This code exemplifies a graphical user interface (GUI)-based chat application utilizing PyQt5, which is a widely used Python binding for the Qt library. The chat application enables users to exchange messages with a server, facilitating both sending and receiving messages seamlessly.

The code consists of two main classes: Window and ClientThread.

The Window class inherits from the QDialog class and creates the GUI of the chat application. The class has the following attributes:

The \`Window\` class contains a method named \`send\`, which is connected to the \`clicked\` signal of the \`btnSend\` widget. When triggered, the \`send\` method retrieves the text entered in the \`chatTextField\` widget, formats it appropriately, and then adds it to the chat widget for display. Additionally, the method transmits the message to the server via the \`tcpClientA\` socket.

The \`ClientThread\` class is derived from the \`Thread\` class and serves to instantiate a thread responsible for establishing a connection with the server and handling the reception of messages from it. This class is equipped with the following attributes:

- window: The \`window\` attribute denotes an instance of the \`Window\` class, which is passed to the \`ClientThread\` instance upon its instantiation. This facilitates communication between the \`ClientThread\` and the GUI window, enabling the thread to update the user interface and manage message display within the chat application.

In the `ClientThread` class, the `run` method is implemented to perform the following tasks:

1. Create a TCP socket.

2. Connect to the server using the specified host and port (in this case, host is typically the server's IP address or domain name, and port 80 is the standard HTTP port).

3. Continuously receive messages from the server.

4. Display the received messages in the chat widget of the `Window` instance.

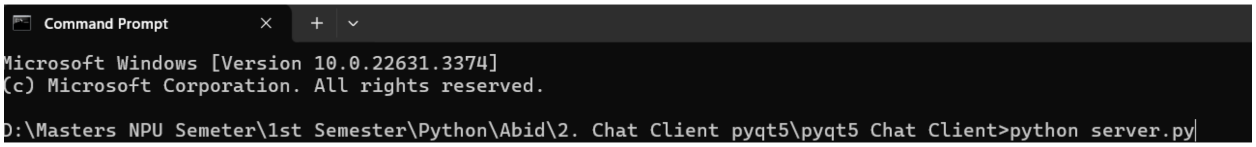
The code also has a main block that creates an instance of the

QApplication class, the Window class, and the ClientThread class. The ClientThread instance is started and the exec method of the Window instance is called to display the GUI.

Finally, the code uses the sys.exit method to exit the application when it's closed.

Chatbot Screenshots

1) Set the Server



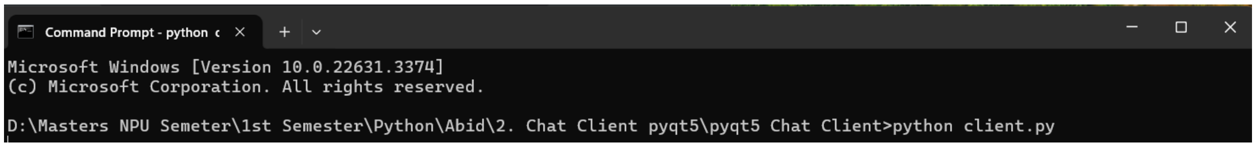
```
Microsoft Windows [Version 10.0.22631.3374]
(c) Microsoft Corporation. All rights reserved.

D:\Masters NPU Semeter\1st Semester\Python\Abid\2. Chat Client pyqt5\pyqt5 Chat Client>python server.py
```

There are two ways to start the program

(1) Typing in the cmd (command prompt) **python server.py** then server start.

(2) Another way by opening the project in IDE then press the play button.

2) Set the Client



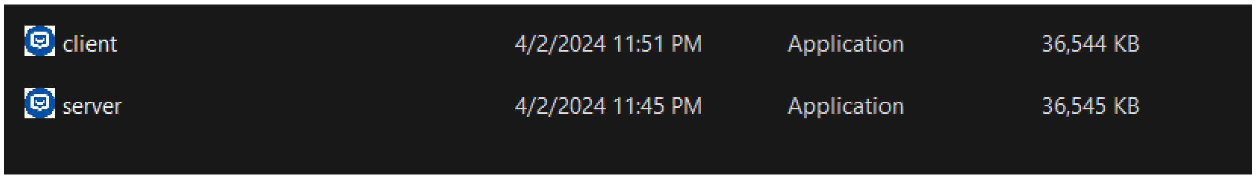
```
Microsoft Windows [Version 10.0.22631.3374]
(c) Microsoft Corporation. All rights reserved.

D:\Masters NPU Semeter\1st Semester\Python\Abid\2. Chat Client pyqt5\pyqt5 Chat Client>python client.py
```

There are two ways to start the program

(1) Typing in the cmd (command prompt) **python client.py** then server start.

(2) Another way by opening the project in IDE then press the play button.

3) Server and Client Executable

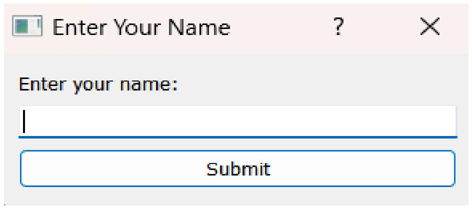


client	4/2/2024 11:51 PM	Application	36,544 KB
server	4/2/2024 11:45 PM	Application	36,545 KB

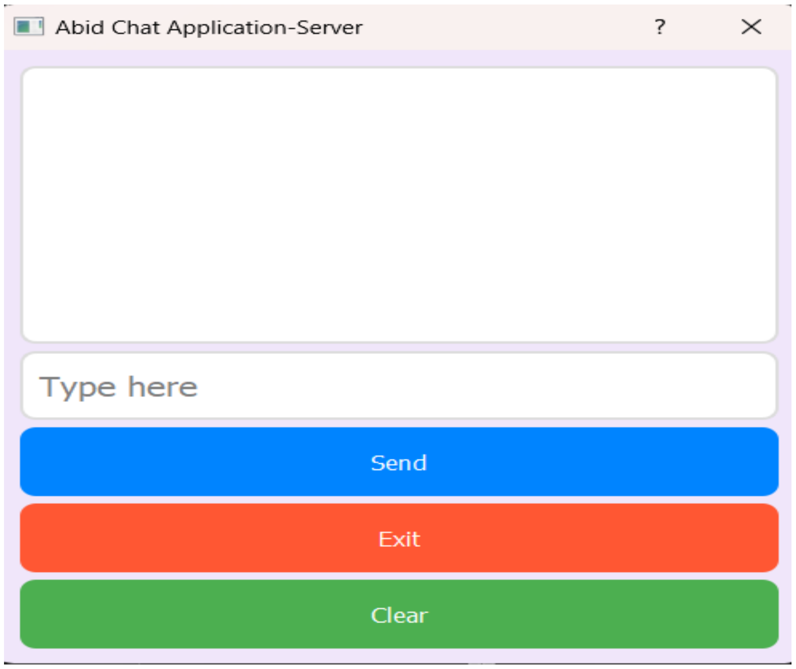
There is a folder named pyqt5 Chat Client App that contains the execuatable file.

