*ADVANCED PYTHON PROGRAMMING*

*FLAPPY BIRD*

*NAME: PREETI PANDY*

*UID: 19BIT033*

*NAME: PRINCY DENIS PATIL*

*UID: 19BIT035*

**CODE:**

import random # For generating random numbers

import sys # We will use sys.exit to exit the program

import pygame

from pygame.locals import \* # Basic pygame imports

# Global Variables for the game

FPS = 32     # frame per second.it shows that how many frames is rendering in per seconds.in one seconds 32 images will flashes

SCREENWIDTH = 289

SCREENHEIGHT = 511

SCREEN = pygame.display.set\_mode((SCREENWIDTH, SCREENHEIGHT))  #this makes a display surface.Initialize a window or screen for display

GROUNDY = SCREENHEIGHT \* 0.8         # we  took the 80percent of screenheight for groundy/base

GAME\_SPRITES = {}   #image display

GAME\_SOUNDS = {}    #image display

PLAYER = 'C:\\Users\\win10\\Desktop\\flappy\\gallery\\sprites\\bird.png'

BACKGROUND = 'C:\\Users\\win10\\Desktop\\flappy\\gallery\\sprites\\background.jpg'

PIPE = 'C:\\Users\\win10\\Desktop\\flappy\\gallery\\sprites\\pipe.png'

def welcomeScreen():

    #shows welcome images on screen

    #all are typed in integer coz i want only integer value while blitting

    playerx = int(SCREENWIDTH/5) #i want that x position of my  player  should be 1/5 of my screenwidth

    playery = int((SCREENHEIGHT - GAME\_SPRITES['player'].get\_height())/2) # for displaying in center

    #subtracting height of screen with the height of bird.getheight is a function

    messagex = int((SCREENWIDTH - GAME\_SPRITES['message'].get\_width())/2)

    #for displaying msg at center

    messagey = int(SCREENHEIGHT\*0.13) #i can do modification if its improper

    basex = 0 #X=0. BASE IS THAT GROUND

    while True:

        for event in pygame.event.get():

            #PERFORMING event when user click on something

            # if user clicks on cross button, close the game

            if event.type == QUIT or (event.type==KEYDOWN and event.key == K\_ESCAPE):

                pygame.quit()

                sys.exit()

            elif event.type==KEYDOWN and (event.key==K\_SPACE or event.key == K\_UP):

                 # If the user presses space bar  or up key, start the game for them

                 #if someone pressed that specified  button then start the maingame() and  stop displaying welcomescreen

                return

            else:

                 #display the image on screen which takes image and coordinates

                SCREEN.blit(GAME\_SPRITES['background'], (0, 0))

                SCREEN.blit(GAME\_SPRITES['player'], (playerx, playery))

                SCREEN.blit(GAME\_SPRITES['message'], (messagex,messagey ))

                SCREEN.blit(GAME\_SPRITES['base'], (basex, GROUNDY))

                pygame.display.update() # screen will not change until and unless  pygame.display.update() runs

                FPSCLOCK.tick(FPS) #FPS should not exit more than specified.

                # i want to control frames per second of my game and tick it

def mainGame():

    score = 0

    playerx = int(SCREENWIDTH/5) #playerx would seem on  screenwidth/5 origin

    playery = int(SCREENWIDTH/2)

    basex = 0 # groundy

    # Create 2 pipes for blitting on the screen

    newPipe1 = getRandomPipe()   #

    newPipe2 = getRandomPipe()

    # my List of upper pipes

    upperPipes = [

        #my pipe will be at screenwidth+200 position

        {'x': SCREENWIDTH+200, 'y':newPipe1[0]['y']},

        {'x': SCREENWIDTH+200+(SCREENWIDTH/2), 'y':newPipe2[0]['y']},

    ]

    # my List of lower pipes

    lowerPipes = [

        {'x': SCREENWIDTH+200, 'y':newPipe1[1]['y']},

        {'x': SCREENWIDTH+200+(SCREENWIDTH/2), 'y':newPipe2[1]['y']},

    ]

    pipeVelX = -4 #velocity of pipe x .pipe has to move oppsite

    playerVelY = -9 #player will get down at -9 velocity and he'll get acceleration

    playerMaxVelY = 100 #player will get up at 10 velocity . so he should not directly go up quickly

    playerMinVelY = -8

    playerAccY = 1

    playerFlapAccv = -8 # velocity while flapping.(speed of bird while flapping)

    playerFlapped = False # It is true only when the bird is flapping.this is binary variable

    while True:

        for event in pygame.event.get():

            if event.type == QUIT or (event.type == KEYDOWN and event.key == K\_ESCAPE):

                pygame.quit()

                sys.exit()

            if event.type == KEYDOWN and (event.key == K\_SPACE or event.key == K\_UP):

                if playery > 0:

                    playerVelY = playerFlapAccv

                    playerFlapped = True

                    GAME\_SOUNDS['wing'].play()

        crashTest = isCollide(playerx, playery, upperPipes, lowerPipes) # This function will return true if the player is crashed

        if crashTest:

            return

        #check for score

        playerMidPos = playerx + GAME\_SPRITES['player'].get\_width()/2

        for pipe in upperPipes:

            pipeMidPos = pipe['x'] + GAME\_SPRITES['pipe'][0].get\_width()/2

            if pipeMidPos<= playerMidPos < pipeMidPos +4:

                score +=1

                print(f"Your score is {score}")

