PYTHON PROJECT

A PROJECT REPORT ON

Table Tennis Video Game

submitted to Manipal University, Jaipur

Bachelor of Technology (B.Tech)

In Computer Science Engineering
(2018-2022)

SUBMITTED BY

Namit Varshney
RegNo. : 189302111
Email : namit.189302111@muj.manipal.edu

INSPIRED BY LIFE

MANIPAL UNIVERSITY
JAIPUR

Established under the Manipal University Jaipur Act (No. 21 of 2011)

Under the guidance of

Dr. Mahesh Jangid

Department of Computer Science and Engineering
School of Computing and Information Technology
Manipal University Jaipur
Jaipur, Rajasthan

INDEX

PROJECT OVERVIEW

Introduction

This project is designed to make a Table Tennis Video Game in Python. The Game is fundamental in its approach with minimal but standard graphics and sound.
Python language is totally used to make the entire game all the graphics and sound abilities are coded in Python.
As we know Python is a universal language, we can easily exploit Python's humongous libraries and Modules for our use.
The code is written in an old school format where is used both programming paradigm, i.e. Procedural & Object-Oriented Style.
This Game highlights how fantastic it is to make something so real, and exhaustive through an easy coding language.

Motivation

The motivation behind this project is my Department Elective Course - Python Programming IT 1552.
This course gave me the right exposure to python programming and it's uses. I learnt many things throughout this course which deepened my understanding of what Python is.
Hence to know and test my knowledge, my professor Mr. Mahesh Jangid gave me the project to make in python or any it's related technologies.

Software Requirements

- Operating System - Win, Mac, Linux
- Python latest version installed, i.e. 3.9.0
- IDE or Code Editor, like Vs Code etc.

Methodology & Python Libraries used

Making the Graphical Interface for the Game

I have made the game with the help of **Turtle Module**.
Turtle is a pre-installed Python library that enables users to create pictures and shapes by providing them with a virtual canvas. The onscreen pen that you use for drawing is called the turtle and this is what gives the library its name. It contains large number of classes which can be used for different graphical representation and 2D- Game Building. The good thing about turtle is that it's a built-in library, so you don't need to install any new packages. All you need to do is import the library into your Python environment.
The turtle module provides turtle graphics primitives, in both object-oriented and procedure-oriented ways. Because it uses tkinter for the underlying graphics, it needs a version of Python installed with Tk support.

For More in-depth knowledge about Turtle:
https://docs.python.org/3/library/turtle.html?highlight=trace#module-turtle

Adding Sound to the game

I used **WinSound Module** to add sound to my game, after a player hits the ball with paddle the sound should come, similarly when a player misses the ball different sound should play to improve gaming experience. Similar to Turtle good thing about Winsound is that it's a built-in library, so you don't need to install any new packages. All you need to do is import the library into your Python environment.

The winsound module provides access to the basic sound-playing machinery provided by Windows platforms. It includes functions and several constants.

- winsound.Beep(frequency, duration)
- winsound.PlaySound(sound, flags)
- winsound.MessageBeep(type=MB_OK)
- winsound.SND_FILENAME
- winsound.SND_ALIAS, and many more...
-

