ESTI - Escola Superior da Tecnologia da Informação

EDC - Graduação em Engenharia de Computação

Fundamentos de programação com Python

TP3

Aluno: Eloy Francisco Barbosa

Professor: Cassius Figueiredo

Data: 30/11/2018

Sumário

#1.
Usando Python, faça o que se pede (código e printscreen):
#a) Crie uma lista vazia;
#b) Adicione os elementos: 1, 2, 3, 4 e 5, usando append();
#c) Imprima a lista;
#d) Agora, remova os elementos 3 e 6 (não esqueça de checar se eles estão na lista);
#e) Imprima a lista modificada;
#f) Imprima também o tamanho da lista usando a função len();
#g) Altere o valor do último elemento para 6 e imprima a lista modificada.

```
lista = []
for i in range(1, 6):
    lista.append(i)

print(lista)
print()
print('Agora vamos remover o elemento 3 e 6 da lista caso eles
existam na lista.')
print()
if 3 in lista:
    lista.remove(3)
    print('O elemento 3 foi excluido da lista')
    print()

else:
    print('O elemento 3 não existe na lista')
    print()

if 6 in lista:
    lista.remove(6)
    print('O elemento 6 foi excluido da lista')
    print()
else:
    print('O elemento 6 não existe na lista')
    print()


print(lista)
tamanho=len(lista)
print()
print ('O tamanho da lista é de', tamanho, 'elementos')
print()
print('Agora vamos alterar o valor do ultimo elemento da lista
para 6.')

lista[-1]=6
print()
print(lista)
```

#2.
Escreva um programa em Python que leia um vetor de 5 números inteiros e mostre-os. (código)

```
print('Insira 5 números para criarmos nosso vertor')

vetor=[]
i=0

for i in range (0, 5):
    n=int(input('Insira os numeros para o vetor:'))
    vetor.append(n)

print()
print('Nosso vetor ficou da seguinte forma:', vetor)
```

#3.
Escreva um programa em Python que leia um vetor de 10 palavras e mostre-as na ordem inversa de leitura. (código)

```
print('Insira 10 palavras para criarmos um vetor')
print()

vetor10 = []

for i in range (0, 10):
    palavra=str(input('Insira as palavras para o vetor:'))
    vetor10.append(palavra)

vetor10_reverso=vetor10[::-1]
print()
print('O nosso vetor de 10 palavras na ordem inversa de leitura
ficaria assim:', vetor10_reverso)
```

#4.
Escreva um programa em Python que leia um vetor de números de tamanho t. Leia t previamente. Em seguida, faça seu programa verificar quantos números iguais a 0 existem nele. (código)

```
print('Nessa atividade vamos criar um vetor somente de números
inteiros com o tamanho a ser definido.\n Primeiro vamos inserir
o número tambem inteiro que vai definir o tamanho do nosso
vetor.\n Depois de acordo com o tamanho do nosso vetor iremos
definir os números que irão definir o vetor um por um.')
print()

vetor=[]
while True:
    try:
        t = int(input('Primeiramente insira o tamanho do vetor:
'))
        print()
```

```
        print('Muito bom nosso vetor terá', t, 'números
inteiros')
        print()

        for i in range (0, t):
            n=int(input('Insira os numeros para o vetor:'))
            vetor.append(n)


    except ValueError:
        print("Favor digitar um núnero inteiro")
    else:
        break

print()

print('Nosso vetor ficou da seguinte forma:', vetor)

z=vetor.count(0)
print()
print('O número 0 apareceu no vetor ', z, ' vezes')
```

#5.
Escreva um programa em Python que leia nomes de alunos e suas alturas em metros até que um nome de aluno seja o código de saída “Sair”. O programa deve possuir uma função que indica todos os alunos que tenham altura acima da média (a média aritmética das alturas de todos os alunos lidos). (código)

```
print('Nesta atividade vamos inserir o nome dos alunos e em
seguida a sua altura em metros e depois vamos calcular  a média
da altura dos alunos informados e retornar quais são os alunos
que tem a altura acima da média.\n OBS.:Quando quiser parar de
inserir nomes é só digitar a palavra "Sair".')

temp = []
final = []
soma = 0
acima = []

temp.append(str(input("Insira o nome do primeiro aluno: ")))
while temp[0] != 'sair':
    temp.append(float(input('Agora insira sua altura:')))
    final.append(temp[:])
    temp.clear()
    temp.append(str(input("Insira o nome do próximo aluno: ")))

tamanho_final = len(final)

for a in range (0, tamanho_final):
    idade=(final[a][1])
    soma=soma+(final[a][1])

media= soma/tamanho_final
```

```
for n in range (0, tamanho_final):
    if final[n][1] > media:
       acima.append(final[n][0])

str_acima = ', '.join(acima)

print(f'A altura média dos alunos informados é {round(media,
2)}m, os alunos que tem altura acima da média são: {str_acima}')
```

#6.
Escreva um programa em Python que leia diversas frases até a palavra “Sair” ser digitada. Indique quais frases apresentam a palavra “eu”. (código)

```
temp = []
frases = []
eu_real=0

print('Nessa atividade vamos inserir varias frases e depois
vamos indicar em qual frase a palavra "eu" foi utilizada.')

temp.append(str(input("Insira a primeira frase ")))
while temp[-1] != 'sair':
    temp.append(str(input("Insira a proxima frase: ")))
temp.remove('sair')

tamanho=len(temp)

for c in range (0, tamanho):
    if temp[c].count("eu") !=0:
        frases.append(temp[c])

num_frases=len(frases)

if num_frases !=0:
    print()
    print('As frases na qual a palavra eu foi utilizada são as
seguintes:')
    for f in range (0, num_frases):
        print(frases[f])
else:
    print()
    print('Nenhuma Frase com a palavra eu foi inserida')
```

#7.
Escreva um programa em Python que realiza operações de inclusão e remoção em listas. Seu programa deve perguntar ao usuário qual operação deseja fazer: (código)
#a. Mostrar lista;
#b. Incluir elemento;
#c. Remover elemento;
#d. Apagar todos os elementos da lista.
#Se a opção for a alternativa (a), seu programa deve apenas mostrar o conteúdo da lista. Se a opção for a alternativa (b), seu programa deve pedir o valor do elemento a ser incluído. Se a opção for a alternativa (c), seu programa deve pedir o valor do elemento a ser removido. E se a opção for a alternativa (d), deve-se apenas exibir se a operação foi concluída.

```
lista=['Primavera', 'Verão', 'Inverno', 'Outono']

print('Nessa atividade temos uma lista pré definida, logo abaixo
será mostrada uma lista de opções, favor informar oque deseja
fazer!')
print()
print(' a. Mostrar Lista.\n b. Incluir Elemento.\n c. Remover
Elemento.\n d. Apagar todos os elementos da lista.')
print()

opcao=str(input('Favor escolha a opção desejada:')).upper()
while opcao !='A' and opcao !='B' and opcao !='C' and opcao
!='D':
    opcao=str(input('Favor escolha uma opção válida:')).upper()

if opcao == "A":
    print()
    print('Esta é nossa lista!')
    print(lista)
elif opcao == "B":
    print()
    print('Você escolheu inserir um elemento na lista.')
    print()
    elemento=input('Digite o elemento que deseja inserir na
lista:')
    lista.append(elemento)

elif opcao == "C":
    print()
    print('Você escolheu remover um elemento da lista')
    remover=input('Insira o elemento que deseja remover da
lista.')
    while remover not in lista:
        remover=input('Favor inserir um elemento que esteja na
lista')
    lista.remove(remover)
    print(lista)

elif opcao == "D":
    lista.clear()
    print()
```

```
    print('Você removeu todos os elementos da lista.')
else:
    print()
    print('Você não escolheu uma opção válida, execute
novamente.')
```