(1) Press the server.exe then type your usename
(2) Press the client.exe then type your usename

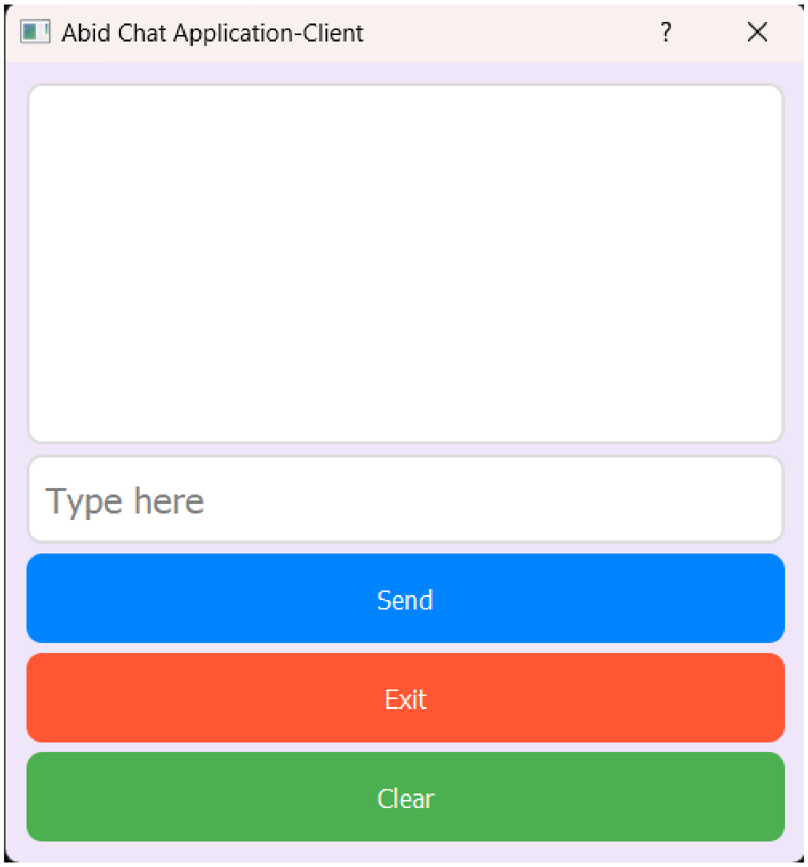
4) Prompting Username



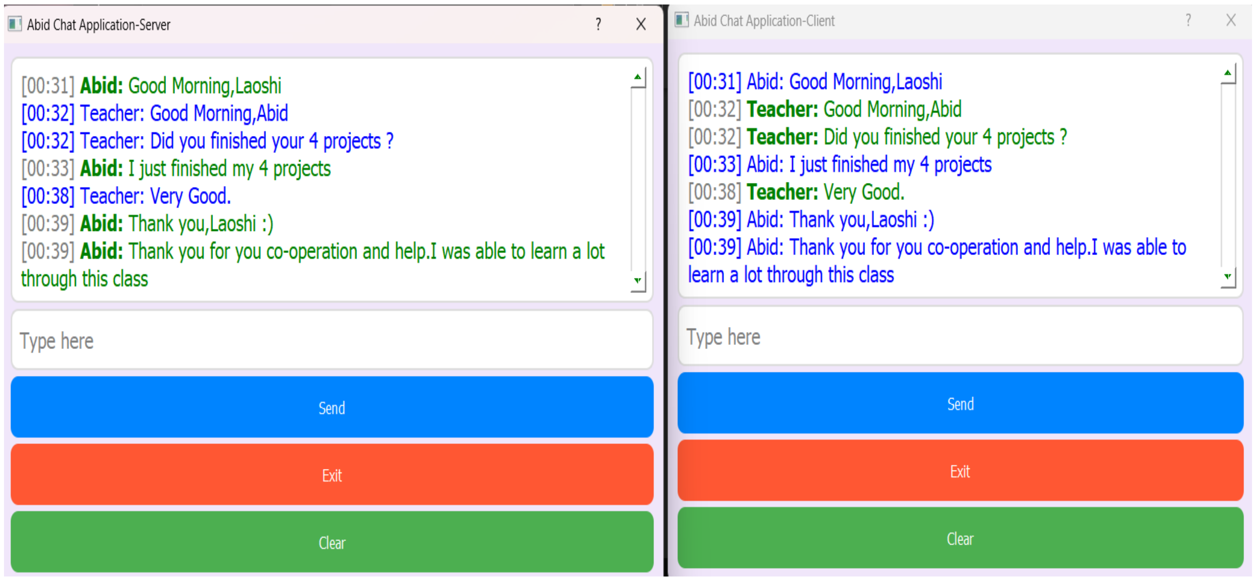
5) Chat Application Server



6) Chat Application Client



7) Chatbot Communication



```
[00:31] Abid: Good Morning,Laoshi
[00:32] Teacher: Good Morning,Abid
[00:32] Teacher: Did you finished your 4 projects ?
[00:33] Abid: I just finished my 4 projects
[00:38] Teacher: Very Good.
[00:39] Abid: Thank you,Laoshi :)
[00:39] Abid: Thank you for you co-operation and help.I was able to learn a lot
through this class
```

```
[00:31] Abid: Good Morning,Laoshi
[00:32] Teacher: Good Morning,Abid
[00:32] Teacher: Did you finished your 4 projects ?
[00:33] Abid: I just finished my 4 projects
[00:38] Teacher: Very Good.
[00:39] Abid: Thank you,Laoshi :)
[00:39] Abid: Thank you for you co-operation and help.I was able to
learn a lot through this class
```

Green color means send message and Blue color mean received message

Assignment 3 : Snake

Description

In this attempt to build a fun arcade game, I used a program in python using a pygame module that will animate a 2D Snake Game using many multimedia functionalities like images, audio and video. This game's concept introduces a 90's retro obstacle avoidance game to modern situations from a fun perspective. The player needs to move using arrow keys (left, right, up and down).

Libraries

- import pygame
- import random
- from pygame.locals import *
- from pygame import mixer

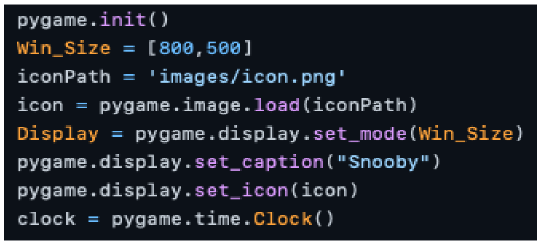
Code Snippets

- def snake(block_width,block_height,SnakeList):
  It initializes and creates a single rectangle block snake.