                GAME\_SOUNDS['point'].play()

        if playerVelY <playerMaxVelY and not playerFlapped:

            playerVelY += playerAccY

        if playerFlapped:

            playerFlapped = False

        playerHeight = GAME\_SPRITES['player'].get\_height()

        playery = playery + min(playerVelY, GROUNDY - playery - playerHeight)

        # move pipes to the left

        for upperPipe , lowerPipe in zip(upperPipes, lowerPipes):

            upperPipe['x'] += pipeVelX

            lowerPipe['x'] += pipeVelX

        # Add a new pipe when the first is about to cross the leftmost part of the screen

        if 0<upperPipes[0]['x']<5:

            newpipe = getRandomPipe()

            upperPipes.append(newpipe[0])

            lowerPipes.append(newpipe[1])

        # if the pipe is out of the screen, remove it

        if upperPipes[0]['x'] < -GAME\_SPRITES['pipe'][0].get\_width():

            upperPipes.pop(0)

            lowerPipes.pop(0)

        # Lets blit our sprites now

        SCREEN.blit(GAME\_SPRITES['background'], (0, 0))

        for upperPipe, lowerPipe in zip(upperPipes, lowerPipes):

            SCREEN.blit(GAME\_SPRITES['pipe'][0], (upperPipe['x'], upperPipe['y']))

            SCREEN.blit(GAME\_SPRITES['pipe'][1], (lowerPipe['x'], lowerPipe['y']))

        SCREEN.blit(GAME\_SPRITES['base'], (basex, GROUNDY))

        SCREEN.blit(GAME\_SPRITES['player'], (playerx, playery))

        myDigits = [int(x) for x in list(str(score))]

        width = 0

        for digit in myDigits:

            width += GAME\_SPRITES['numbers'][digit].get\_width()

        Xoffset = (SCREENWIDTH - width)/2

        for digit in myDigits:

            SCREEN.blit(GAME\_SPRITES['numbers'][digit], (Xoffset, SCREENHEIGHT\*0.12))

            Xoffset += GAME\_SPRITES['numbers'][digit].get\_width()

        pygame.display.update()

        FPSCLOCK.tick(FPS)

def isCollide(playerx, playery, upperPipes, lowerPipes):

    if playery> GROUNDY - 25  or playery<0:

        GAME\_SOUNDS['hit'].play()

        return True

    for pipe in upperPipes:

        pipeHeight = GAME\_SPRITES['pipe'][0].get\_height()

        if(playery < pipeHeight + pipe['y'] and abs(playerx - pipe['x']) < GAME\_SPRITES['pipe'][0].get\_width()):

            GAME\_SOUNDS['hit'].play()

            return True

    for pipe in lowerPipes:

        if (playery + GAME\_SPRITES['player'].get\_height() > pipe['y']) and abs(playerx - pipe['x']) < GAME\_SPRITES['pipe'][0].get\_width():

            GAME\_SOUNDS['hit'].play()

            return True

    return False

def getRandomPipe():

    """

    Generate positions of two pipes(one bottom straight and one top rotated ) for blitting on the screen

    """

    pipeHeight = GAME\_SPRITES['pipe'][0].get\_height() #'tuple' object has no attribute 'get\_height'[[0] is height of pipe]

    offset = SCREENHEIGHT/3

    y2 = offset + random.randrange(0, int(SCREENHEIGHT - GAME\_SPRITES['base'].get\_height()  - 1.2 \*offset))

    #random.randrange() is function which helps to generate random numbers

    pipeX = SCREENWIDTH + 10

    y1 = pipeHeight - y2 + offset

    pipe = [ #this is a list

        {'x': pipeX, 'y': -y1}, #upper Pipe

        {'x': pipeX, 'y': y2} #lower Pipe

    ]

    return pipe

if \_\_name\_\_ == "\_\_main\_\_": #whatever we write in main that only get executes when we are runniung from some program

 # This will be the main point from where our game will start

    pygame.init() # Initialize all pygame's modules

    FPSCLOCK = pygame.time.Clock()  #Creates a new Clock object that can be used to track an amount of time.

