Pong V3 Algorithm

Based on "Template - Events Algorithm"

main program

create window (U)

create Game called game, using window (U)

play game (U)

close window (L)

function create window (out: window)

initialize graphics library (L)

open window with title 'Pong', size 500 by 400 (L)

class Game

window

bg color

pause time

close clicked

continue game

Right paddle

Left paddle

ball

**set key repeat to 20,20**

Game function create (in: window, out: game)

set window using argument

create Color object, called bg color, using 'black' (L)

set pause time to value

set close clicked to false

set continue game to true

Create Right paddle

Create left paddle

Create ball object (U)

Game function play (in: self)

draw self (U)

while not close clicked

play frame (B)

Game block play frame

handle event on self (U)

if continue game

update self (U)

decide continue on self (U)

draw self (U)

pause for pause time (L)

Game function handle event (in: self)

get next event (L)

if type of event equals window close

set close clicked to true

**if type of event equals KEYDOWN and continue game**

**if pygame.key.get\_pressed equals ‘q’**

**if paddle.top > 6**

**Move left paddle up by 6 pixels**

**elif paddle.top > 0**

**set left paddle.top to 0**

**if pygame.key.get\_pressed equals ‘a’**

**if paddle.top + paddle.height + 6 < surface.height**

**Move left paddle down by 6 pixels**

**elif paddle.top + paddle.height < surface.height**

**set left paddle.top to surface.height - left paddle.top**

**if pygame.key.get\_pressed equals ‘p’**

**if paddle.top > 6**

**Move right paddle up by 6 pixels**

**elif paddle.top > 0**

**set right paddle to 0**

**if pygame.key.get\_pressed equals ‘l’**

**if paddle.top + paddle.height + 6 < surface.height**

**Move right paddle down by 6 pixels**

**elif paddle.top + paddle.height < surface.height**

**set right paddle to surface.height - right paddle.top**

Game function draw (in: self)

fill window using bg color (L)

draw ball (U)

draw left paddle (U)

Draw right paddle (U)

Draw score(U)

update display (L)

Game function draw score(in: self)

Draws the left score

Draws the right score

Game function update (in: self)

move ball (U)

If ball.xvelocity > 0 #this means the ball is moving right

Collide right paddle (U)

Else

collide left paddle(U)

Game function decide continue (in: self)

if the score on either side = 11

continue\_game is False

Game function collide paddle(in:self, paddle)

#paddle is the paddle rectangle we are checking, either left or right

If the center of the ball has collided with the paddle xvelocity = -xvelocity

class Ball

radius

center

color

velocity

Window

Left score

Right score

Ball function create(in:radius,center,color,velocity,window out:ball)

set window using argument

set radius using argument

set center using argument

set color using argument

set velocity using argument

Set left score = 0

Set right score =0

Ball function draw (in: self)

Draw a circle with radius = 5, color = white, position = middle of screen(L)

Ball function move (in: self)

For coord in range(0, 2) #x and y

Set coord to coord + velocity

If coord >= screenwidth - radius:

velocity = -velocity

If coord = 0: #x

Increase left score(U)

Elif coord <= 0 + radius

velocity = -velocity

If coord = 0: #x

Increase right score(U)