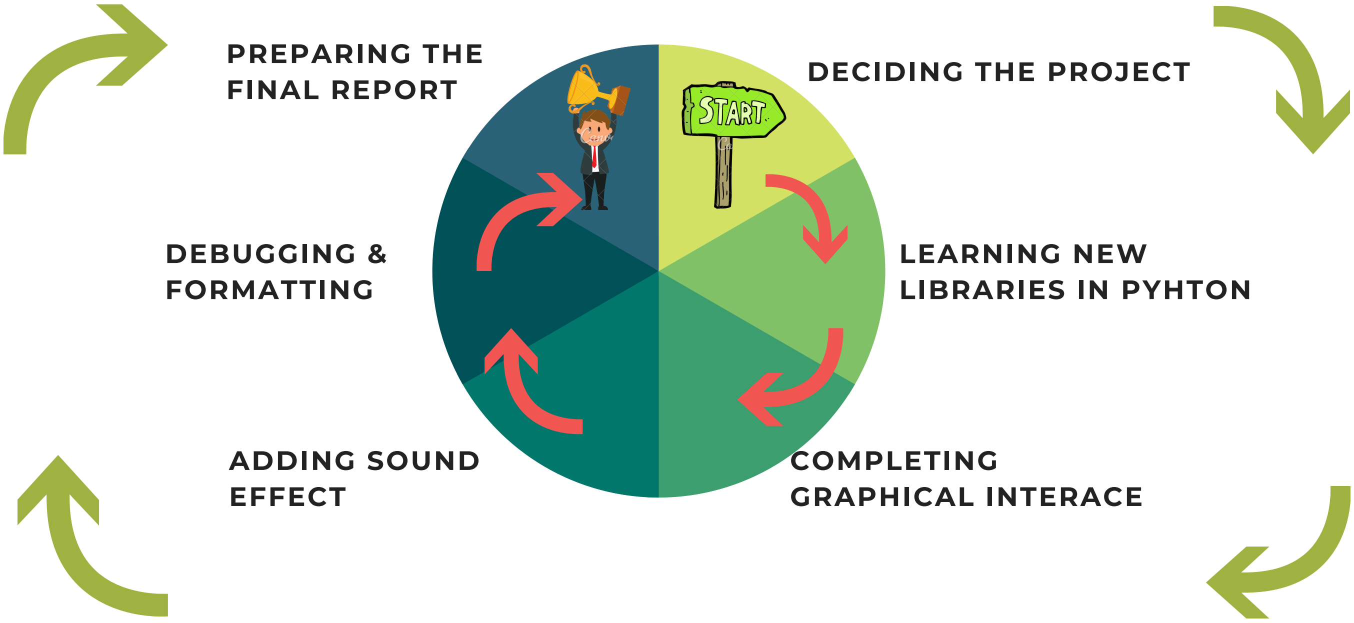
For More in-depth knowledge about WinSound: https://docs.python.org/3/library/winsound.html

MADE BY NAMIT VARSHNEY

- GANTT CHART

13th Nov,2020	Research & Planning
15th Nov,2020	Making a Blueprint
16th-19th Nov,2020	Coding (Trial & Error Phase)
20th-22th Nov,2020	Debugging and Formatting
23rd Nov,2020	FINAL SUBMISSION