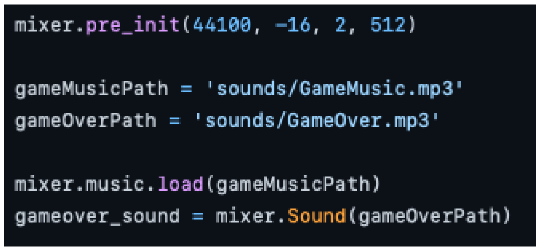
pygame.draw.rect(): This function is used to draw a rectangle.It takes the surface, color, and pygame Rect object as an input parameter and draws a rectangle on the surface mainly for creating the snake.

pygame.display() - This function sets the display mode in pygame and creates a visible image surface on the monitor. This surface can either cover the full screen, or be windowed on platforms that support a window manager.

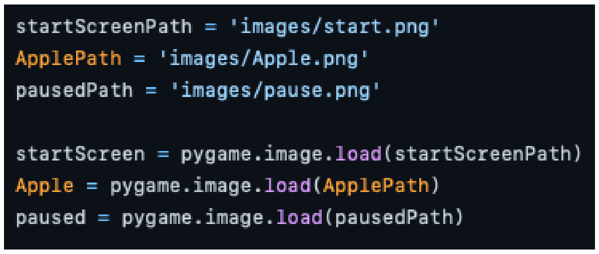
- **Initialization :** This step firstly initializes all the pygame modules and then specifies the window size that needs to be created. It specifies the path of each image in the game with suitable audio path specific to each image.



```
pygame.init()
Win_Size = [800,500]
iconPath = 'images/icon.png'
icon = pygame.image.load(iconPath)
Display = pygame.display.set_mode(Win_Size)
pygame.display.set_caption("Snooby")
pygame.display.set_icon(icon)
clock = pygame.time.Clock()
```

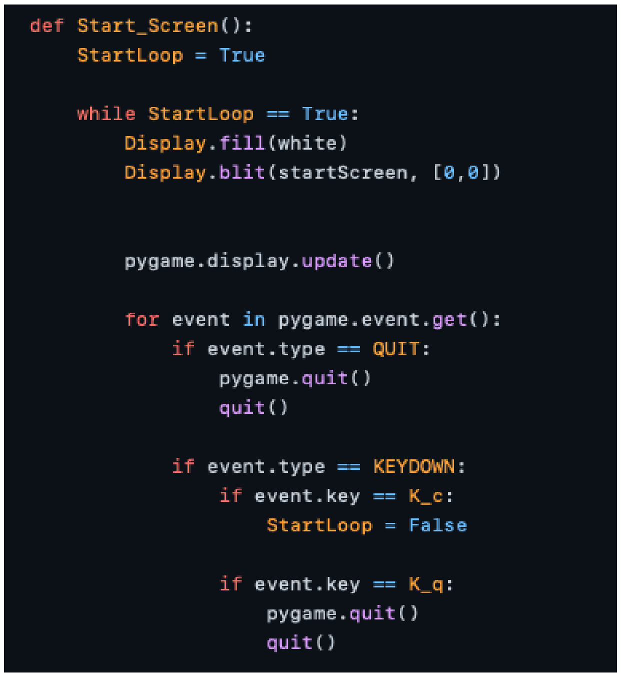


```
mixer.pre_init(44100, -16, 2, 512)

gameMusicPath = 'sounds/GameMusic.mp3'
gameOverPath = 'sounds/GameOver.mp3'

mixer.music.load(gameMusicPath)
gameover_sound = mixer.Sound(gameOverPath)
```



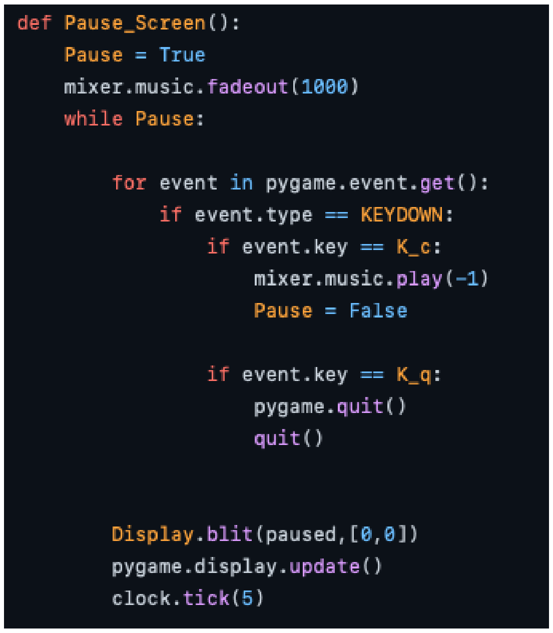
```
startScreenPath = 'images/start.png'
ApplePath = 'images/Apple.png'
pausedPath = 'images/pause.png'

startScreen = pygame.image.load(startScreenPath)
Apple = pygame.image.load(ApplePath)
paused = pygame.image.load(pausedPath)
```

➢ **Start_Screen()** - It creates a surface object which displays the start screen display object on the initial window surface object. This is mainly the game call.



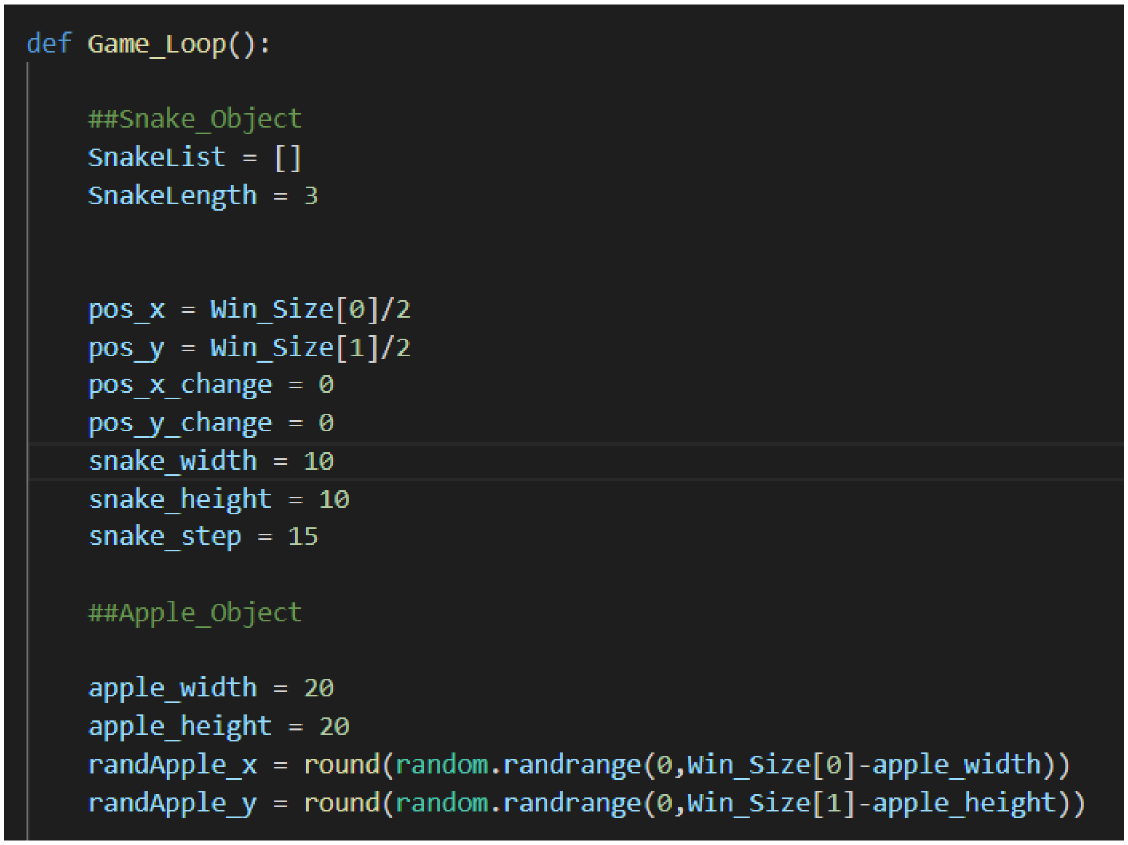
```
def Start_Screen():
    StartLoop = True

    while StartLoop == True:
        Display.fill(white)
        Display.blit(startScreen, [0,0])


        pygame.display.update()

        for event in pygame.event.get():
            if event.type == QUIT:
                pygame.quit()
                quit()

            if event.type == KEYDOWN:
                if event.key == K_c:
                    StartLoop = False

                if event.key == K_q:
                    pygame.quit()
                    quit()
```

