```
#-------------------------------------------------------------------------------
# Name:        module1
# Purpose:
#
# Author:      Avinash
#
# Created:     01/05/2014
# Copyright:   (c) Avinash 2014
# Licence:     <your licence>
#-------------------------------------------------------------------------------

from Tkinter import *
import random
import time
import sys

counter = 0
a = 0
b = 0
c = 0
d = 0
e = 0
f = 0
g = 0

tk = Tk()
tk.title("Bounce!")
tk.resizable(0,0)
tk.wm_attributes("-topmost", 1)
canvas = Canvas(tk, width = 535, height = 400, bd = 0, highlightthickness=0)
canvas.pack()
canvas.update()

class Ball:

    def __init__(self,canvas,paddle,block,color):

        self.canvas = canvas
        self.paddle = paddle
        self.block = block
        self.id = canvas.create_oval(10,10,25,25, fill = color)
        self.canvas.move(self.id, 245,100)
        starts = [-3,-2,-1,1,2,3]
        random.shuffle(starts)
        self.x = starts[0]
        self.y = -1
        self.canvas_height = self.canvas.winfo_height()
```

```
    self.canvas_width = self.canvas.winfo_width()
    self.hit_bottom = False
    #self.canvas.create_text( 470, 10, text = "0", anchor = NE, font = ("Courier", 28))
def score(self):
    global counter
    counter += 1
    tk.title("Bounce! Score = " + str(counter))

def hit_block(self,pos):
    block_pos = self.canvas.coords(self.block.id)
    List = [block_pos]
    for i in List:
        if pos[0] >= i[0] and pos[2] <= i[2]:
            if pos[1] >= i[1] and pos[1] <= i[3]:
                self.score()
                global a
                a += 1
                return True
    return False

def hit_block1(self,pos):
    block_pos1 = self.canvas.coords(self.block.id1)
    List = [block_pos1]
    for i in List:
        if pos[0] >= i[0] and pos[2] <= i[2]:
            if pos[1] >= i[1] and pos[1] <= i[3]:
                self.score()
                global b
                b += 1
                return True

    return False
def hit_block2(self,pos):
    block_pos2 = self.canvas.coords(self.block.id2)
    List = [block_pos2]
    for i in List:
        if pos[0] >= i[0] and pos[2] <= i[2]:
            if pos[1] >= i[1] and pos[1] <= i[3]:
                self.score()
                global c
                c += 1
                return True
    return False

def hit_block3(self,pos):
    block_pos3 = self.canvas.coords(self.block.id3)
    List = [block_pos3]
```

```
    for i in List:
        if pos[0] >= i[0] and pos[2] <= i[2]:
            if pos[1] >= i[1] and pos[1] <= i[3]:
                self.score()
                global d
                d += 1
                return True
    return False

def hit_block4(self,pos):
    block_pos4 = self.canvas.coords(self.block.id4)
    List = [block_pos4]
    for i in List:
        if pos[0] >= i[0] and pos[2] <= i[2]:
            if pos[1] >= i[1] and pos[1] <= i[3]:
                self.score()
                global e
                e += 1
                return True

    return False

def hit_block5(self,pos):
    block_pos5 = self.canvas.coords(self.block.id5)
    List = [block_pos5]
    for i in List:
        if (pos[0]  >= i[0] and pos[2] <= i[2]):
            if pos[1] >= i[1] and pos[1] <= i[3]:
                self.score()
                global f
                f += 1
                return True
        if pos[2] >= i[0] and pos[0] <= i[2]:
            if pos[3] >= i[1] and pos[3] <= i[3]:
                f += 1
                return False


def hit_block6(self,pos):
    block_pos6 = self.canvas.coords(self.block.id6)
    List = [block_pos6]
    for i in List:
        if pos[0] >= i[0] and pos[2] <= i[2]:
            if pos[1] >= i[1] and pos[1] <= i[3]:
                self.score()
                global g
                g += 1
```

```
            return True
        if pos[2] >= i[0] and pos[0] <= i[2]:
            if pos[3] >= i[1] and pos[3] <= i[3]:
                g += 1
                return False



def hit_paddle(self,pos):
    paddle_pos = self.canvas.coords(self.paddle.id)
    if pos[2] >= paddle_pos[0] and pos[0] <= paddle_pos[2]:
        if pos[3] >= paddle_pos[1] and pos[3] <= paddle_pos[3]:
            self.x += self.paddle.x
            self.y += 1
            self.score()
            return True
    return False
def draw(self):
    self.canvas.move(self.id,self.x,self.y)
    pos = self.canvas.coords(self.id)
    if pos[1] <= 0:
        self.y = 3
    if self.hit_block(pos) == True:
        self.y = 3
        self.block.id = canvas.create_rectangle(10,10,110,20,fill="yellow")

    if self.hit_block1(pos) == True:
        self.y = 3
        self.block.id1 = canvas.create_rectangle(115,10,215,20,fill="yellow")

    if self.hit_block2(pos) == True:
        self.y = 3
        self.block.id2 = canvas.create_rectangle(220,10,320,20,fill="yellow")

    if self.hit_block3(pos) == True:
        self.y = 3
        self.block.id3 = canvas.create_rectangle(325,10,425,20,fill="yellow")

    if self.hit_block4(pos) == True:
        self.y = 3
        self.block.id4 = canvas.create_rectangle(430,10,530,20,fill="yellow")

    if self.hit_block5(pos) == True:
         self.y = 3
         self.block.id5 = canvas.create_rectangle(100,150,200,160,fill="yellow")
```

```
        if self.hit_block5(pos) == False:
            self.y = -3
            self.block.id5 = canvas.create_rectangle(100,150,200,160,fill="yellow")

        if self.hit_block6(pos) == False:
            self.y = -3
            self.block.id6 = canvas.create_rectangle(350,150,450,160,fill="yellow")

        if self.hit_block6(pos) == True:
            self.y = 3
            self.block.id6 = canvas.create_rectangle(350,150,450,160,fill = "yellow")



        if pos[3] >= self.canvas_height:
            self.hit_bottom = True
        if pos[3] >= self.canvas_height:
            self.y = 3
        if self.hit_paddle(pos) == True:
            self.y = -3
            canvas.delete(block)
        if pos[0] <= 0:
            self.x = 3
        if pos[2] >= self.canvas_width:
            self.x = -3


class Block:
    def __init__(self,canvas,color):
        self.canvas = canvas
        self.id = canvas.create_rectangle(10,10,110,20,fill=color )
        self.id1 = canvas.create_rectangle(115,10,215,20,fill=color)
        self.id2 = canvas.create_rectangle(220,10,320,20,fill=color)
        self.id3 = canvas.create_rectangle(325,10,425,20,fill=color)
        self.id4 = canvas.create_rectangle(430,10,530,20,fill=color)
        self.id5 = canvas.create_rectangle(100,150,200,160,fill=color)
        self.id6 = canvas.create_rectangle(350,150,450,160,fill=color)


        self.x = 0
class Paddle:
    def __init__(self,canvas,color):
        self.canvas = canvas
        self.id = canvas.create_rectangle(0,0,100,10,fill=color)
        self.canvas.move(self.id,230,300)
        self.x = 0
        self.canvas_width = self.canvas.winfo_width()
```

```
        self.started = False
        self.canvas.bind_all("<KeyPress-Left>", self.turn_left)
        self.canvas.bind_all("<KeyPress-Right>", self.turn_right)
        self.canvas.bind_all("<Button-1>", self.start_game)

    def draw(self):
        self.canvas.move(self.id, self.x, 0)
        pos = self.canvas.coords(self.id)
        if pos[0] <= 0:
            self.x = 0
        elif pos[2] >= self.canvas_width:
            self.x = 0
    def turn_left(self, evt):
        pos = self.canvas.coords(self.id)
        if pos[0] >= 0:
            self.x = -3
    def turn_right(self, evt):
        pos = self.canvas.coords(self.id)
        if pos[2] <= self.canvas_width:
            self.x = 3
    def start_game(self,evt):
        self.started = True

paddle = Paddle(canvas, "blue")
block = Block(canvas,"green")
ball = Ball(canvas, paddle, block, 'red')


while 1:

    if ball.hit_bottom == False and paddle.started == True:
        ball.draw()
        paddle.draw()
    if ball.hit_bottom == True:
        time.sleep(1)
        canvas.create_text(270,200, text = "GAME OVER" , font = 28)
        canvas.create_text(270,250, text = " Score = " + str(counter), font = 28)
    if a and b and c and d and e and f and g >= 1:
        time.sleep(1)
        ball.y = 0
        ball.x = 0
        paddle.x = 0
        canvas.create_text(270,200, text = "YOU WIN!" , font = 28)
        canvas.create_text(270,250, text = " Score = " + str(counter), font = 28)
        break
```

```
    tk.update_idletasks()
    tk.update()
    time.sleep(0.01)

mainloop()
```

```
#-------------------------------------------------------------------------------
# Name:        module1
# Purpose:
#
# Author:      Avinash
#
# Created:     01/05/2014
# Copyright:   (c) Avinash 2014
# Licence:     <your licence>
#-------------------------------------------------------------------------------


from Tkinter import *
import random
import time
import sys

win = 0
counter = 0
a = 0
b = 0
c = 0
d = 0
e = 0
f = 0
g = 0

tk = Tk()
tk.title("Bounce!")