- PIE-CHART SHOWING THE MAIN FOUNDTION BLOCKS OF THE PROJECT



DECIDING THE PROJECT

LEARNING NEW LIBRARIES IN PYHTON

COMPLETING GRAPHICAL INTERACE

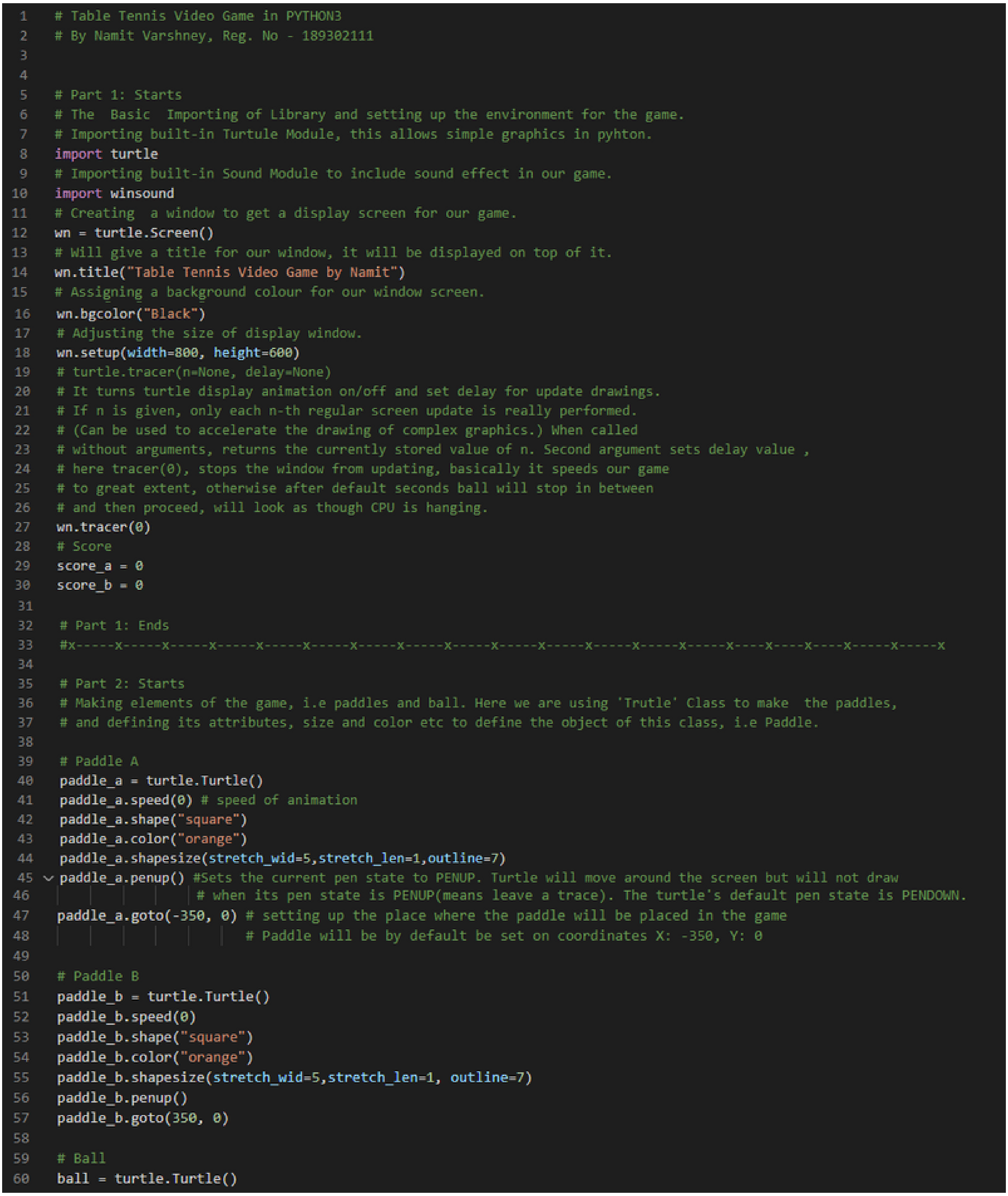
ADDING SOUND EFFECT

DEBUGGING & FORMATTING

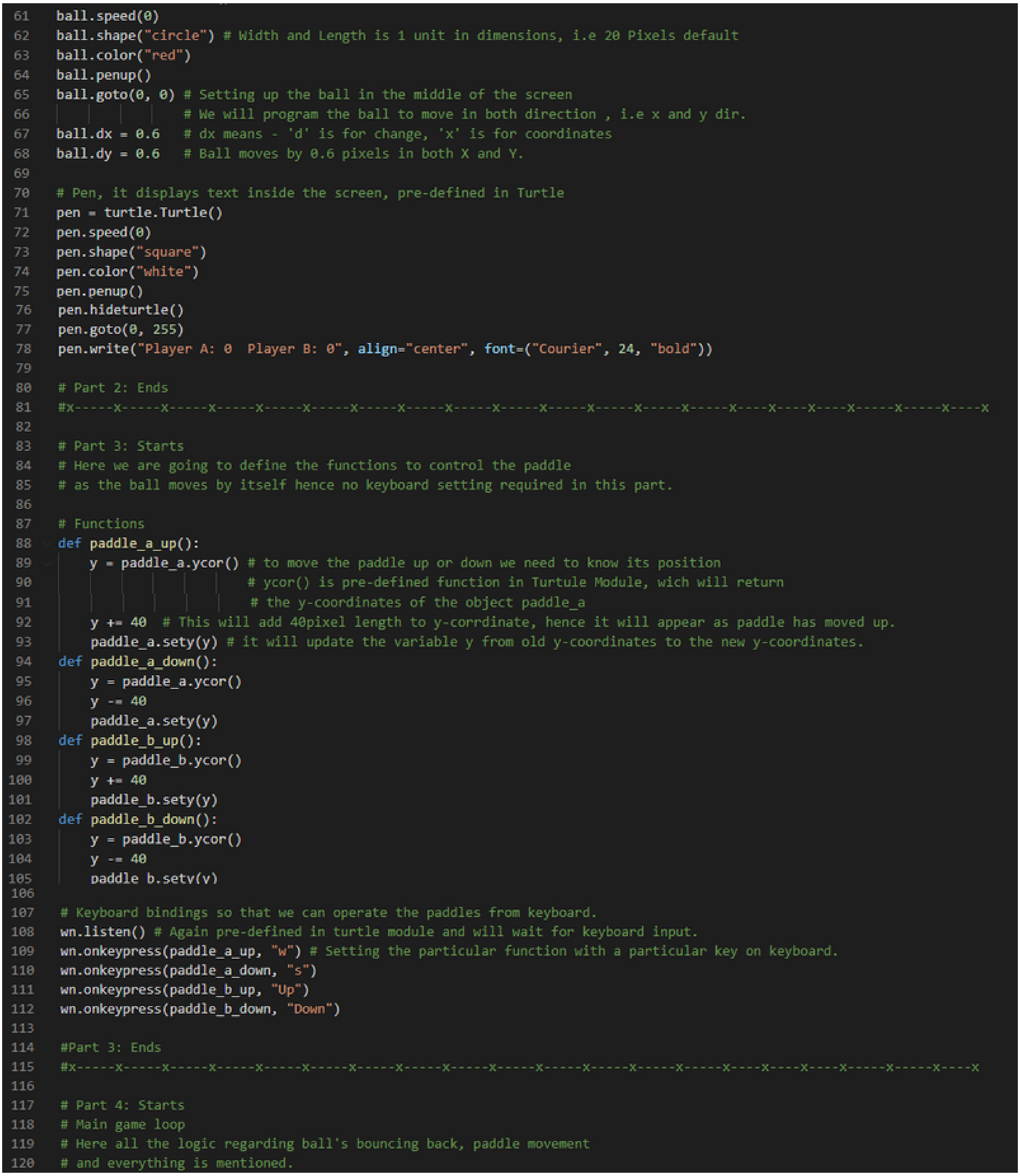
PREPARING THE FINAL REPORT

THE HARD & SMART WORK: PYTHON CODE

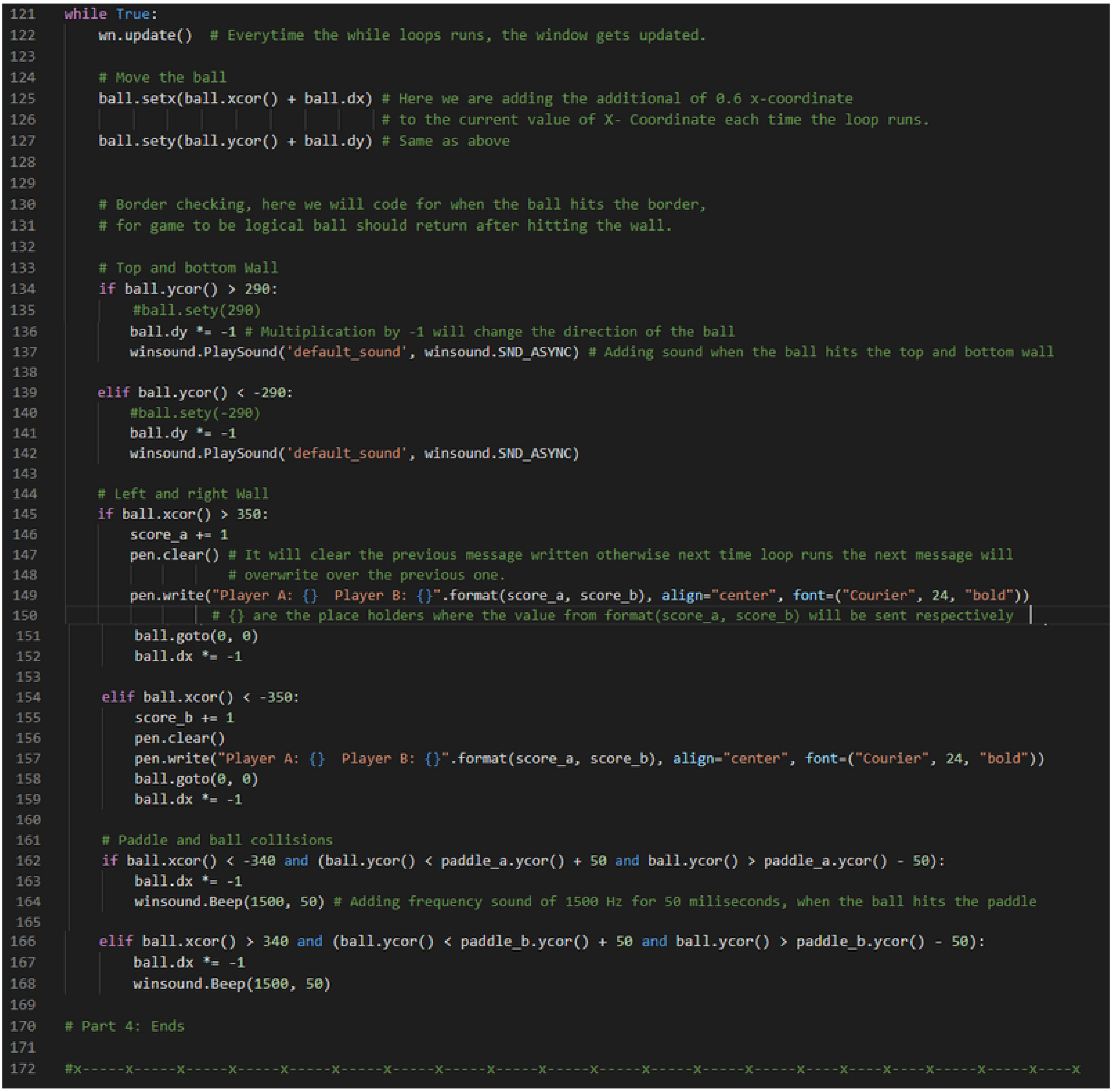
MADE BY NAMIT VARSHNEY



```python
# Table Tennis Video Game in PYTHON3
# By Namit Varshney, Reg. No - 189302111


# Part 1: Starts
# The  Basic  Importing of Library and setting up the environment for the game.
# Importing built-in Turtule Module, this allows simple graphics in pyhton.
import turtle
# Importing built-in Sound Module to include sound effect in our game.
import winsound
# Creating  a window to get a display screen for our game.
wn = turtle.Screen()
# Will give a title for our window, it will be displayed on top of it.
wn.title("Table Tennis Video Game by Namit")
# Assigning a background colour for our window screen.
wn.bgcolor("Black")
# Adjusting the size of display window.
wn.setup(width=800, height=600)
# turtle.tracer(n=None, delay=None)
# It turns turtle display animation on/off and set delay for update drawings.
# If n is given, only each n-th regular screen update is really performed.
# (Can be used to accelerate the drawing of complex graphics.) When called
# without arguments, returns the currently stored value of n. Second argument sets delay value ,
# here tracer(0), stops the window from updating, basically it speeds our game
# to great extent, otherwise after default seconds ball will stop in between
# and then proceed, will look as though CPU is hanging.
wn.tracer(0)
# Score
score_a = 0
score_b = 0

# Part 1: Ends
#x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x----x----x----x-----x-----x