➢ **Pause_Screen() -** When we call the pause function, we fade the music after 3000ms (3 sec) and show the pause frame. User has 2 options: continue (press s) and quit (press q). All the events are stored in an event queue and then they are initiated according to the calling order.



```python
def Pause_Screen():
    Pause = True
    mixer.music.fadeout(1000)
    while Pause:

        for event in pygame.event.get():
            if event.type == KEYDOWN:
                if event.key == K_c:
                    mixer.music.play(-1)
                    Pause = False

                if event.key == K_q:
                    pygame.quit()
                    quit()


        Display.blit(paused,[0,0])
        pygame.display.update()
        clock.tick(5)
```

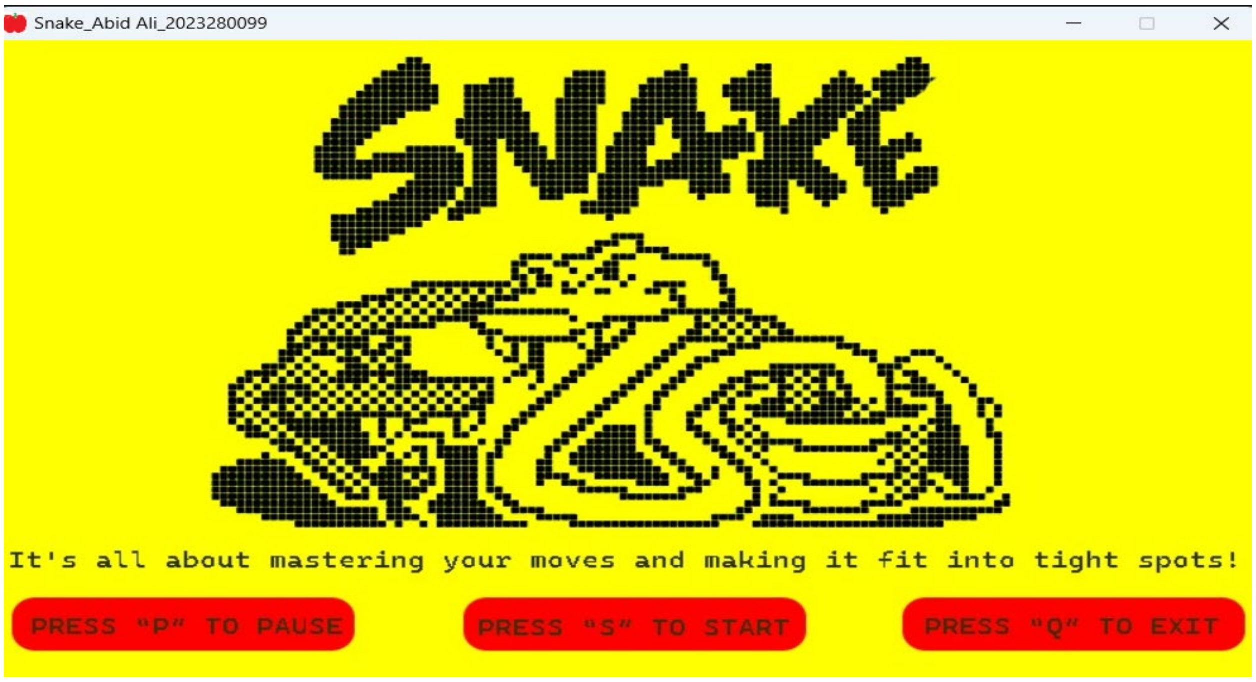
- **Game_loop() -** It is used for looping the game to start the next round which includes all game over and while playing game conditions like wall collisions, snake collision with apple, self-collision, etc.