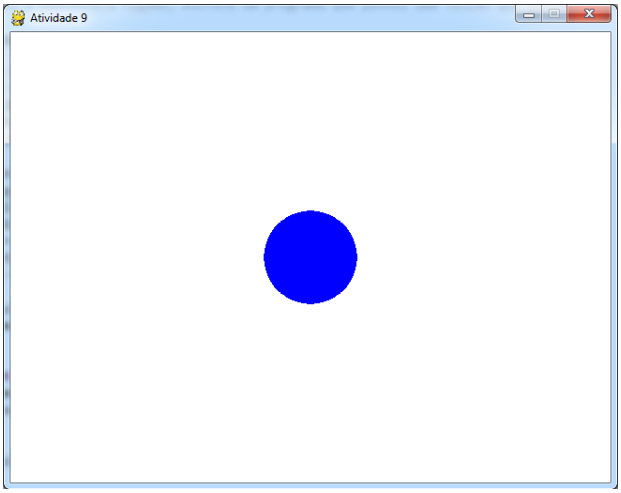
#8.
Faça uma função um programa em Python que simula um lançamento de dados. Lance o dado 100 vezes e armazene os resultados em um vetor. Depois, mostre quantas vezes cada valor foi conseguido. Dica: use um vetor de contadores (1-6) e uma função do módulo 'random' de Python para gerar números aleatórios, simulando os lançamentos dos dados. (código)
import random

```
print('Nesta atividade vamos simular o lançamento de um dado 100
vezes e depois vamos mostrar quantas vezes cada valor foi
conseguido durante a simulação.')
print()
dado = []
dado_lancado=[]
for d in range (1, 7):
    dado.append(d)

for c in range (0, 100):
    lancado=random.choice(dado)
    dado_lancado.append(lancado)

for i in range (1, 7):
    print('O número', i, 'foi conseguido',
dado_lancado.count(i), 'vezes.')
```

#9.
Usando a biblioteca Pygame, escreva um programa que possui uma função que desenha um círculo azul de 100 px de diâmetro no centro da tela. (código e printscreen)

```
import pygame

branco = (255,255,255)
vermelho = (255,0,0)
verde = (0,255,0)
azul = (0,0,255)
preto = (0, 0, 0)

pygame.init()
tela = pygame.display.set_mode([640, 480])
tela.fill(branco)
pygame.display.set_caption("Atividade 9")
relogio = pygame.time.Clock()
relogio.tick(27)
terminou = False

def circulo_azul():
    pygame.draw.circle(tela, azul, (320, 240), 50)
```

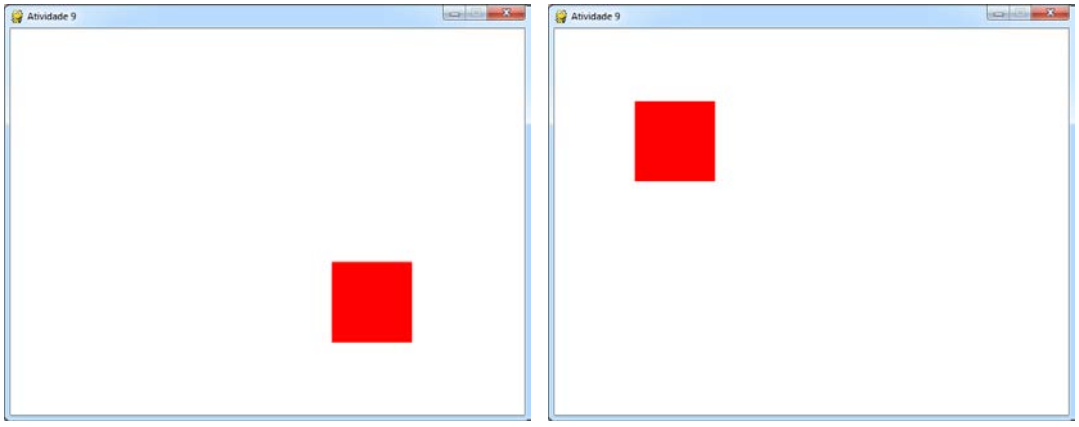
```
while not terminou:
    pygame.display.update()
    for event in pygame.event.get():
          if event.type == pygame.QUIT:
                terminou = True
    circulo_azul()


pygame.display.quit()
pygame.quit()
```



#10.
Usando a biblioteca Pygame, escreva um programa que possui uma função que desenha um quadrado vermelho de 100 px de lado no centro da tela. O quadrado deve ser capaz de se movimentar vertical e horizontalmente através de teclas do computador. Pode ser 'a','s','d','w' ou as setas do teclado. (código e printscreen)

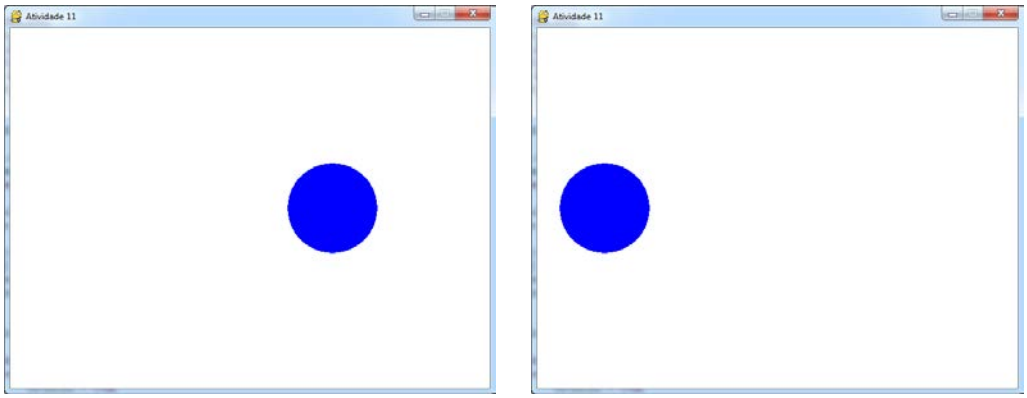
```
import pygame

branco = (255,255,255)
vermelho = (255,0,0)
verde = (0,255,0)
azul = (0,0,255)
preto = (0, 0, 0)

pos_x=10
pos_y=10


pygame.init()
tela = pygame.display.set_mode([640, 480])

pygame.display.set_caption("Atividade 9")
relogio = pygame.time.Clock()
```

```
terminou = False

def quadrado_vermelho():
    pygame.draw.rect(tela, vermelho, (pos_x, pos_y, 100, 100))


while not terminou:

    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            terminou = True

        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_LEFT:
                pos_x += -10
            if event.key == pygame.K_RIGHT:
                pos_x += 10
            if event.key == pygame.K_UP:
                pos_y += -10
            if event.key == pygame.K_DOWN:
                pos_y += 10

    quadrado_vermelho()

    pygame.display.update()
    relogio.tick(50)
    tela.fill(branco)
pygame.display.quit()
pygame.quit()
```



#11.
Usando a biblioteca Pygame, escreva um programa que possui uma função que desenha um círculo azul de 100 px de diâmetro no centro da tela que se move da esquerda para a direita. Sempre que chegar na extremidade direita, o círculo deve voltar à extremidade esquerda, retomando o movimento da esquerda para a direita. (código e printscreen)