# Part 2: Starts
# Making elements of the game, i.e paddles and ball. Here we are using 'Trutle' Class to make  the paddles,
# and defining its attributes, size and color etc to define the object of this class, i.e Paddle.

# Paddle A
paddle_a = turtle.Turtle()
paddle_a.speed(0) # speed of animation
paddle_a.shape("square")
paddle_a.color("orange")
paddle_a.shapesize(stretch_wid=5,stretch_len=1,outline=7)
paddle_a.penup() #Sets the current pen state to PENUP. Turtle will move around the screen but will not draw
                 # when its pen state is PENUP(means leave a trace). The turtle's default pen state is PENDOWN.
paddle_a.goto(-350, 0) # setting up the place where the paddle will be placed in the game
                       # Paddle will be by default be set on coordinates X: -350, Y: 0

# Paddle B
paddle_b = turtle.Turtle()
paddle_b.speed(0)
paddle_b.shape("square")
paddle_b.color("orange")
paddle_b.shapesize(stretch_wid=5,stretch_len=1, outline=7)
paddle_b.penup()
paddle_b.goto(350, 0)

# Ball
ball = turtle.Turtle()
```



```python
ball.speed(0)
ball.shape("circle") # Width and Length is 1 unit in dimensions, i.e 20 Pixels default
ball.color("red")
ball.penup()
ball.goto(0, 0) # Setting up the ball in the middle of the screen
                # We will program the ball to move in both direction , i.e x and y dir.
ball.dx = 0.6   # dx means - 'd' is for change, 'x' is for coordinates
ball.dy = 0.6   # Ball moves by 0.6 pixels in both X and Y.

# Pen, it displays text inside the screen, pre-defined in Turtle
pen = turtle.Turtle()
pen.speed(0)
pen.shape("square")
pen.color("white")
pen.penup()
pen.hideturtle()
pen.goto(0, 255)
pen.write("Player A: 0  Player B: 0", align="center", font=("Courier", 24, "bold"))

# Part 2: Ends
#x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x----x----x----x-----x-----x----x