tk.resizable(0,0)
tk.wm_attributes("-topmost", 1)
canvas = Canvas(tk, width = 535, height = 400, bd = 0, highlightthickness=0)
canvas.pack()
canvas.create_rectangle(0,0,535,400, fill = "black")

canvas.update()
```

```
class Ball:

    def __init__(self,canvas,paddle,block,color):

        self.canvas = canvas
        self.paddle = paddle
        self.block = block
        self.id = canvas.create_oval(10,10,25,25, fill = color)
        self.canvas.move(self.id, 245,100)
        starts = [-3,-2,-1,1,2,3]
        random.shuffle(starts)
        self.x = starts[0]
        self.y = -1
        self.canvas_height = self.canvas.winfo_height()
        self.canvas_width = self.canvas.winfo_width()
        self.hit_bottom = False
        #self.canvas.create_text( 470, 10, text = "0", anchor = NE, font = ("Courier", 28))
    def score(self):
        global counter
        counter += 1
        tk.title("Bounce! Score = " + str(counter))

    def hit_block(self,pos):
        try:
            block_pos = self.canvas.coords(self.block.id)
            List = [block_pos]
            for i in List:
                if pos[0] >= i[0] and pos[2] <= i[2]:
                    if pos[1] >= i[1] and pos[1] <= i[3]:
                        self.score()
                        global a
                        a += 1
                        return True
            return False
        except:
            return False


    def hit_block1(self,pos):
        try:
            block_pos1 = self.canvas.coords(self.block.id1)
            List = [block_pos1]
            for i in List:
                if pos[0] >= i[0] and pos[2] <= i[2]:
                    if pos[1] >= i[1] and pos[1] <= i[3]:
                        self.score()
```

```
                global b
                b += 1
                return True

        return False
    except:
        return False


def hit_block2(self,pos):
    try:
        block_pos2 = self.canvas.coords(self.block.id2)
        List = [block_pos2]
        for i in List:
            if pos[0] >= i[0] and pos[2] <= i[2]:
                if pos[1] >= i[1] and pos[1] <= i[3]:
                    self.score()
                    global c
                    c += 1
                    return True
        return False
    except:
        return False


def hit_block3(self,pos):
    try:
        block_pos3 = self.canvas.coords(self.block.id3)
        List = [block_pos3]
        for i in List:
            if pos[0] >= i[0] and pos[2] <= i[2]:
                if pos[1] >= i[1] and pos[1] <= i[3]:
                    self.score()
                    global d
                    d += 1
                    return True
        return False
    except:
        return False


def hit_block4(self,pos):
    try:
        block_pos4 = self.canvas.coords(self.block.id4)
        List = [block_pos4]
        for i in List:
            if pos[0] >= i[0] and pos[2] <= i[2]:
```

```
                if pos[1] >= i[1] and pos[1] <= i[3]:
                    self.score()
                    global e
                    e += 1
                    return True

        return False
    except:
        return False

def hit_block5(self,pos):
    try:
        block_pos5 = self.canvas.coords(self.block.id5)
        List = [block_pos5]
        for i in List:
            if (pos[0]  >= i[0] and pos[2] <= i[2]):
                if pos[1] >= i[1] and pos[1] <= i[3]:
                    self.score()
                    global f
                    f += 1
                    return True
            if pos[2] >= i[0] and pos[0] <= i[2]:
                if pos[3] >= i[1] and pos[3] <= i[3]:
                    f += 1
                    return False
    except:
        return 1


def hit_block6(self,pos):
    try:
        block_pos6 = self.canvas.coords(self.block.id6)
        List = [block_pos6]
        for i in List:
            if pos[0] >= i[0] and pos[2] <= i[2]:
                if pos[1] >= i[1] and pos[1] <= i[3]:
                    self.score()
                    global g
                    g += 1
                    return True
            if pos[2] >= i[0] and pos[0] <= i[2]:
                if pos[3] >= i[1] and pos[3] <= i[3]:
                    g += 1