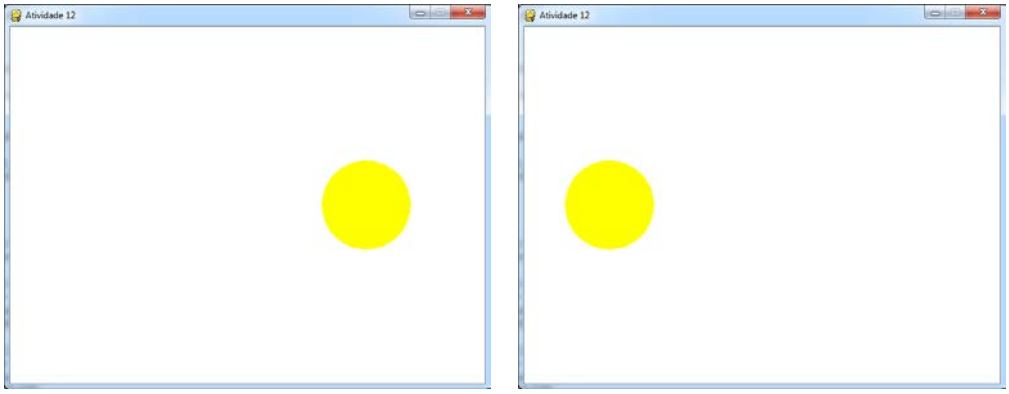
```
import pygame

branco = (255,255,255)
vermelho = (255,0,0)
verde = (0,255,0)
azul = (0,0,255)
preto = (0, 0, 0)
pos_bola = [320, 240]

pygame.init()
tela = pygame.display.set_mode([640, 480])

pygame.display.set_caption("Atividade 11")
relogio = pygame.time.Clock()
terminou = False

def circulo_azul():
    pygame.draw.circle(tela, azul, pos_bola, 60)

def movimento():
    pos_bola[0] += 5
    while pos_bola[0] == 640:
        pos_bola[0]=0


while not terminou:

    for event in pygame.event.get():
          if event.type == pygame.QUIT:
                  terminou = True

    circulo_azul()
    movimento()


    pygame.display.update()
    relogio.tick(60)
    tela.fill(branco)
pygame.display.quit()
pygame.quit()
```



#12.
Usando a biblioteca Pygame, escreva um programa que possui uma função que desenha um círculo amarelo de 100 px de diâmetro no centro da tela que se move sempre em velocidade permanente na direção que o usuário indicar através das teclas. Similar ao item anterior, sempre que chegar em uma extremidade, o círculo deve voltar à extremidade oposta e continuar o com a última direção que o usuário indicou. (código e printscreen)

```
import pygame

branco = (255,255,255)
vermelho = (255,0,0)
verde = (0,255,0)
azul = (0,0,255)
preto = (0, 0, 0)
amarelo = (255, 255, 0)

pos_bola = [320, 240]

pygame.init()
tela = pygame.display.set_mode([640, 480])

pygame.display.set_caption("Atividade 12")
relogio = pygame.time.Clock()
terminou = False
esquerda = True
direita= True

def circulo_azul():
    pygame.draw.circle(tela, amarelo, pos_bola, 60)

def move_direita():
    pos_bola[0] += +5
    while pos_bola[0] == 640:
        pos_bola[0]=0

def move_esquerda():
    pos_bola[0] += -5
    while pos_bola[0] == 0:
        pos_bola[0]=640
```

```
while not terminou:

    for event in pygame.event.get():
        if event.type == pygame.QUIT:
              terminou = True

        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_LEFT:
                direita = not direita

            if event.key == pygame.K_RIGHT:
                esquerda = not esquerda



    if not direita:
        move_esquerda()

    if not esquerda:
        move_direita()

    circulo_azul()


    pygame.display.update()
    relogio.tick(27)
    tela.fill(branco)
pygame.display.quit()
pygame.quit()
```

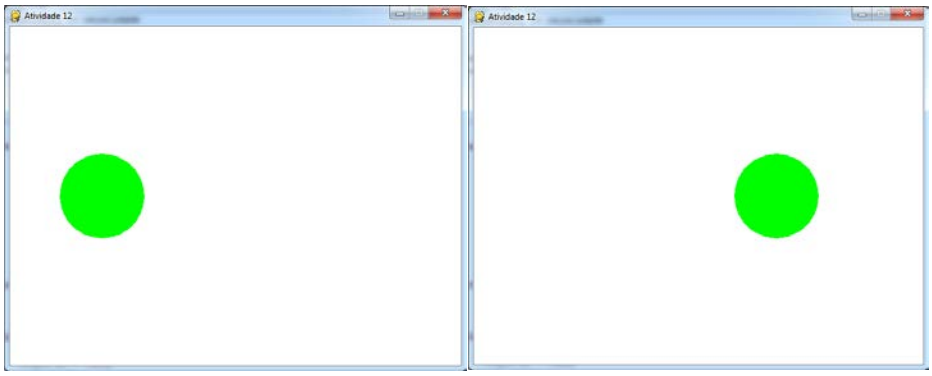


#13.
Usando a biblioteca Pygame, escreva um programa que possui uma função que desenha um círculo verde de 100 px de diâmetro no centro da tela que se inicie o movimento da esquerda para a direita. Sempre que chegar em alguma extremidade, o círculo deve trocar a direção e aumentar a velocidade em 1. (código e printscreen)

```
import pygame
```

```
branco = (255,255,255)
vermelho = (255,0,0)
verde = (0,255,0)
azul = (0,0,255)
preto = (0, 0, 0)
amarelo = (255, 255, 0)
teste = [320, 240]
pos_bola = [320, 240]

pygame.init()
tela = pygame.display.set_mode([640, 480])

pygame.display.set_caption("Atividade 12")
relogio = pygame.time.Clock()
terminou = False
esquerda = False
direita= True
velocidade = 5


def circulo_azul():
    pygame.draw.circle(tela, verde, pos_bola, 60)

def move_direita():
    pos_bola[0] += velocidade


def move_esquerda():
    pos_bola[0] += -velocidade


while not terminou:

    for event in pygame.event.get():
        if event.type == pygame.QUIT:
              terminou = True

        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_LEFT:
                direita = not direita

            if event.key == pygame.K_RIGHT:
                esquerda = not esquerda

    if pos_bola[0] >= 640:
        direita = False
        esquerda = True
        velocidade += +1
    elif pos_bola[0] <= 0:
        direita = True
        esquerda = False
        velocidade += +1

    if not direita:
        move_esquerda()
```

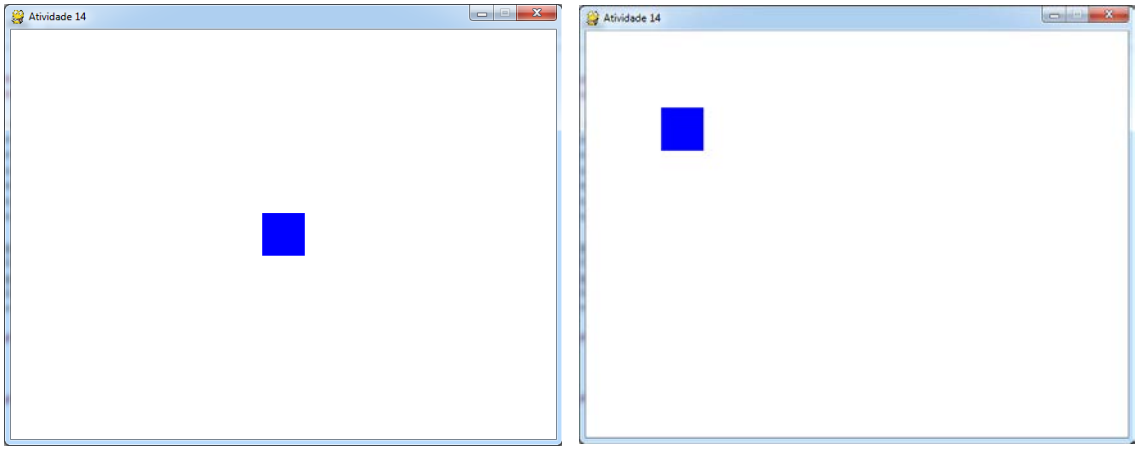
```
    if not esquerda:
        move_direita()

    circulo_azul()


    pygame.display.update()
    relogio.tick(27)
    tela.fill(branco)

while pos_bola[0] == 0:
    direita = True
pygame.display.quit()
pygame.quit()
```