# Part 3: Starts
# Here we are going to define the functions to control the paddle
# as the ball moves by itself hence no keyboard setting required in this part.

# Functions
def paddle_a_up():
    y = paddle_a.ycor() # to move the paddle up or down we need to know its position
                        # ycor() is pre-defined function in Turtule Module, wich will return
                        # the y-coordinates of the object paddle_a
    y += 40  # This will add 40pixel length to y-corrdinate, hence it will appear as paddle has moved up.
    paddle_a.sety(y) # it will update the variable y from old y-coordinates to the new y-coordinates.
def paddle_a_down():
    y = paddle_a.ycor()
    y -= 40
    paddle_a.sety(y)
def paddle_b_up():
    y = paddle_b.ycor()
    y += 40
    paddle_b.sety(y)
def paddle_b_down():
    y = paddle_b.ycor()
    y -= 40
    paddle_b.sety(y)

# Keyboard bindings so that we can operate the paddles from keyboard.
wn.listen() # Again pre-defined in turtle module and will wait for keyboard input.
wn.onkeypress(paddle_a_up, "w") # Setting the particular function with a particular key on keyboard.
wn.onkeypress(paddle_a_down, "s")
wn.onkeypress(paddle_b_up, "Up")
wn.onkeypress(paddle_b_down, "Down")

#Part 3: Ends
#x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x----x----x----x-----x-----x----x

# Part 4: Starts
# Main game loop
# Here all the logic regarding ball's bouncing back, paddle movement
# and everything is mentioned.
```



```
while True:
    wn.update()  # Everytime the while loops runs, the window gets updated.

    # Move the ball
    ball.setx(ball.xcor() + ball.dx) # Here we are adding the additional of 0.6 x-coordinate
                                     # to the current value of X- Coordinate each time the loop runs.
    ball.sety(ball.ycor() + ball.dy) # Same as above


