PONG

A SIMPLE VIDEO GAME

AUM PAUSKAR & SHRIRAM NAIK

CLASS 12 TH B

**INDEX**

|  |  |  |
| --- | --- | --- |
| SR NO. | CONTENT | PAGE NO. |
| 1 | **INTRODUCTION** | **2** |
| 2 | **SYSTEM SPECIFICATIONS** | **3** |
| 3 | **FILES CREATED** | **3** |
| 4 | **HOW TO PLAY** | **4** |
| 5 | **SOURCE CODE** | **5** |
| 6 | **OUTPUT** | **16** |
| 7 | **BIBLIOGRAPHY** | **19** |

**INTRODUCTION**

Pong is one of the most famous arcade games, simulating table tennis. Each player controls a paddle in the game by dragging it vertically across the screen’s left or right side. Players use their paddles to strike back and forth on the ball.

**CODING LANGUAGE USED:** PYTHON 3.8.5

**IDE’s used**: Python IDE and Sublime Text 3

**MODULES USED :** 1) TURTLE

2) TIME

The game is created using the concept of ‘**COORDINATE SYSTEM**’ , with help of which the paddles and the balls navigate.

And to save the game result **‘FILE HANDLING**’ is used.

**SYSTEM SPECIFICATIONS**

1. **OPERATING SYSTEM:**  WINDOWS, MAC OS, LINUX
2. **INTERPRETOR:** PYTHON IDLE
3. **DISK SPACE ( required by the game file):** 40 KB

**==========================================================**

**FILES CREATED**

|  |  |  |
| --- | --- | --- |
| **FILE NAME** | **TYPE** | **DECRIPTION** |
| **PONG3.py** | Python file | Source code |
| **data.txt** | Text file | Saves game result |
| **serial.txt** | Text file | Saves game number |
| **gameover.png** | Png image | Game over image |

**HOW TO PLAY**

* It’s a two-player game so select who will play on which side.
* The left player (blue) will use ‘W’ to go up and ‘S’ to go down.
* The right player (red) will use the standard ‘UP’ and ‘DOWN’ keys.
* To quit or save press ‘Q’.