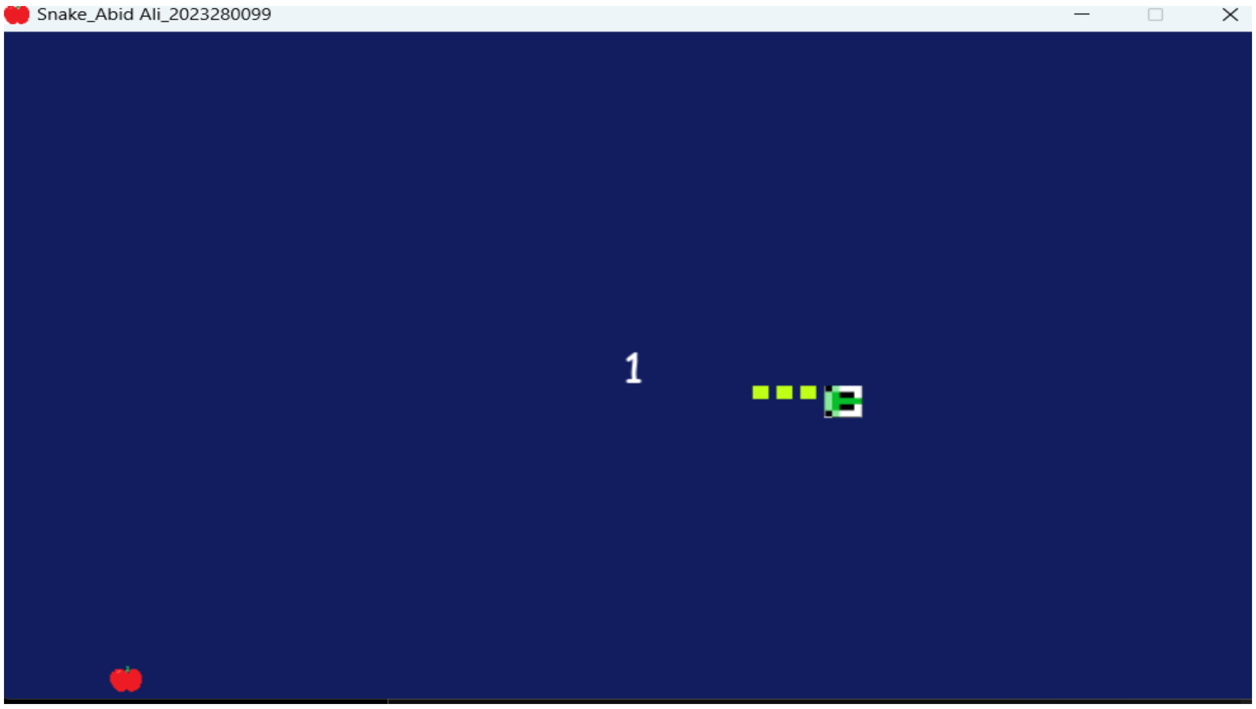
```python
def Game_Loop():

    ##Snake_Object
    SnakeList = []
    SnakeLength = 3


    pos_x = Win_Size[0]/2
    pos_y = Win_Size[1]/2
    pos_x_change = 0
    pos_y_change = 0
    snake_width = 10
    snake_height = 10
    snake_step = 15

    ##Apple_Object

    apple_width = 20
    apple_height = 20
    randApple_x = round(random.randrange(0,Win_Size[0]-apple_width))
    randApple_y = round(random.randrange(0,Win_Size[1]-apple_height))
```

Gameplay Screenshots

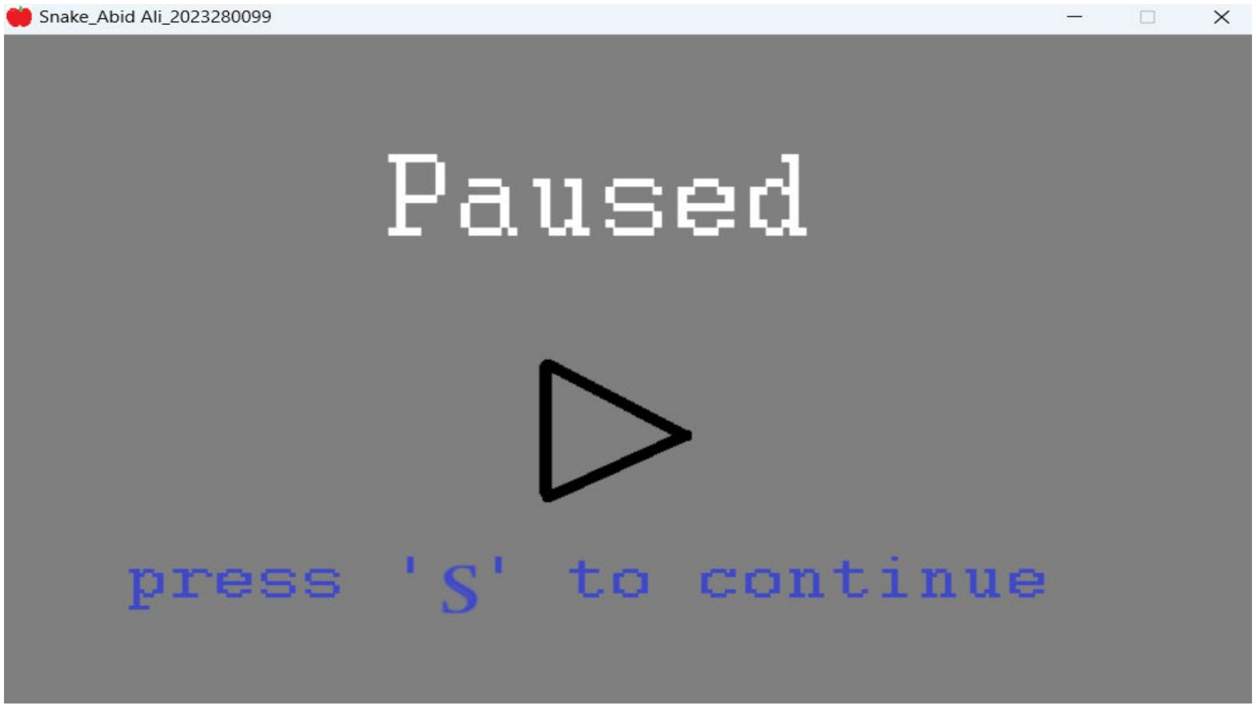
1) Start Screen


Snake_Abid Ali_2023280099
SNAKE
It's all about mastering your moves and making it fit into tight spots!
PRESS "P" TO PAUSE
PRESS "S" TO START
PRESS "Q" TO EXIT