#14.
Usando a biblioteca Pygame, escreva um programa que possui uma função que desenha um quadrado de tamanho 50 no centro da tela. Quando o usuário clicar em alguma área da janela, o quadrado deve se mover para a posição clicada. (código e printscreen)

```
import pygame

branco = (255,255,255)
vermelho = (255,0,0)
verde = (0,255,0)
azul = (0,0,255)
preto = (0, 0, 0)
pos_inicial = (295, 215)
posicao = (295, 215)

pygame.init()
tela = pygame.display.set_mode([640, 480])
pygame.display.set_caption("Atividade 14")
relogio = pygame.time.Clock()
terminou = False

def quadrado():
    pygame.draw.rect(tela, azul, (pos_inicial[0],
pos_inicial[1], 50, 50))


def mudar_posicao():
    pos_inicial=posicao
```

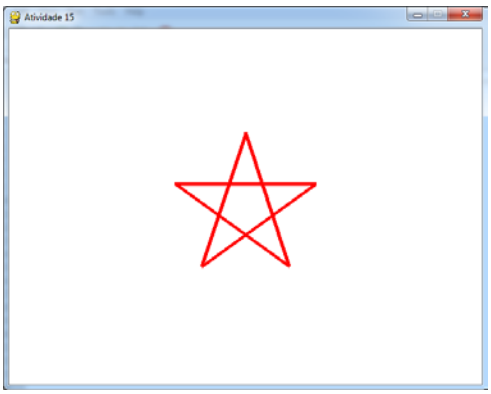
```
while not terminou:

    for event in pygame.event.get():
        if event.type == pygame.QUIT:
              terminou = True

        if event.type == pygame.MOUSEBUTTONDOWN:
            posicao = pygame.mouse.get_pos()
            pos_inicial=posicao


    quadrado()

    pygame.display.update()
    relogio.tick(27)
    tela.fill(branco)
pygame.display.quit()
pygame.quit()
```



#15.
Usando a biblioteca Pygame, escreva um programa que possui uma função que desenha na tela um estrela de 5 pontas no tamanho que preferir. (código e printscreen)

```
import pygame
import math

branco = (255,255,255)
vermelho = (255,0,0)
verde = (0,255,0)
azul = (0,0,255)
preto = (0, 0, 0)

pygame.init()
tela = pygame.display.set_mode([640, 480])
tela.fill(branco)
pygame.display.set_caption("Atividade 15")
```

```
relogio = pygame.time.Clock()
relogio.tick(27)
terminou = False


def estrela():

    pontos = []

    for angulo in range(90, 360 + 90, 360 // 5):
        x_ponto = 320 + math.cos(math.radians(angulo)) * 100
        y_ponto = 240 - math.sin(math.radians(angulo)) * 100
        pontos.append((x_ponto, y_ponto))

    for index, ponto in enumerate(pontos):
        if (index + 2 >= len(pontos)):
            prox_ponto = pontos[abs(index + 2 - len(pontos))]
        else:
            prox_ponto = pontos[index + 2]

        pygame.draw.line(tela, vermelho, ponto, prox_ponto, 5)


while not terminou:
    pygame.display.update()
    for event in pygame.event.get():
          if event.type == pygame.QUIT:
                terminou = True
    estrela()


pygame.display.quit()
pygame.quit()
```

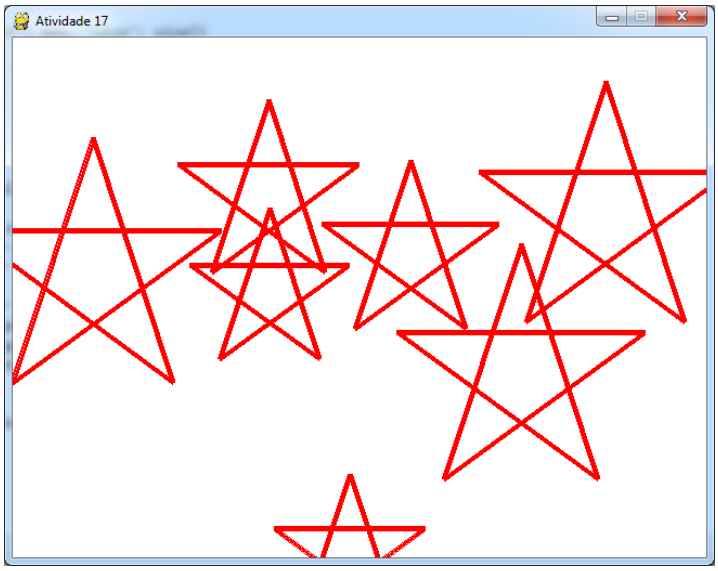


#16.
Usando a biblioteca Pygame, escreva um programa que desenha na tela estrelas de 5 pontas de tamanhos aleatórios a cada vez que o usuário clicar na tela. A ponta superior da estrela deve estar situada onde o usuário clicou. (código e printscreen)

```
import pygame
import math
import random
```

```
branco = (255,255,255)
vermelho = (255,0,0)
verde = (0,255,0)
azul = (0,0,255)
preto = (0, 0, 0)
pos_inicial = (295, 215)
posicao = (295, 215)

pygame.init()
tela = pygame.display.set_mode([640, 480])
pygame.display.set_caption("Atividade 17")
relogio = pygame.time.Clock()
terminou = False

tamanho = random.randint(50, 150)

estrelas = []

def star(estrela):
    pontos = []

    for angulo in range(90, 360 + 90, 360 // 5):
        x_ponto = estrela["pos"][0] +
math.cos(math.radians(angulo)) * estrela["size"]
        y_ponto = estrela["pos"][1] -
math.sin(math.radians(angulo)) * estrela["size"]
        pontos.append((x_ponto, y_ponto))

    for index, ponto in enumerate(pontos):
        if (index + 2 >= len(pontos)):
            prox_ponto = pontos[abs(index + 2 - len(pontos))]
        else:
            prox_ponto = pontos[index + 2]

        pygame.draw.line(tela, vermelho, ponto, prox_ponto, 5)


def add_star(pos):
    size = random.randint(50, 150)
    pos = list(pos)
    pos[1] += size
    estrelas.append({"pos": pos, "size": size})

def mudar_posicao():
    pos_inicial=posicao


while not terminou:
    relogio.tick(27)
    for estrela in estrelas:
        star(estrela)
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
              terminou = True
```

```
        if event.type == pygame.MOUSEBUTTONDOWN:
            add_star(pygame.mouse.get_pos())
            posicao = pygame.mouse.get_pos()
            pos_inicial=posicao


    pygame.display.update()

    tela.fill(branco)
pygame.display.quit()
pygame.quit()
```



#17.
Usando a biblioteca Pygame, escreva um programa que implemente o jogo "Pong" (visto no curso), com uma modificação. Tal modificação consiste em incluir o aumento da velocidade da bola. O aumento será feito de maneira gradual, isto é, cada 10 vezes que a bola bater na paleta do jogador1 a velocidade aumenta em 1. (código e printscreen)