**SOURCE CODE**

**# Import required libraray**

**import turtle; import time**

**#--------------------------------------------**

**#program variables**

**xdir=440**

**x\_collision=412**

**collision\_lrpad=230**

**gamestate='active' #active or passive**

**savestate='active' #active or passive**

**#these functions become passive on certain conditions**

**blueup\_function='active'**

**bluedown\_function='active'**

**redup\_function='active'**

**reddown\_function='active'**

**#or passive**

**#--------------------------------------------**

**#screen attributes**

**screen= turtle.Screen()**

**screen.title("Pong")**

**screen.bgcolor("#f0f0f0")**

**screen.setup(width=1000, height=600)**

**#net attributes**

**net=turtle.Turtle()**

**net.pensize(8)**

**net.goto(0,600)**

**net.color('black')**

**net.goto(0,-600)**

**if gamestate=='active':**

**# Left paddle**

**lpad = turtle.Turtle()**

**lpad.speed(0)**

**lpad.shape("square")**

**lpad.color("blue")**

**lpad.shapesize(stretch\_wid=6, stretch\_len=1)**

**lpad.penup()**

**lpad.goto(-xdir, 0)**

**#right paddle**

**rpad = turtle.Turtle()**

**rpad.speed(0)**

**rpad.shape("square")**

**rpad.color("red")**

**rpad.shapesize(stretch\_wid=6, stretch\_len=1)**

**rpad.penup()**

**rpad.goto(xdir, 0)**

**elif gamestate=='passive':**

**pass**

**# Ball of circle shape**

**ball = turtle.Turtle()**

**ball.speed(20)**

**ball.shape("circle")**

**ball.color("green")**

**ball.penup()**

**ball.goto(0, 0)**

**ball.dx = 5**

**ball.dy = -5**

**# Initialize the score**

**left\_player = 0**

**right\_player = 0**

**# Displays the score**

**score = turtle.Turtle()**

**score.speed(0)**

**score.color("blue")**

**score.penup()**

**score.hideturtle()**

**score.goto(0, 260)**

**score.write("Left\_player : 0 Right\_player: 0",align="center", font=("Courier", 24, "normal"))**

**#--------------------------------------------**

**# Functions to move paddle vertically**

**def BlueUp():**

**global gamestate**

**if gamestate=='active':**

**global blueup\_function ;global bluedown\_function**

**if blueup\_function=='active':**

**y = lpad.ycor()**

**y += 20**

**lpad.sety(y)**

**elif blueup\_function=='passive':**

**pass**

**bluedown\_function='active'**

**elif gamestate=='passive':**

**pass**

**def BlueDown():**

**global gamestate**

**if gamestate=='active':**

**global blueup\_function ;global bluedown\_function**

**if bluedown\_function=='active':**

**yblue\_cor=lpad.ycor()**

**yblue\_cor-= 20**

**lpad.sety(yblue\_cor)**

**elif bluedown\_function=='passive':**

**pass**

**blueup\_function='active'**

**elif gamestate=='passive':**

**pass**

**def RedUp():**

**global gamestate**

**if gamestate=='active':**

**global redup\_function ;global reddown\_function**

**if redup\_function=='active':**

**yred\_cor= rpad.ycor()**

**yred\_cor+= 20**

**rpad.sety(yred\_cor)**

**elif redup\_function=='passive':**

**pass**

**reddown\_function='active'**

**elif gamestate=='passive':**

**pass**

**def RedDown():**

**global gamestate**

**if gamestate=='active':**

**global redup\_function ;global reddown\_function**

**if reddown\_function=='active':**

**yred\_cor=rpad.ycor()**

**yred\_cor-= 20**

**rpad.sety(yred\_cor)**

**elif reddown\_function=='passive':**

**pass**

**redup\_function='active'**

**elif gamestate=='passive':**

**pass**

**def Save():**

**global gamestate; global savestate**

**global redup\_function; global reddown\_function; global blueup\_function; global bluedown\_function**

**if savestate=='active':**

**global left\_player ;global right\_player**

**hour=str(time.strftime('%I'))**

**minute=str(time.strftime('%M'))**

**second=str(time.strftime('%S'))**

**am\_pm=str(time.strftime('%p'))**

**date=str(time.strftime('%d'))**

**month=str(time.strftime('%m'))**

**year=str(time.strftime('%Y'))**

**day=str(time.strftime('%a'))**

**curtime=hour+':'+minute+':'+second+' '+am\_pm**

**curdate=date+'/'+month+'/'+year+' - '+day**

**if left\_player==0 and right\_player==0:**

**status='Start'**

**elif left\_player==right\_player:**

**status='Tie'**

**elif left\_player>right\_player:**

**status='Blue wins'**

**elif left\_player<right\_player:**

**status='Red wins'**

**serial=open('serial.txt', 'r')**

**if serial.read()=='':**

**serial=open('serial.txt', 'w')**

**serial.write('1')**

**serial.close()**

**else:**

**serial=open('serial.txt', 'r')**

**var1=serial.read()**

**var1=int(var1)**

**var1+=1**

**var1=str(var1)**

**serial=open('serial.txt', 'w')**

**serial.write(var1)**

**serial.close()**

**data=open('data.txt', 'a')**

**data.write('\n\nGame number: '+var1+'\n')**

**data.write('Blue score: '+str(left\_player)+'\n')**

**data.write('Red score: '+str(right\_player)+'\n')**

**data.write('Status: '+status+'\n')**

**data.write('Recorded at: '+curtime+'\n')**

**data.write('At date: '+curdate+'\n-------------------------------------')**

**elif savestate=='passive':**

**pass**

**gamestate='passive'; savestate='passive'**

**#-----------------------------------------**

**#key bindings**

**screen.listen()**

**screen.onkeypress(BlueUp, "w")**

**screen.onkeypress(BlueDown, "s")**

**screen.onkeypress(RedUp, "Up")**

**screen.onkeypress(RedDown, "Down")**

**screen.onkeypress(Save, 'q')**

**while True:**

**screen.update()**

**if gamestate=='active':**

**if lpad.ycor()<=-(collision\_lrpad):**

**bluedown\_function='passive'**

**if lpad.ycor()>=(collision\_lrpad):**

**blueup\_function='passive'**

**if rpad.ycor()<=-(collision\_lrpad):**

**reddown\_function='passive'**

**if rpad.ycor()>=(collision\_lrpad):**

**redup\_function='passive'**

**ball.setx(ball.xcor()+ball.dx)**

**ball.sety(ball.ycor()+ball.dy)**

**# Checking borders**

**if ball.ycor() > 280:**

**ball.sety(280)**

**ball.dy \*= -1**

**if ball.ycor() < -280:**

**ball.sety(-280)**

**ball.dy \*= -1**

**#for bouncing off the y axis border**

**if ball.xcor() > 500:**

**ball.goto(0, 0)**

**ball.dy \*= -1**

**left\_player += 1**

**score.clear()**

**score.write("Left\_player : {} Right\_player: {}".format(left\_player, right\_player), align="center", font=("Courier", 24, "normal"))**