    # Border checking, here we will code for when the ball hits the border,
    # for game to be logical ball should return after hitting the wall.

    # Top and bottom Wall
    if ball.ycor() > 290:
        #ball.sety(290)
        ball.dy *= -1 # Multiplication by -1 will change the direction of the ball
        winsound.PlaySound('default_sound', winsound.SND_ASYNC) # Adding sound when the ball hits the top and bottom wall

    elif ball.ycor() < -290:
        #ball.sety(-290)
        ball.dy *= -1
        winsound.PlaySound('default_sound', winsound.SND_ASYNC)

    # Left and right Wall
    if ball.xcor() > 350:
        score_a += 1
        pen.clear() # It will clear the previous message written otherwise next time loop runs the next message will
                    # overwrite over the previous one.
        pen.write("Player A: {}  Player B: {}".format(score_a, score_b), align="center", font=("Courier", 24, "bold"))
                 # {} are the place holders where the value from format(score_a, score_b) will be sent respectively
        ball.goto(0, 0)
        ball.dx *= -1

    elif ball.xcor() < -350:
        score_b += 1
        pen.clear()
        pen.write("Player A: {}  Player B: {}".format(score_a, score_b), align="center", font=("Courier", 24, "bold"))
        ball.goto(0, 0)
        ball.dx *= -1

    # Paddle and ball collisions
    if ball.xcor() < -340 and (ball.ycor() < paddle_a.ycor() + 50 and ball.ycor() > paddle_a.ycor() - 50):
        ball.dx *= -1
        winsound.Beep(1500, 50) # Adding frequency sound of 1500 Hz for 50 miliseconds, when the ball hits the paddle

    elif ball.xcor() > 340 and (ball.ycor() < paddle_b.ycor() + 50 and ball.ycor() > paddle_b.ycor() - 50):
        ball.dx *= -1
        winsound.Beep(1500, 50)

# Part 4: Ends

#x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x-----x----x----x----x-----x-----x----x
```