2) Gameplay



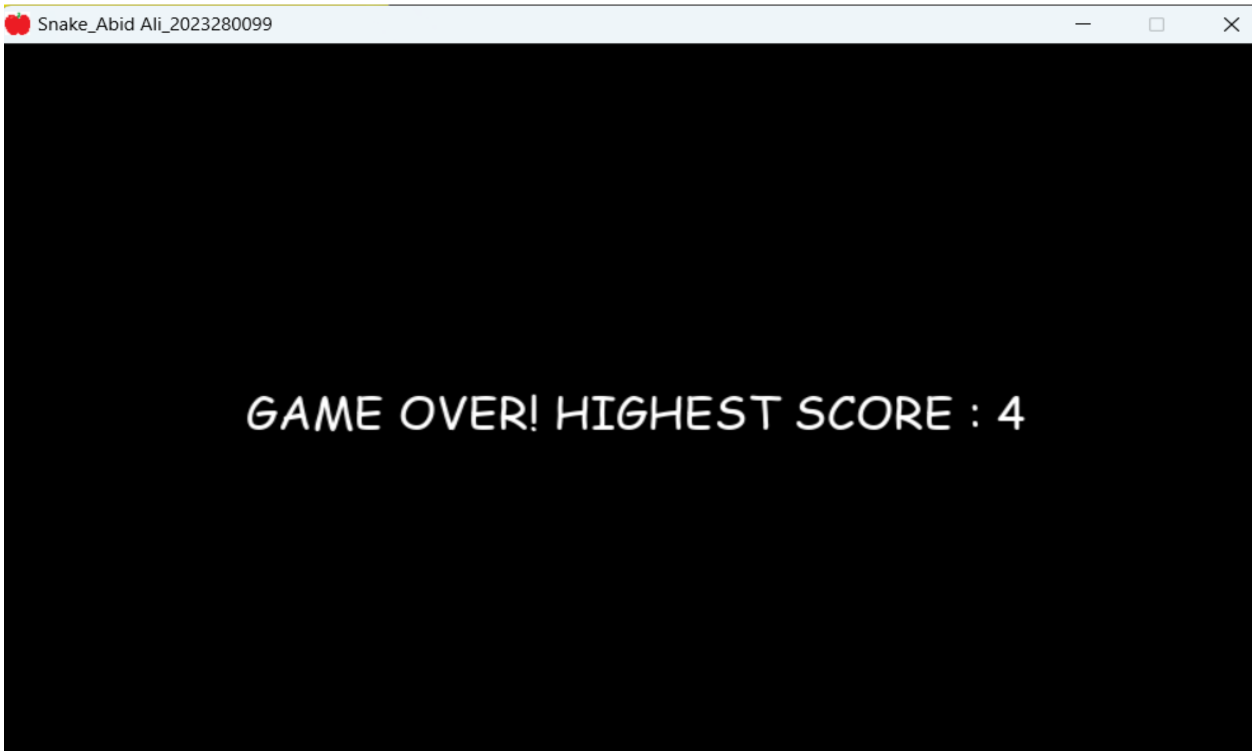
3) Paused Screen


Snake_Abid Ali_2023280099

Paused

press 'S' to continue

4) Game Over


Snake_Abid Ali_2023280099

GAME OVER! HIGHEST SCORE : 4

Assignment 4 : Realtime_PyAudio_FFT

Description

A simple package to do realtime audio analysis in native Python, using PyAudio and Numpy to extract and visualize FFT features from a live audio stream.

Just run `python run_FFT_analyzer.py` and play a sound on your machine!

You can run the stream_analyzer in headless mode and use the FFT features in any Python Application that requires live musical features

Requirements

`pip install -r requirements.txt`

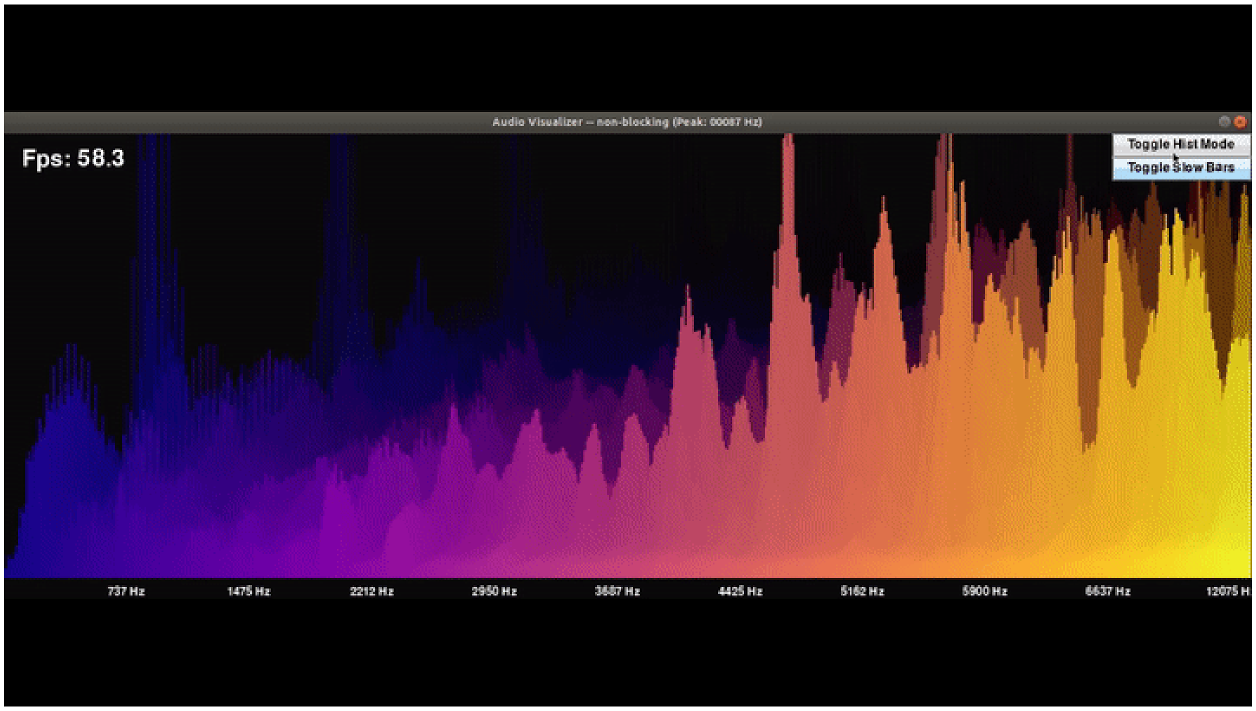
Code Explanation

- Starts a stream_reader that pulls live audio data from any source using PyAudio (soundcard, microphone, ...)
- Reads data from this stream many times per second (eg 1000 updates per second) and stores that data in a fifo buffer.
- When triggered by `.get_audio_features()`, the stream_analyzer, applies a Fast-Fourier-Transform to the most recent audio window in the buffer.
- When `visualize` is enabled, the visualizer displays these FFT features in realtime using a PyGame GUI (I made two display modes: 2D and 3D)
- Other platforms like Mac/Windows should work if PyGame can find your display and Python finds your audio card (these can be tricky with [WSL] (https://research.wmz.ninja/articles/2017/11/setting-up-wsl-with-graphics-and-audio.html))

- For Mac OSX (tested on Catalina 10.15.4), please make sure you run with Python downloaded from [Python.org] (https://www.python.org/downloads/release/python-377/) (`pygame` doesn't work well with the default/Homebrew Python)