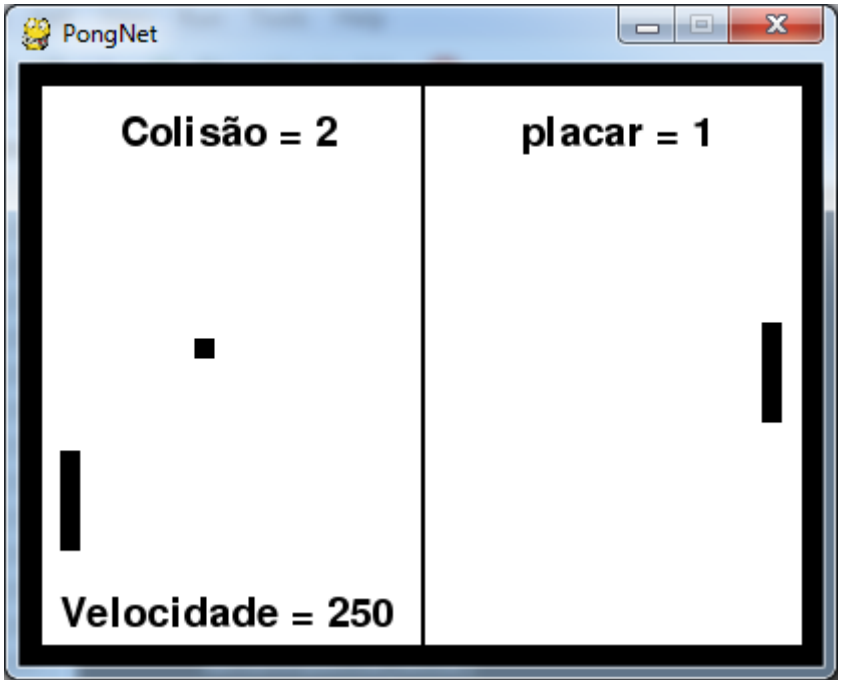
```
import pygame, sys
from pygame.locals import *

# CONSTANTES
# Constantes para o tamanho da tela
LARGURA_TELA = 400
ALTURA_TELA = 300
# Valores para o desenho das paletas e do fundo
LARGURA_LINHA = 10
PALETA_TAMANHO = 50
PALETAOFFSET = 20
# Cores
PRETO      = (0, 0, 0)
BRANCO = (255,255,255)
#VARIAVEIS
# Será utilizado para a velocidade do jogo
```

```
fps = 250
colisao1=0
loop=0

# Função para desenhar a arena
def desenhaArena():
    TELA.fill(BRANCO)
    # Desenha a quadra
    pygame.draw.rect(TELA, PRETO, ((0,0),
(LARGURA_TELA,ALTURA_TELA)), LARGURA_LINHA*2)
    # Desenha a linha no centro
    pygame.draw.line(TELA, PRETO, ((LARGURA_TELA//2),0),
((LARGURA_TELA//2),ALTURA_TELA), (LARGURA_LINHA//4))

# Função para desenhar a paleta
def desenhaPaleta(paleta):
    #Impede da paleta ir  além da borda do fundo
    if paleta.bottom > ALTURA_TELA - LARGURA_LINHA:
        paleta.bottom = ALTURA_TELA - LARGURA_LINHA
    #Impede da paleta ir  além da borda do topo
    elif paleta.top < LARGURA_LINHA:
        paleta.top = LARGURA_LINHA
        #Desenha a paleta
    pygame.draw.rect(TELA, PRETO, paleta)


# Função para desenhar a bola
def desenhaBola(bola):
    pygame.draw.rect(TELA, PRETO, bola)

#altera a direção da bola e retorna ela
def moveBola(bola, bolaDirX, bolaDirY):
        bola.x += bolaDirX
        bola.y += bolaDirY
        return bola

# Verifica por colisão com as bordas
# Retorna uma nova posição caso exista colisão
def verificaColisao(bola, bolaDirX, bolaDirY):
    if bola.top == (LARGURA_LINHA) or bola.bottom ==
(ALTURA_TELA - LARGURA_LINHA):
            bolaDirY = bolaDirY * -1
    if bola.left == (LARGURA_LINHA) or bola.right ==
(LARGURA_TELA - LARGURA_LINHA):
            bolaDirX = bolaDirX * -1
    return bolaDirX, bolaDirY

# Rotina de IA para o NPC.
def inteligenciaArtificial(bola, bolaDirX, paleta2):
# Movimentar a paleta quando a bola vem em direção da paleta
    if bolaDirX == 1:
            if paleta2.centery < bola.centery:
                paleta2.y += 1
            else:
                paleta2.y -=1
    return paleta2
```

```
#Verifica a colisão da bola com a paleta1 ou paleta2
def verificaColisaoBola(bola, paleta1, paleta2, bolaDirX):

    if bolaDirX == -1 and paleta1.right == bola.left and
paleta1.top < bola.top and paleta1.bottom > bola.bottom:
        return -1

    elif bolaDirX == 1 and paleta2.left == bola.right and
paleta2.top < bola.top and paleta2.bottom > bola.bottom:
        return -1
    else:
        return 1


#Verifica se o jogador fez ponto e retorna o novo valor do
placar
def verificaPlacar(paleta1, bola, placar, bolaDirX):
    #zera a contagem se a bola acerta a borda do jogador
    if bola.left == LARGURA_LINHA:
        return 0
    #1 ponto por acertar a bola
    elif bolaDirX == 1 and paleta1.right == bola.left and
paleta1.top < bola.top and paleta1.bottom > bola.bottom:
        placar += 1
        return placar
    #10 pontos se vender a paleta do computador
    elif bola.right == LARGURA_TELA - LARGURA_LINHA:
        placar += 10
        return placar
    #retorna o mesmo placar se nenhum ponto foi adicionado
    else: return placar

def verificacolisao(paleta1, bola, colisao1, bolaDirX):

    if bolaDirX == 1 and paleta1.right == bola.left and
paleta1.top < bola.top and paleta1.bottom > bola.bottom:
        colisao1 += 1
        return colisao1
    else: return colisao1



def desenhaPlacar(placar):
    resultadoSurf = BASICFONT.render('placar = %s' %(placar),
True, PRETO)
    resultadoRect = resultadoSurf.get_rect()
    resultadoRect.topleft = (LARGURA_TELA - 150, 25)
    TELA.blit(resultadoSurf, resultadoRect)

def desenhaColisoes(colisao):
    resultadoSurf = BASICFONT.render('Colisão = %s' %(colisao),
True, PRETO)
    resultadoRect = resultadoSurf.get_rect()
    resultadoRect.topleft = (LARGURA_TELA - 350, 25)
    TELA.blit(resultadoSurf, resultadoRect)
```

```
def desenhavelocidade(fps):
    resultadoSurf = BASICFONT.render('Velocidade = %s' %(fps),
True, PRETO)
    resultadoRect = resultadoSurf.get_rect()
    resultadoRect.topleft = (LARGURA_TELA - 380, 265)
    TELA.blit(resultadoSurf, resultadoRect)



#funcao Principal
def main():
    pygame.init()

    #Informação da fonte
    global BASICFONT, BASICFONTSIZE
    BASICFONTSIZE = 20
    BASICFONT = pygame.font.Font('freesansbold.ttf',
BASICFONTSIZE)

    global TELA

    FPSCLOCK = pygame.time.Clock()
    TELA = pygame.display.set_mode((LARGURA_TELA,ALTURA_TELA))
    pygame.display.set_caption('PongNet')
    colisao10=False

    #Iniciando as variáveis nas posições iniciais
    #Estas variáveis serão alteradas ao longo da execução
    bolaX = LARGURA_TELA//2 - LARGURA_LINHA//2
    bolaY = ALTURA_TELA//2 - LARGURA_LINHA//2
    jogadorUm_posicao = (ALTURA_TELA - PALETA_TAMANHO) //2
    jogadorDois_posicao = (ALTURA_TELA - PALETA_TAMANHO) //2
    placar = 0
    colisao1 = 0
    fps = 250


    #altera a posição da bola
    bolaDirX = -1
    bolaDirY = -1

    #Criando os retangulos para a bola e paletas.
    paleta1 =
pygame.Rect(PALETAOFFSET,jogadorUm_posicao,LARGURA_LINHA,PALETA_
TAMANHO)
    paleta2 = pygame.Rect(LARGURA_TELA - PALETAOFFSET -
LARGURA_LINHA, jogadorDois_posicao,
LARGURA_LINHA,PALETA_TAMANHO)
    bola = pygame.Rect(bolaX, bolaY, LARGURA_LINHA,
LARGURA_LINHA)

    #Desenhando as posições iniciais da Arena
    desenhaArena()
    desenhaPaleta(paleta1)
    desenhaPaleta(paleta2)
```

```
    desenhaBola(bola)

    pygame.mouse.set_visible(0)

    while True: #Loop principal
        for event in pygame.event.get():
                if event.type == QUIT:
                   pygame.quit()
                   sys.exit()
                elif event.type == MOUSEMOTION:
                    mouseX, mouseY = event.pos
                    paleta1.y = mouseY

        if colisao1 == 3:
            colisao10=True


        if colisao10 == True:
            fps+=10
            colisao10=False
            colisao1=0

        desenhaArena()
        desenhaPaleta(paleta1)
        desenhaPaleta(paleta2)
        desenhaBola(bola)


        bola = moveBola(bola, bolaDirX, bolaDirY)
        bolaDirX, bolaDirY = verificaColisao(bola, bolaDirX,
bolaDirY)
        bolaDirX = bolaDirX * verificaColisaoBola(bola, paleta1,
paleta2, bolaDirX)
        paleta2 = inteligenciaArtificial(bola, bolaDirX,
paleta2)

        placar = verificaPlacar(paleta1, bola, placar, bolaDirX)
        colisao1 = verificacolisao(paleta1, bola, colisao1,
bolaDirX)
        desenhaPlacar(placar)
        desenhaColisoes(colisao1)
        desenhavelocidade(fps)


        pygame.display.update()
        FPSCLOCK.tick(fps)


if __name__=='__main__':
 main()
```