**if ball.xcor() < -500:**

**ball.goto(0, 0)**

**ball.dy \*= -1**

**right\_player += 1**

**score.clear()**

**score.write("Left\_player : {} Right\_player: {}".format(left\_player, right\_player), align="center", font=("Courier", 24, "normal"))**

**#for bouncing off the x axis border**

**# Paddle and ball collision**

**if ball.xcor() > x\_collision and ball.xcor() < x\_collision+10 and ball.ycor() < rpad.ycor()+40 and ball.ycor() > rpad.ycor()-40:**

**ball.setx(x\_collision)**

**ball.dx\*=-1**

**if ball.xcor()<-x\_collision and ball.xcor()>-(x\_collision+10) and ball.ycor()<lpad.ycor()+40 and ball.ycor()>lpad.ycor()-40:**

**ball.setx(-x\_collision)**

**ball.dx\*=-1**

**elif gamestate=='passive':**

**screen.bgpic('gameover.png')**

**net.clear()**

**lpad.clear()**

**rpad.clear()**

**ball.clear()**

**score.clear()**

**OUTPUT**

1. **START OF THE GAME:**

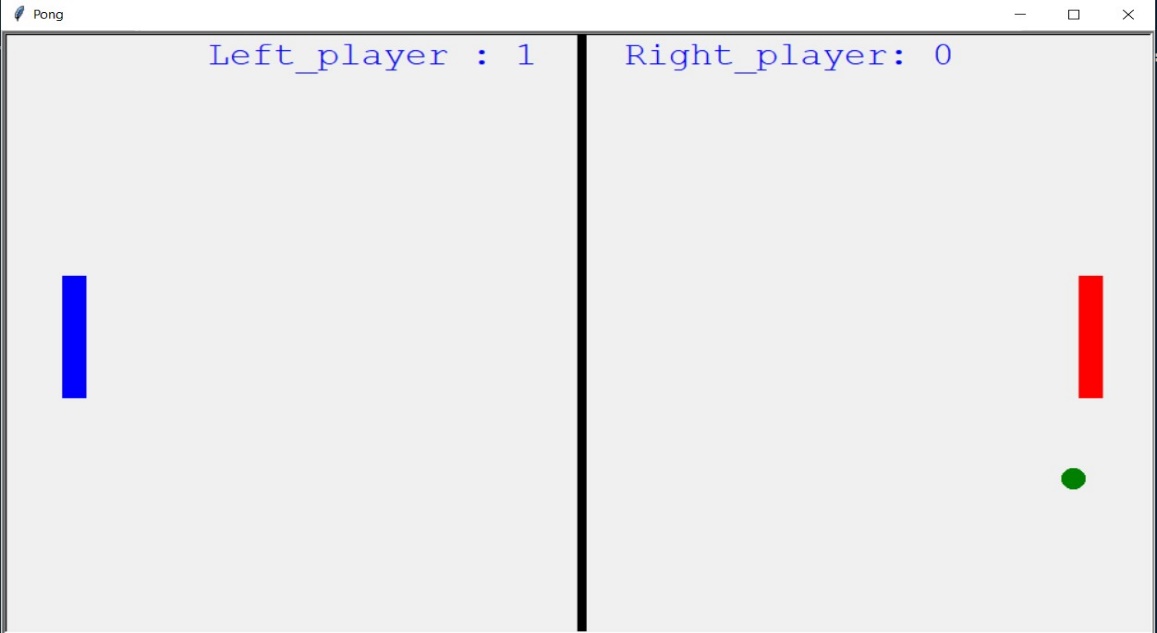
After pressing f5 in Python IDE.



1. **SCORING:**

Keys are defined for both players to navigate the paddles.

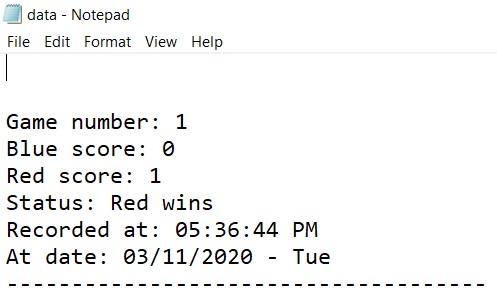
If one misses the other gets score.



1. **TO QUIT**

If the players press ‘**Q’**. The game will end the game over screen will appear. The game result will be saved in the data.txt file.





**BIBLIOGRAPHY**

**The following things helped us complete our project:**

1. INTERNET: Geeks for Geeks (turtle functions)
2. BOOK: COMPUTER SCIENCE WITH PYTHON – Sumita Arora

(Class 11th & 12th)