                    return False
    except:
        return 1
```

```
    def hit_paddle(self,pos):
        paddle_pos = self.canvas.coords(self.paddle.id)
        if pos[2] >= paddle_pos[0] and pos[0] <= paddle_pos[2]:
            if pos[3] >= paddle_pos[1] and pos[3] <= paddle_pos[3]:
                self.x += self.paddle.x
                self.y += 1
                self.score()
                return True
        return False
    def draw(self):
        global win
        self.canvas.move(self.id,self.x,self.y)
        pos = self.canvas.coords(self.id)
        if pos[1] <= 0:
            self.y = 3

#########################################################################

        if self.hit_block(pos) == True:
            self.y = 3

            canvas.itemconfig(self.block.id, fill = "yellow")

            global a
            if  a == 2:
                canvas.delete(self.block.id)
                win += 1
#########################################################################

        if self.hit_block1(pos) == True:
            self.y = 3

            canvas.itemconfig(self.block.id1, fill = "yellow")

            global b
            if  b == 2:
                canvas.delete(self.block.id1)
                win += 1

#########################################################################

        if self.hit_block2(pos) == True:
            self.y = 3

            canvas.itemconfig(self.block.id2, fill = "yellow")
```

```
            global c
            if c == 2:
                canvas.delete(self.block.id2)
                win += 1

########################################################################
        if self.hit_block3(pos) == True:
            self.y = 3
            canvas.itemconfig(self.block.id3, fill = "yellow")

            global d
            if d == 2:
                canvas.delete(self.block.id3)
                win += 1


########################################################################
        if self.hit_block4(pos) == True:
            self.y = 3
            canvas.itemconfig(self.block.id4, fill = "yellow")

            global e
            if e == 3:
                canvas.delete(self.block.id4)
                win += 1
########################################################################
#
        global f
        if f == 4:
            canvas.delete(self.block.id5)
            win += 1
        elif self.hit_block5(pos) == True and f <= 2:
             self.y = 3
             canvas.itemconfig(self.block.id5, fill = "yellow")
        elif self.hit_block5(pos) == False and f <= 2:
             self.y = -3
             canvas.itemconfig(self.block.id5, fill = "yellow")



########################################################################
#######
        global g
        if g == 4:
            canvas.delete(self.block.id6)
            win += 1
```

```
        elif self.hit_block6(pos) == True and g <= 2:
            self.y = 3
            canvas.itemconfig(self.block.id6, fill = "yellow")
        elif self.hit_block6(pos) == False and g <= 2:
            self.y = -3
            canvas.itemconfig(self.block.id6, fill = "yellow")



        if pos[3] >= self.canvas_height:
            self.hit_bottom = True
        if pos[1] <= 0:
            self.y = 3
        if self.hit_paddle(pos) == True:
            self.y = -3
            canvas.delete(block)
        if pos[0] <= 0:
            self.x = 3
        if pos[2] >= self.canvas_width:
            self.x = -3


class Block:
    def __init__(self,canvas,color):
        self.canvas = canvas
        self.id = canvas.create_rectangle(10,10,110,20,fill=color )
        self.id1 = canvas.create_rectangle(115,10,215,20,fill=color)
        self.id2 = canvas.create_rectangle(220,10,320,20,fill=color)
        self.id3 = canvas.create_rectangle(325,10,425,20,fill=color)
        self.id4 = canvas.create_rectangle(430,10,530,20,fill=color)
        self.id5 = canvas.create_rectangle(100,150,200,160,fill=color)
        self.id6 = canvas.create_rectangle(350,150,450,160,fill=color)
        self.x = 0
class Paddle:
    def __init__(self,canvas,color):
        self.canvas = canvas
        self.id = canvas.create_rectangle(0,0,100,10,fill=color)