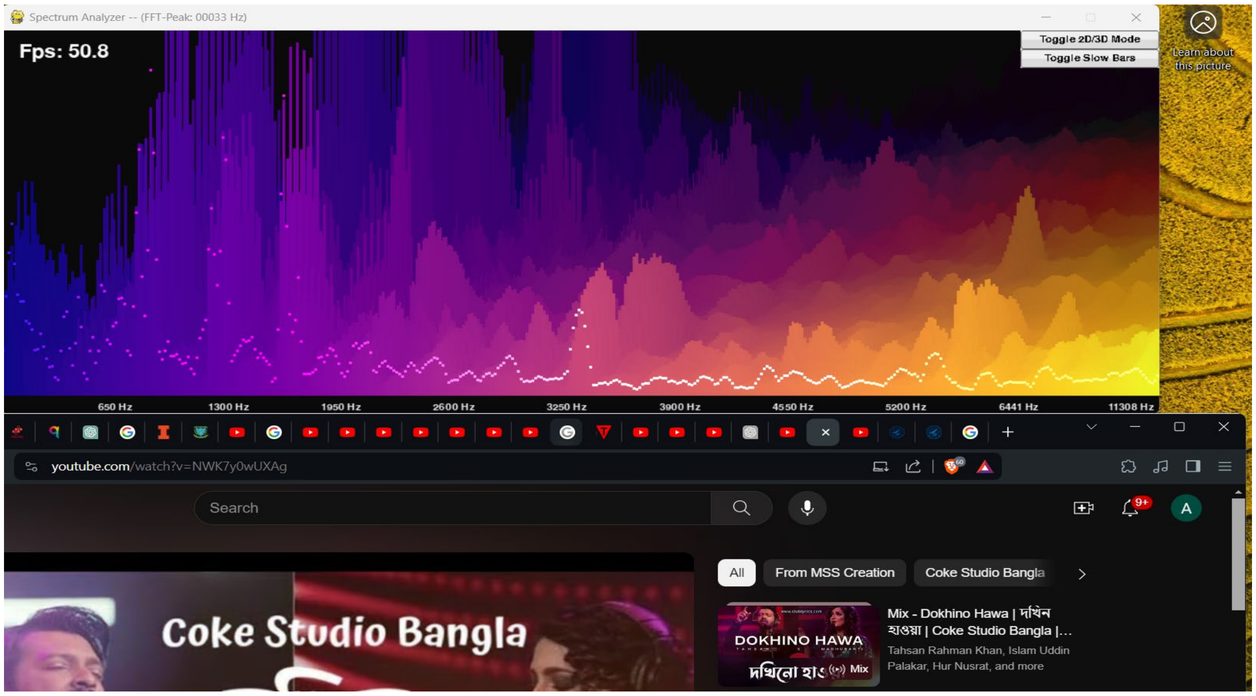
Fast Fourier Transform Feature Screenshots

1) Realtime_PyAudio_FFT GUI

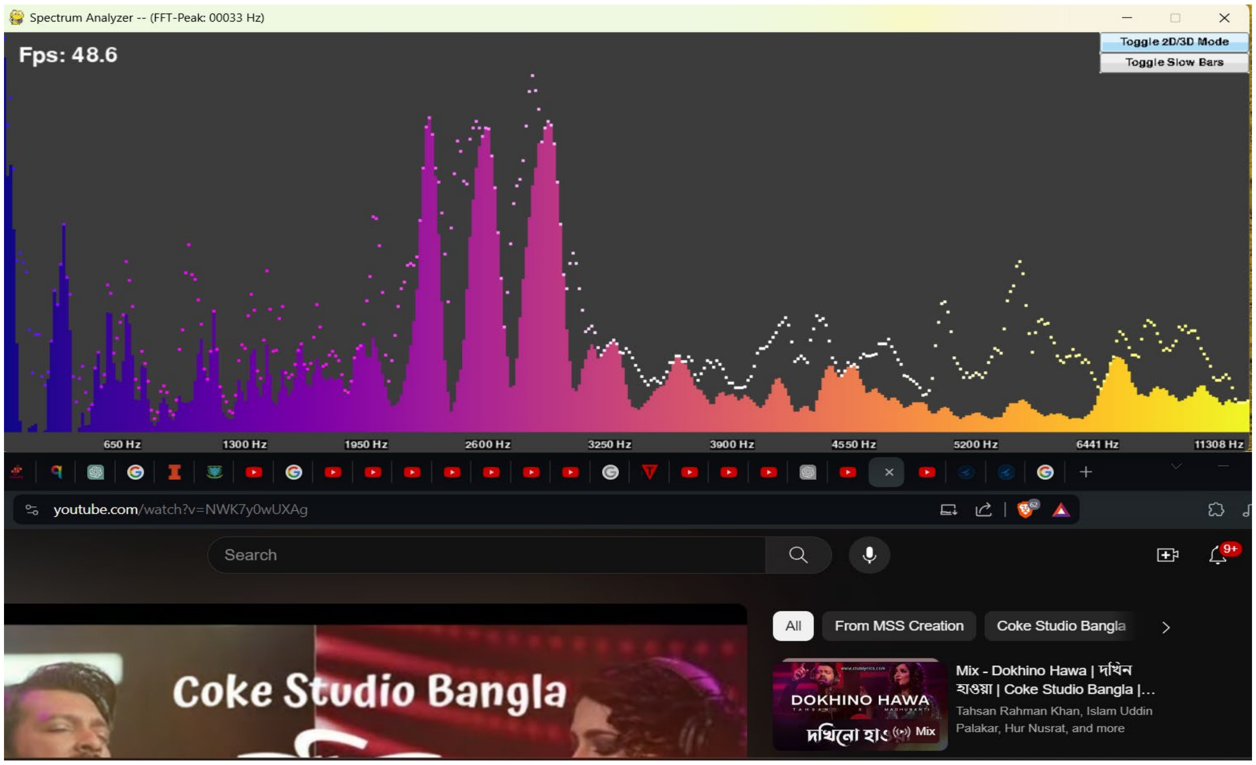


2) Toggle 2D/3D Mode

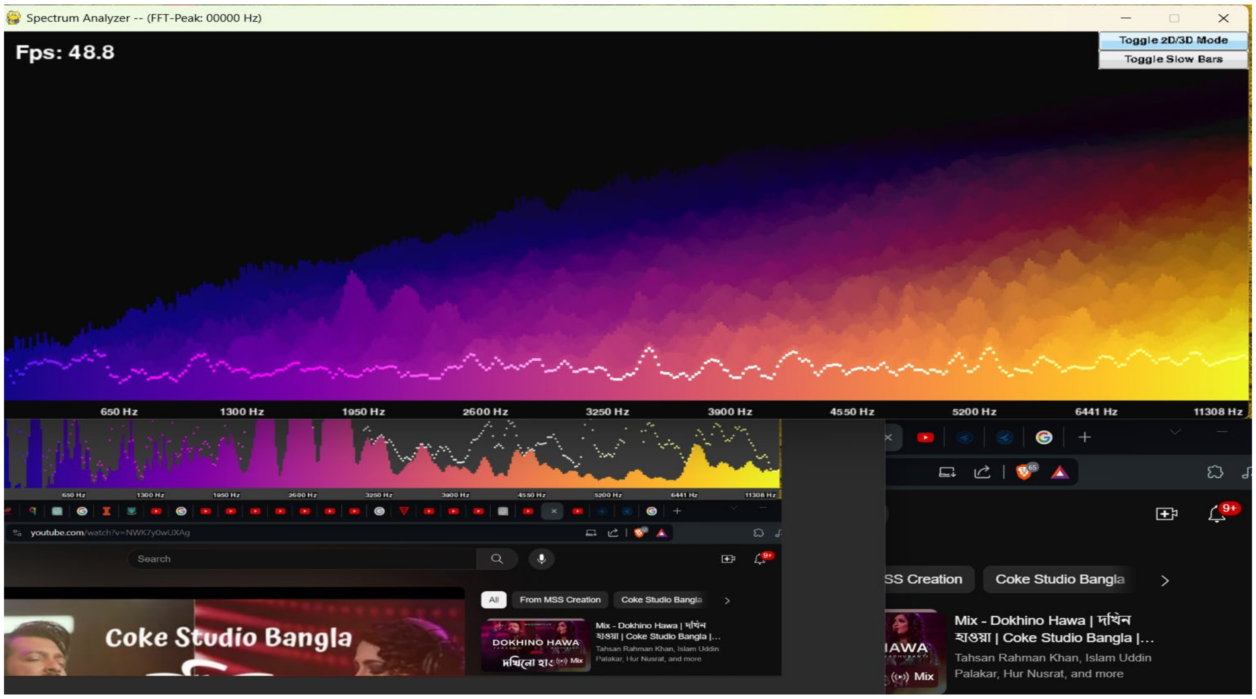
3) 3D Mode Visualization While Playing Music