MADE BY NAMIT VARSHNEY

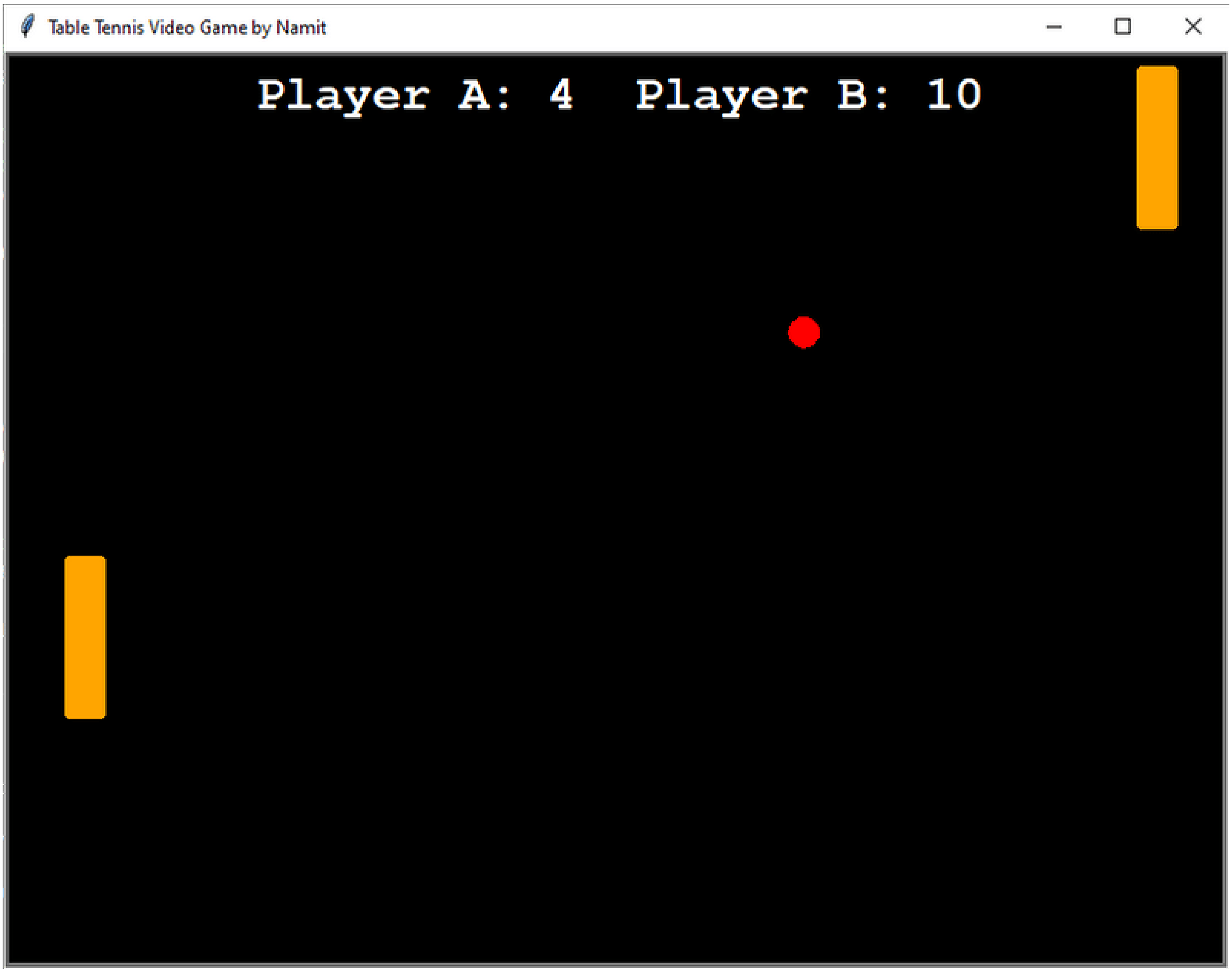
THE ACCOMPLISHMENT:
THE TABLE TENNIS GAME

MADE BY NAMIT VARSHNEY

• HIGHLIGHT OF THE TABLE TENNIS VIDEO GAME

Below is the snippet of Actual Display Screen of the Table Tennis Game. All the code written in the above python file boils down to the image pasted below. As mentioned in code and the documentation above it is generic type of Game, made entirely in Python.

So now it is the time to enjoy the game, and get some score.



BIBLIOGRAPHY

- ***"FreeCodeCamp"***

"https://www.freecodecamp.org/"

- ***"Python Documentation"***
  - https://docs.python.org/3/
  - https://docs.python.org/3/library/turtle.html?highlight=trace#module-turtle

- ***"RealPython(****https://realpython.com/****)"***

https://realpython.com/beginners-guide-python-turtle/

- ***"Google Search"***
  - Effective google search, on various challenging coding problems(Errors and exceptions) came throughout the project.
  - https://stackoverflow.com/