4 Building Classic Pong : Part I

For the next three week we will work to build the game *Pong*[2] using the turtle module in Python. This project takes inspiration from Christian Thompson's fantastic tutorial and we will be building upon his implementation to make it even better.

As you are working through these notes make sure to try the code out for yourself in the IDE of your choice. Information about IDE's can be found on the course web page. If you are still unclear about anything after working through the notes and code make sure to visit the Facebook group.

4.1 The turtle module

For this project we will be making use of the turtle module that is built into Python; the module offers us the ability to draw graphics to the screen and is relatively simple to use.

To get started with the turtle module we first need to import it, we can do that by running the following command.

```
import turtle
```

As described in previous weeks, importing a module then allows us to make use of that modules functionality.

4.2 Setting up the window environment

Now we have imported the turtle module we will set up the window environment we will be using to display the game. The code we will use to do this is written below.

```
#Setting up the window using the turle module.

window = turtle.Screen()
window.title('Pong by Kyle')
window.bgcolor('black') #Note the spelling of colour
window.setup(width=800,height=600) #Set the screen size to 800 x 600 pixels
window.tracer(0) #This turns off the build in animation
                 #we will be animating this ourself
```

Lets work through this code line by line to understand what it is doing:

[2] *Pong* is a 2D video game based on table tennis, it was first released in 1972 by Atari.

(line 1) Here we declare a new screen using the turtle module and we give it the name 'window'.

(line 2) In this line we give the window a name, this will be displayed in the top bar of our window when we run the script.

(line 3) In this line we set the background colour to that classic pong black. Try changing the colour of the background in your implementation.

(line 4) This is an important line as we are setting the screen size. For our project make sure to stick with a size of 800×600.

(line 5) Here we turn off the built in animation function of turtle, this will allow us to have much more control over the animation later in the code.

4.3 The main game loop

Before we run our code for the first time we need to write the main game loop. As you learned in the previous week there are several types of loop that we can use in Python, we will be using a while loop. What we are aiming to do is start an infinite[3] loop that will update the screen and handle all of our animations. We can do this using the following code.

```
#Main game loop

while True:
    window.update()
```

Now would be an ideal point to run your code!

Don't write too much before testing your code as that can make finding the problem harder.

4.4 Building the Court

Our next job is going to be drawing the court lines. Before we do this it is important to understand the geometry of the window that we have already created.

[3] loop that will not stop until it is interrupted

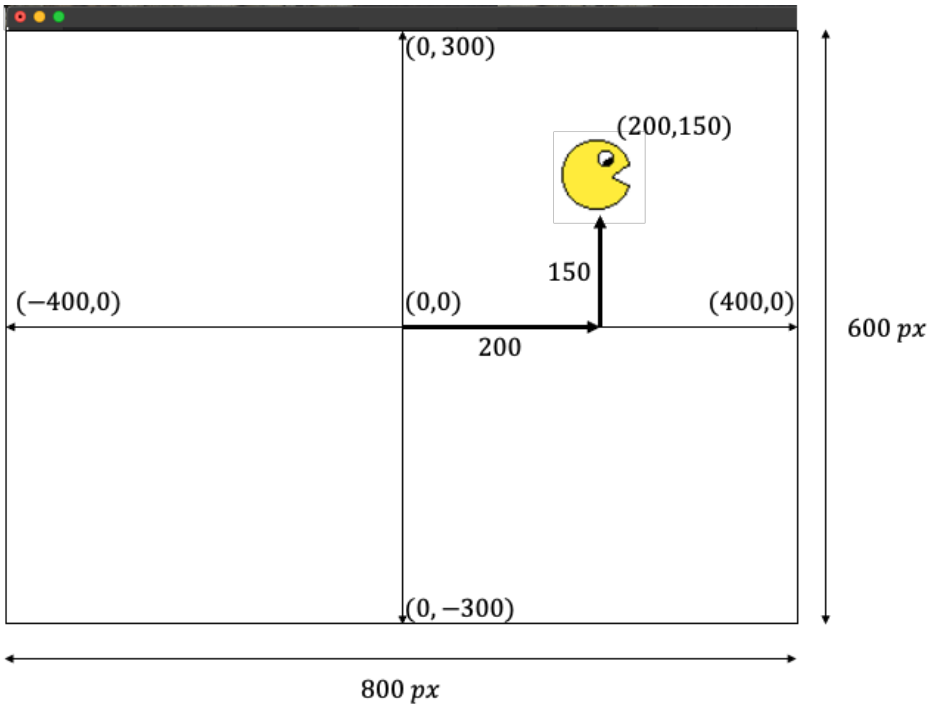

Figure 1: Outline of the window geometry; distances are centered with (0,0).

From figure (1) we can see that distances are measured relative to the center with coordinate (0,0). So in order to describe the location of pacman on the window we must travel 200 pixels to the right and 150 pixels up. The aim with building the court will be to add a dividing line down the center and a line (set of lines) that encompasses the outline of the window. We will do this using the turtle.Turtle() method.

```
#Dividing Line

1 line = turtle.Turtle()
2 line.hideturtle()
3 line.color('white')
4 line.penup()
5 line.goto(0,300)
6 line.pendown()
7 line.goto(0,-300)
```

Lets explore what we did in the above code line by line:

(line 1) In the first line we declare a new object called line and make it a Turtle. This is much like what we did with declaring a new window earlier. Note also the capital T on turtle.

(line 2) In this line we hide the 'turtle' (little onscreen figure) as we don't need it to be visible, we only want the pen lines that the turtle draws to be visible.

(line 3) We set the line colour to be white.

(line 4) When we construct line using a turtle it is much like using a pen on paper and so if we don't want a line to be drawn when we move the turtle then we need to pick the pen up, that is what we do on this line.

(line 5) We relocate the turtle to the edge of the window at (0,300). The default start position is (0,0).

(line 6) We want to start drawing a line so we put the pen down.

(line 7) We move the turtle to the bottom of the screen, this causes a line to be drawn that divides the left and right side - exactly what we wanted!

Exercise: It is your job now to draw a line that goes around the rest of the screen. Use what we already did with drawing the dividing line and extend it.
(Solution in appendix.)

4.5 Paddles and Ball

Week 4: Code Appendix

```
'''
Classic Pong - Inspired by the work of @TokyoEdTech.
File Version #: V1.0
Date: 5th September 2020

Pong is a 2D game that simulates table tennis. In this
implementation the human player will be playing against
the computer. We make use of the turtle module to draw
to the screen.

The following code is useful for WEEK 4 of the introduction
to Python course. Link avaible here:
https://futureskillsprogramme.github.io/Intro_to_python/

Course and code written by Kyle Fogarty.
'''

import turtle

#Setting up the window using the turle module.

window = turtle.Screen()
window.title('Pong by Kyle')
window.bgcolor('black') #Note the spelling of colour
window.setup(width=800,height=600) #Set the screen size to 800 x 600 pixels
window.tracer(0) #This turns off the build in animation
                 #we will be animating this ourself



#Dividing Line & Edge
line = turtle.Turtle()
line.hideturtle()
line.color('white')
line.penup()
line.goto(0,300)
line.pendown()
line.goto(0,-300)
line.goto(-400,-300)
line.goto(-400,300)
line.goto(400,300)
line.goto(400,-300)
line.goto(0,-300)
```

```
#Main Game Loop

while True:
    window.update() #As tracer is turned off we will use this method to
                    #update the window.
```