Spectrum Analyzer -- (FFT-Peak: 00033 Hz)
Fps: 50.8
Toggle 2D/3D Mode
Toggle Slow Bars
650 Hz
1300 Hz
1950 Hz
2600 Hz
3250 Hz
3900 Hz
4550 Hz
5200 Hz
6441 Hz
11308 Hz
youtube.com/watch?v=NWK7y0wUXAg
Search
All
From MSS Creation
Coke Studio Bangla
Mix - Dokhino Hawa | দখিন হাওয়া | Coke Studio Bangla |...
Tahsan Rahman Khan, Islam Uddin Palakar, Hur Nusrat, and more

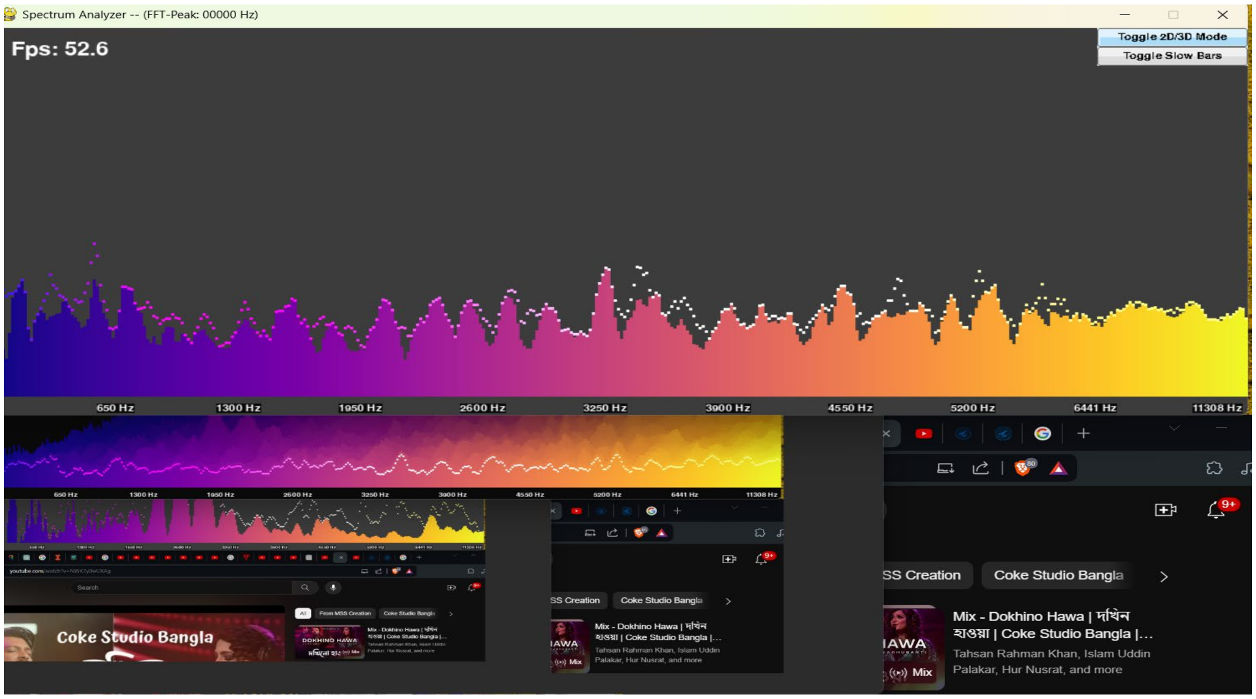
3) 2D Mode Visualization While Playing Music


Spectrum Analyzer -- (FFT-Peak: 00033 Hz)
Fps: 48.6
Toggle 2D/3D Mode
Toggle Slow Bars
650 Hz
1300 Hz
1950 Hz
2600 Hz
3250 Hz
3900 Hz
4550 Hz
5200 Hz
6441 Hz
11308 Hz
youtube.com/watch?v=NWK7y0wUXAg
Search
All
From MSS Creation
Coke Studio Bangla
Mix - Dokhino Hawa | দখিন হাওয়া | Coke Studio Bangla |...
Tahsan Rahman Khan, Islam Uddin Palakar, Hur Nusrat, and more

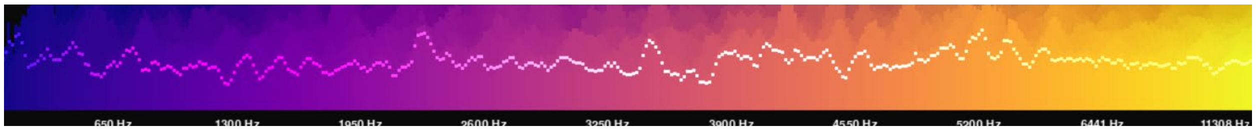
4) 3D Mode Visualization While Paused



5) 2D Mode Visualization While Paused



6) Toggle Slow Bar



Real time static bar to increase the aesthetic beauty by giving a techno style visualization

Acknowledgement

I wrote all of these codes from scratch, although I have prior experience with Python and it remains my preferred language for coding. While my background primarily revolves around artificial intelligence (AI), many of the topics covered in this assignment, such as TCP communication and PyQt5 for GUI development, were new to me. Nevertheless, I thoroughly enjoyed the learning process and found it rewarding to expand my knowledge of Python. Working on unfamiliar concepts broadened my horizons and provided valuable insights into different areas of programming. Python's versatility allowed me to delve into various domains, from AI to web development and GUI applications. Overall, exploring new technologies has been an enriching experience, and I look forward to continuing my journey of learning and growth in Python and beyond.

I want to express my gratitude to our instructor for their exceptional support and clear communication throughout their teaching. Despite being heavily involved in lab research due to my decision to complete my master's program in just two years, I am eager to continue working with our esteemed professor and further expand my knowledge.

Attachment: The zip file containing all the codes.