Como o enunciado pedia para aumentar o a velocidade da bola a gradativamente a cada 10 colisões com a paleta jogador1 implementei um contador de colisões na paleta 1 e a cada 10 colisões o contador zera e aumenta 10 na velocidade (no enunciado pede 1 mas achei muito pouco)

#18.
Usando a biblioteca Pygame, escreva um programa que implemente o jogo "Pong" alterado na questão anterior e que adicione uma nova modificação. Tal modificação consiste em aumentar o ganho de pontos para cada vez que a bola encostar na paleta do jogador1. O aumento da pontuação será também realizada de maneira gradual, porém somente a cada 2 aumentos da velocidade da bola, isto é 1 ponto será atribuído ao total de pontos que o jogador ganha a cada vez que a bola bate na paleta. (código e printscreen)

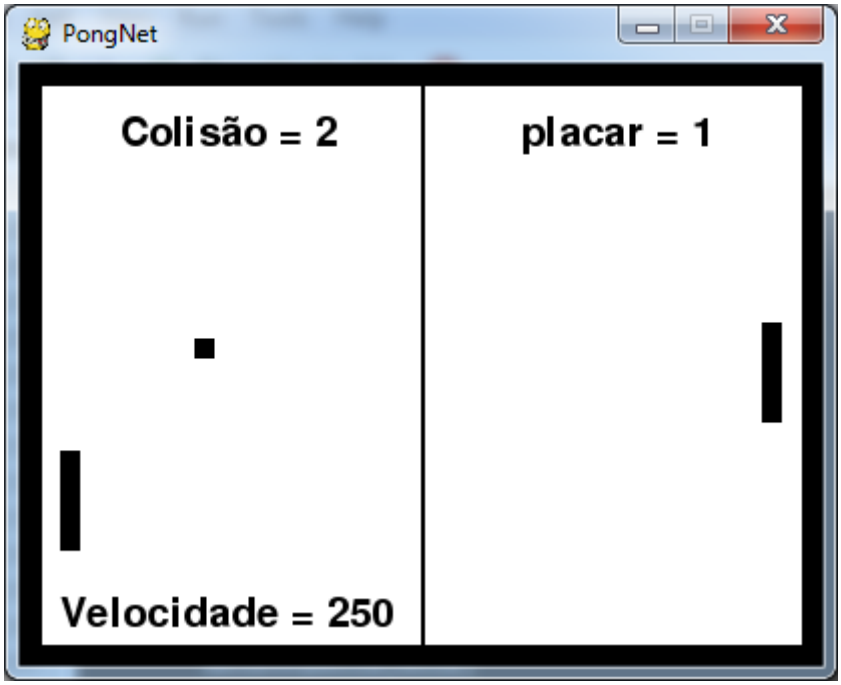
```
import pygame, sys
from pygame.locals import *

# CONSTANTES
# Constantes para o tamanho da tela
LARGURA_TELA = 400
ALTURA_TELA = 300
# Valores para o desenho das paletas e do fundo
LARGURA_LINHA = 10
PALETA_TAMANHO = 50
PALETAOFFSET = 20
# Cores
PRETO      = (0, 0, 0)
BRANCO = (255,255,255)
#VARIAVEIS
# Será utilizado para a velocidade do jogo
fps = 250
```

```
colisao1=0
loop=0

# Função para desenhar a arena
def desenhaArena():
    TELA.fill(BRANCO)
    # Desenha a quadra
    pygame.draw.rect(TELA, PRETO, ((0,0),
(LARGURA_TELA,ALTURA_TELA)), LARGURA_LINHA*2)
    # Desenha a linha no centro
    pygame.draw.line(TELA, PRETO, ((LARGURA_TELA//2),0),
((LARGURA_TELA//2),ALTURA_TELA), (LARGURA_LINHA//4))

# Função para desenhar a paleta
def desenhaPaleta(paleta):
    #Impede da paleta ir  além da borda do fundo
    if paleta.bottom > ALTURA_TELA - LARGURA_LINHA:
        paleta.bottom = ALTURA_TELA - LARGURA_LINHA
    #Impede da paleta ir  além da borda do topo
    elif paleta.top < LARGURA_LINHA:
        paleta.top = LARGURA_LINHA
        #Desenha a paleta
    pygame.draw.rect(TELA, PRETO, paleta)


# Função para desenhar a bola
def desenhaBola(bola):
    pygame.draw.rect(TELA, PRETO, bola)

#altera a direção da bola e retorna ela
def moveBola(bola, bolaDirX, bolaDirY):
        bola.x += bolaDirX
        bola.y += bolaDirY
        return bola

# Verifica por colisão com as bordas
# Retorna uma nova posição caso exista colisão
def verificaColisao(bola, bolaDirX, bolaDirY):
    if bola.top == (LARGURA_LINHA) or bola.bottom ==
(ALTURA_TELA - LARGURA_LINHA):
            bolaDirY = bolaDirY * -1
    if bola.left == (LARGURA_LINHA) or bola.right ==
(LARGURA_TELA - LARGURA_LINHA):
            bolaDirX = bolaDirX * -1
    return bolaDirX, bolaDirY

# Rotina de IA para o NPC.
def inteligenciaArtificial(bola, bolaDirX, paleta2):
# Movimentar a paleta quando a bola vem em direção da paleta
    if bolaDirX == 1:
            if paleta2.centery < bola.centery:
                paleta2.y += 1
            else:
                paleta2.y -=1
    return paleta2
```

```
#Verifica a colisão da bola com a paleta1 ou paleta2
def verificaColisaoBola(bola, paleta1, paleta2, bolaDirX):

    if bolaDirX == -1 and paleta1.right == bola.left and
paleta1.top < bola.top and paleta1.bottom > bola.bottom:
        return -1

    elif bolaDirX == 1 and paleta2.left == bola.right and
paleta2.top < bola.top and paleta2.bottom > bola.bottom:
        return -1
    else:
        return 1


#Verifica se o jogador fez ponto e retorna o novo valor do
placar
def verificaPlacar(paleta1, bola, placar, bolaDirX, speedloop):
    #zera a contagem se a bola acerta a borda do jogador
    if bola.left == LARGURA_LINHA:
        return 0
    #1 ponto por acertar a bola
    elif bolaDirX == 1 and paleta1.right == bola.left and
paleta1.top < bola.top and paleta1.bottom > bola.bottom:
        placar += (1+speedloop)
        return placar