        self.canvas.move(self.id,230,300)
        self.x = 0
        self.canvas_width = self.canvas.winfo_width()
        self.started = False
        self.canvas.bind_all("<KeyPress-Left>", self.turn_left)
        self.canvas.bind_all("<KeyPress-Right>", self.turn_right)
        self.canvas.bind_all("<Button-1>", self.start_game)

    def draw(self):
        self.canvas.move(self.id, self.x, 0)
```

```
        pos = self.canvas.coords(self.id)
        if pos[0] <= 0:
            self.x = 0
        elif pos[2] >= self.canvas_width:
            self.x = 0
    def turn_left(self, evt):
        pos = self.canvas.coords(self.id)
        if pos[0] >= 0:
            self.x = -3
    def turn_right(self, evt):
        pos = self.canvas.coords(self.id)
        if pos[2] <= self.canvas_width:
            self.x = 3
    def start_game(self,evt):
        self.started = True

paddle = Paddle(canvas, "blue")
block = Block(canvas,"green")
ball = Ball(canvas, paddle, block, 'red')


while 1:
    print (win)
    print(a,b,c,d,e,f,g)
    print(ball.canvas_width)

    if ball.hit_bottom == False and paddle.started == True:
       ball.draw()
       paddle.draw()
    if ball.hit_bottom == True:
        time.sleep(1)
        canvas.create_text(270,200, text = "GAME OVER" , font = 28, fill = 'white')
        canvas.create_text(270,250, text = " Score = " + str(counter), font = 28, fill =
'white')

    if (a + b + c + d + e + f + g) > 18:
        time.sleep(1)
        ball.y = 0
        ball.x = 0
        paddle.x = 0
        canvas.create_text(270,200, text = "YOU WIN!" , font = 28, fill="white")
        canvas.create_text(270,250, text = " Score = " + str(counter), font = 28, fill =
"white")
```

```
    tk.update_idletasks()
    tk.update()
    time.sleep(0.01)

mainloop()
```

```
from tkinter import *
import random
import time
tk = Tk()
tk. title("Bounce!")
tk.resizable(0, 0)
tk.wm_attributes("-topmost", 1)
canvas = Canvas(tk, width = 500, height = 500, bd = 0, highlightthickness = 0)
canvas.pack()
tk.update()

class Ball:

    def __init__(self, canvas, paddle, color):
        self.canvas = canvas
        self.paddle = paddle
        self.id = canvas.create_oval(10,10,25,25, fill = color)
        self.canvas.move(self.id,245,100)
        start = [-3,-2,-1,0,1,2,3]
        random.shuffle(start)
        self.x = start[0]
        self.y = -3
        self.canvas_height = self.canvas.winfo_height()
        self.canvas_width = self.canvas.winfo_width()
        self.hit_bottom = False

    def hit_paddle(self, pos):
        paddle_pos = self.canvas.coords(self.paddle.id)
        if pos[2] >= paddle_pos[0] and pos[0] <= paddle_pos[2]:
            if pos[3] >= paddle_pos[1] and pos[3] <= paddle_pos[3]:
                return True
            return False

    def draw(self):
        self.canvas.move(self.id, self.x,self.y)
        pos = self.canvas.coords(self.id)
        print(pos)
```

```
        if pos[1] <= 0:
            self.y = 3
        if pos[3] >= self.canvas_height:
            self.hit_bottom = True
            canvas.create_text(245,100,text="Game Over")
        if pos[0] <= 0:
            self.x = 3
        if pos[2] >= self.canvas_width:
            self.x = -3
        if self.hit_paddle(pos) == True:
            self.y = -3



class Paddle:

    def __init__(self, canvas, color):
        self.canvas = canvas
        self.id = canvas.create_rectangle(0,0,100,10, fill=color)
        self.canvas.move(self.id, 200, 300)
        self.x = 0
        self.canvas_width = self.canvas.winfo_width()
        self.canvas.bind_all('<KeyPress-Left>', self.turn_left)
        self.canvas.bind_all('<KeyPress-Right>', self.turn_right)

    def draw(self):
        self.canvas.move(self.id, self.x, 0)
        pos = self.canvas.coords(self.id)
        if pos[0] <= 0:
            self.x = 0
        if pos[2] >= self.canvas_width:
            self.x = 0
    def turn_left(self,evt):
        self.x = -2
    def turn_right(self,evt):
        self.x = 2

paddle = Paddle(canvas, 'blue')
ball = Ball(canvas, paddle, 'red')


while 1:
    if ball.hit_bottom == False:
        ball.draw()
        paddle.draw()
    tk.update_idletasks()
    tk.update()
```

```
time.sleep(0.01)
```