    pygame.display.set\_caption('Flappy Bird ') #get the current window caption

    GAME\_SPRITES['numbers'] =(

        # GAME\_SPRITES is a dictionary and

        #convert\_alpha is a function used to render image on scrren  for faster building.change ur pixels and alpha both

        pygame.image.load('C:\\Users\\win10\\Desktop\\flappy\\gallery\\sprites\\1.png').convert\_alpha(),

        pygame.image.load('C:\\Users\\win10\\Desktop\\flappy\\gallery\\sprites\\2.png').convert\_alpha(),

        pygame.image.load('C:\\Users\\win10\\Desktop\\flappy\\gallery\\sprites\\3.png').convert\_alpha(),

        pygame.image.load('C:\\Users\\win10\\Desktop\\flappy\\gallery\\sprites\\4.png').convert\_alpha(),

        pygame.image.load('C:\\Users\\win10\\Desktop\\flappy\\gallery\\sprites\\5.png').convert\_alpha(),

        pygame.image.load('C:\\Users\\win10\\Desktop\\flappy\\gallery\\sprites\\6.png').convert\_alpha(),

        pygame.image.load('C:\\Users\\win10\\Desktop\\flappy\\gallery\\sprites\\7.png').convert\_alpha(),

        pygame.image.load('C:\\Users\\win10\\Desktop\\flappy\\gallery\\sprites\\8.png').convert\_alpha(),

        pygame.image.load('C:\\Users\\win10\\Desktop\\flappy\\gallery\\sprites\\9.png').convert\_alpha(),

    )

    GAME\_SPRITES['message'] =pygame.image.load('C:\\Users\\win10\\Desktop\\flappy\\gallery\\sprites\\message.png').convert\_alpha()

    GAME\_SPRITES['base'] =pygame.image.load('C:\\Users\\win10\\Desktop\\flappy\\gallery\\sprites\\base.png').convert\_alpha()

    GAME\_SPRITES['pipe'] =(pygame.transform.rotate(pygame.image.load(PIPE).convert\_alpha(), 180),

    pygame.image.load(PIPE).convert\_alpha()

    )

    # Game sounds is also a dictionary

    #pygame module for loading and playing sounds

    GAME\_SOUNDS['die'] = pygame.mixer.Sound('C:\\Users\\win10\\Desktop\\flappy\\gallery\\audio\\die.WAV')

    GAME\_SOUNDS['hit'] = pygame.mixer.Sound('C:\\Users\\win10\\Desktop\\flappy\\gallery\\audio\\hit.mp3')

    GAME\_SOUNDS['point'] = pygame.mixer.Sound('C:\\Users\\win10\\Desktop\\flappy\\gallery\\audio\\point.mp3')

    GAME\_SOUNDS['swoosh'] = pygame.mixer.Sound('C:\\Users\\win10\\Desktop\\flappy\\gallery\\audio\\swoosh.mp3')

    GAME\_SOUNDS['wing'] = pygame.mixer.Sound('C:\\Users\\win10\\Desktop\\flappy\\gallery\\audio\\wing.mp3')

    GAME\_SPRITES['background'] = pygame.image.load(BACKGROUND).convert() #BACKGROUND IS A GLOBAL VARIABLES.using convert() only pixels of image gets change for blitting(displaying image on screen quickly)

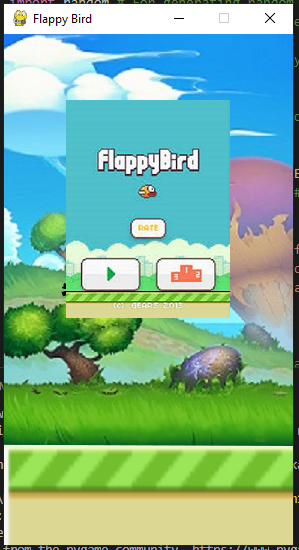
    GAME\_SPRITES['player'] = pygame.image.load(PLAYER).convert\_alpha() #alphas of pixels get change

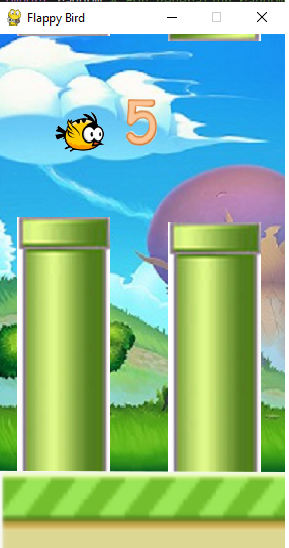
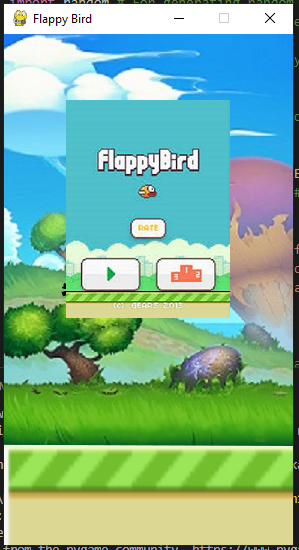
    while True: #game loop which ,must go on

        welcomeScreen() # Shows welcome screen to the user until he presses a button. #welcomeScreen () returns when game starts

        mainGame() # This is the main game function .returns when game gets over

**OUTPUT:**

**SCORE**