    #10 pontos se vender a paleta do computador
    elif bola.right == LARGURA_TELA - LARGURA_LINHA:
        placar += 10
        return placar
    #retorna o mesmo placar se nenhum ponto foi adicionado
    else: return placar

def verificacolisao(paleta1, bola, colisao1, bolaDirX):

    if bolaDirX == 1 and paleta1.right == bola.left and
paleta1.top < bola.top and paleta1.bottom > bola.bottom:
        colisao1 += 1
        return colisao1
    else: return colisao1


def desenhaPlacar(placar):
    resultadoSurf = BASICFONT.render('placar = %s' %(placar),
True, PRETO)
    resultadoRect = resultadoSurf.get_rect()
    resultadoRect.topleft = (LARGURA_TELA - 150, 25)
    TELA.blit(resultadoSurf, resultadoRect)

def desenhaColisoes(colisao):
    resultadoSurf = BASICFONT.render('Colisão = %s' %(colisao),
True, PRETO)
    resultadoRect = resultadoSurf.get_rect()
    resultadoRect.topleft = (LARGURA_TELA - 350, 25)
    TELA.blit(resultadoSurf, resultadoRect)

def desenhavelocidade(fps):
```

```
    resultadoSurf = BASICFONT.render('Velocidade = %s' %(fps),
True, PRETO)
    resultadoRect = resultadoSurf.get_rect()
    resultadoRect.topleft = (LARGURA_TELA - 380, 265)
    TELA.blit(resultadoSurf, resultadoRect)



#funcao Principal
def main():
    pygame.init()

    #Informação da fonte
    global BASICFONT, BASICFONTSIZE
    BASICFONTSIZE = 20
    BASICFONT = pygame.font.Font('freesansbold.ttf',
BASICFONTSIZE)

    global TELA

    FPSCLOCK = pygame.time.Clock()
    TELA = pygame.display.set_mode((LARGURA_TELA,ALTURA_TELA))
    pygame.display.set_caption('PongNet')
    colisao10=False

    #Iniciando as variáveis nas posições iniciais
    #Estas variáveis serão alteradas ao longo da execução
    bolaX = LARGURA_TELA//2 - LARGURA_LINHA//2
    bolaY = ALTURA_TELA//2 - LARGURA_LINHA//2
    jogadorUm_posicao = (ALTURA_TELA - PALETA_TAMANHO) //2
    jogadorDois_posicao = (ALTURA_TELA - PALETA_TAMANHO) //2
    placar = 0
    colisao1 = 0
    fps = 250
    speedloop=0


    #altera a posição da bola
    bolaDirX = -1
    bolaDirY = -1

    #Criando os retangulos para a bola e paletas.
    paleta1 =
pygame.Rect(PALETAOFFSET,jogadorUm_posicao,LARGURA_LINHA,PALETA_
TAMANHO)
    paleta2 = pygame.Rect(LARGURA_TELA - PALETAOFFSET -
LARGURA_LINHA, jogadorDois_posicao,
LARGURA_LINHA,PALETA_TAMANHO)
    bola = pygame.Rect(bolaX, bolaY, LARGURA_LINHA,
LARGURA_LINHA)

    #Desenhando as posições iniciais da Arena
    desenhaArena()
    desenhaPaleta(paleta1)
    desenhaPaleta(paleta2)
    desenhaBola(bola)
```

```
    pygame.mouse.set_visible(0)

    while True: #Loop principal
        for event in pygame.event.get():
                if event.type == QUIT:
                   pygame.quit()
                   sys.exit()
                elif event.type == MOUSEMOTION:
                    mouseX, mouseY = event.pos
                    paleta1.y = mouseY

        if colisao1 == 10:
            colisao10=True



        if colisao10 == True:
            fps+=10
            speedloop+=1
            colisao10=False
            colisao1=0

        desenhaArena()
        desenhaPaleta(paleta1)
        desenhaPaleta(paleta2)
        desenhaBola(bola)


        bola = moveBola(bola, bolaDirX, bolaDirY)
        bolaDirX, bolaDirY = verificaColisao(bola, bolaDirX,
bolaDirY)
        bolaDirX = bolaDirX * verificaColisaoBola(bola, paleta1,
paleta2, bolaDirX)
        paleta2 = inteligenciaArtificial(bola, bolaDirX,
paleta2)

        placar = verificaPlacar(paleta1, bola, placar, bolaDirX,
speedloop)
        colisao1 = verificacolisao(paleta1, bola, colisao1,
bolaDirX)
        desenhaPlacar(placar)
        desenhaColisoes(colisao1)
        desenhavelocidade(fps)

        pygame.display.update()
        FPSCLOCK.tick(fps)


if __name__=='__main__':
 main()
```



#19.
Usando a biblioteca Pygame, escreva um programa que implemente o jogo "Pong" alterado na questão anterior e que adicione uma nova modificação. Tal modificação consiste em inserir um som quando a bola bate nas paletas dos jogadores. (código)

```
import pygame, sys
from pygame.locals import *

# CONSTANTES
# Constantes para o tamanho da tela
LARGURA_TELA = 400
ALTURA_TELA = 300
# Valores para o desenho das paletas e do fundo
LARGURA_LINHA = 10
PALETA_TAMANHO = 50
PALETAOFFSET = 20
# Cores
PRETO      = (0, 0, 0)
BRANCO = (255,255,255)
#VARIAVEIS
# Será utilizado para a velocidade do jogo
fps = 250
colisao1=0
loop=0

# Função para desenhar a arena
def desenhaArena():
    TELA.fill(BRANCO)
    # Desenha a quadra
    pygame.draw.rect(TELA, PRETO, ((0,0),
(LARGURA_TELA,ALTURA_TELA)), LARGURA_LINHA*2)
    # Desenha a linha no centro
```

```
    pygame.draw.line(TELA, PRETO, ((LARGURA_TELA//2),0),
((LARGURA_TELA//2),ALTURA_TELA), (LARGURA_LINHA//4))

def configuracaoSom():
    global pongSom
    global pontoSom
    global ponto10Som
    global derrotaSom
    pongSom = pygame.mixer.Sound("pong.wav")
    pontoSom = pygame.mixer.Sound("ponto.wav")
    ponto10Som = pygame.mixer.Sound("ponto10.wav")
    derrotaSom = pygame.mixer.Sound("derrota.wav")


# Função para desenhar a paleta
def desenhaPaleta(paleta):
    #Impede da paleta ir  além da borda do fundo
    if paleta.bottom > ALTURA_TELA - LARGURA_LINHA:
        paleta.bottom = ALTURA_TELA - LARGURA_LINHA
    #Impede da paleta ir  além da borda do topo
    elif paleta.top < LARGURA_LINHA:
        paleta.top = LARGURA_LINHA
        #Desenha a paleta
    pygame.draw.rect(TELA, PRETO, paleta)


# Função para desenhar a bola
def desenhaBola(bola):
    pygame.draw.rect(TELA, PRETO, bola)

#altera a direção da bola e retorna ela
def moveBola(bola, bolaDirX, bolaDirY):
        bola.x += bolaDirX
        bola.y += bolaDirY
        return bola

# Verifica por colisão com as bordas
# Retorna uma nova posição caso exista colisão
def verificaColisao(bola, bolaDirX, bolaDirY):
    if bola.top == (LARGURA_LINHA) or bola.bottom ==
(ALTURA_TELA - LARGURA_LINHA):
            bolaDirY = bolaDirY * -1
    if bola.left == (LARGURA_LINHA) or bola.right ==
(LARGURA_TELA - LARGURA_LINHA):
            bolaDirX = bolaDirX * -1
    return bolaDirX, bolaDirY

# Rotina de IA para o NPC.
def inteligenciaArtificial(bola, bolaDirX, paleta2):
# Movimentar a paleta quando a bola vem em direção da paleta
    if bolaDirX == 1:
            if paleta2.centery < bola.centery:
                paleta2.y += 1
            else:
                paleta2.y -=1
    return paleta2
```

```
#Verifica a colisão da bola com a paleta1 ou paleta2
def verificaColisaoBola(bola, paleta1, paleta2, bolaDirX):

    if bolaDirX == -1 and paleta1.right == bola.left and
paleta1.top < bola.top and paleta1.bottom > bola.bottom:
        pongSom.play()
        return -1


    elif bolaDirX == 1 and paleta2.left == bola.right and
paleta2.top < bola.top and paleta2.bottom > bola.bottom:
        pongSom.play()
        return -1
    else:
        return 1


#Verifica se o jogador fez ponto e retorna o novo valor do
placar
def verificaPlacar(paleta1, bola, placar, bolaDirX, speedloop):
    #zera a contagem se a bola acerta a borda do jogador
    if bola.left == LARGURA_LINHA:
        return 0
    #1 ponto por acertar a bola
    elif bolaDirX == 1 and paleta1.right == bola.left and
paleta1.top < bola.top and paleta1.bottom > bola.bottom:
        placar += (1+speedloop)
        return placar
    #10 pontos se vender a paleta do computador
    elif bola.right == LARGURA_TELA - LARGURA_LINHA:
        placar += 10
        return placar
    #retorna o mesmo placar se nenhum ponto foi adicionado
    else: return placar

def verificacolisao(paleta1, bola, colisao1, bolaDirX):

    if bolaDirX == 1 and paleta1.right == bola.left and
paleta1.top < bola.top and paleta1.bottom > bola.bottom:
        colisao1 += 1
        return colisao1
    else: return colisao1



def desenhaPlacar(placar):
    resultadoSurf = BASICFONT.render('placar = %s' %(placar),
True, PRETO)
    resultadoRect = resultadoSurf.get_rect()
    resultadoRect.topleft = (LARGURA_TELA - 150, 25)
    TELA.blit(resultadoSurf, resultadoRect)

def desenhaColisoes(colisao):
    resultadoSurf = BASICFONT.render('Colisão = %s' %(colisao),
True, PRETO)
```

```
    resultadoRect = resultadoSurf.get_rect()
    resultadoRect.topleft = (LARGURA_TELA - 350, 25)
    TELA.blit(resultadoSurf, resultadoRect)

def desenhavelocidade(fps):
    resultadoSurf = BASICFONT.render('Velocidade = %s' %(fps),
True, PRETO)
    resultadoRect = resultadoSurf.get_rect()
    resultadoRect.topleft = (LARGURA_TELA - 380, 265)
    TELA.blit(resultadoSurf, resultadoRect)



#funcao Principal
def main():
    pygame.init()

    #Informação da fonte
    global BASICFONT, BASICFONTSIZE
    BASICFONTSIZE = 20
    BASICFONT = pygame.font.Font('freesansbold.ttf',
BASICFONTSIZE)

    global TELA

    FPSCLOCK = pygame.time.Clock()
    TELA = pygame.display.set_mode((LARGURA_TELA,ALTURA_TELA))
    pygame.display.set_caption('PongNet')
    colisao10=False

    #Iniciando as variáveis nas posições iniciais
    #Estas variáveis serão alteradas ao longo da execução
    bolaX = LARGURA_TELA//2 - LARGURA_LINHA//2
    bolaY = ALTURA_TELA//2 - LARGURA_LINHA//2
    jogadorUm_posicao = (ALTURA_TELA - PALETA_TAMANHO) //2
    jogadorDois_posicao = (ALTURA_TELA - PALETA_TAMANHO) //2
    placar = 0
    colisao1 = 0
    fps = 250
    speedloop=0


    #altera a posição da bola
    bolaDirX = -1
    bolaDirY = -1

    #Criando os retangulos para a bola e paletas.
    paleta1 =
pygame.Rect(PALETAOFFSET,jogadorUm_posicao,LARGURA_LINHA,PALETA_
TAMANHO)
    paleta2 = pygame.Rect(LARGURA_TELA - PALETAOFFSET -
LARGURA_LINHA, jogadorDois_posicao,
LARGURA_LINHA,PALETA_TAMANHO)
    bola = pygame.Rect(bolaX, bolaY, LARGURA_LINHA,
LARGURA_LINHA)
```

```
    #Desenhando as posições iniciais da Arena
    desenhaArena()
    desenhaPaleta(paleta1)
    desenhaPaleta(paleta2)
    desenhaBola(bola)

    pygame.mouse.set_visible(0)

    while True: #Loop principal
        for event in pygame.event.get():
                if event.type == QUIT:
                   pygame.quit()
                   sys.exit()
                elif event.type == MOUSEMOTION:
                    mouseX, mouseY = event.pos
                    paleta1.y = mouseY

        if colisao1 == 10:
            colisao10=True



        if colisao10 == True:
            fps+=10
            speedloop+=1
            colisao10=False
            colisao1=0

        desenhaArena()
        desenhaPaleta(paleta1)
        desenhaPaleta(paleta2)
        desenhaBola(bola)
        configuracaoSom()


        bola = moveBola(bola, bolaDirX, bolaDirY)
        bolaDirX, bolaDirY = verificaColisao(bola, bolaDirX,
bolaDirY)
        bolaDirX = bolaDirX * verificaColisaoBola(bola, paleta1,
paleta2, bolaDirX)
        paleta2 = inteligenciaArtificial(bola, bolaDirX,
paleta2)

        placar = verificaPlacar(paleta1, bola, placar, bolaDirX,
speedloop)
        colisao1 = verificacolisao(paleta1, bola, colisao1,
bolaDirX)
        desenhaPlacar(placar)
        desenhaColisoes(colisao1)
        desenhavelocidade(fps)


        pygame.display.update()
        FPSCLOCK.tick(fps)
```

```
if __name__=='__main__':
 main()
```

#20.
Usando a biblioteca Pygame, escreva um programa que implemente o jogo “Pong” alterado na questão anterior e que adicione uma nova modificação. Tal modificação consiste em inserir um som (vitória) quando a bola bate na borda atrás da paleta do computador. (código)

```
import pygame, sys
from pygame.locals import *

# CONSTANTES
# Constantes para o tamanho da tela
LARGURA_TELA = 400
ALTURA_TELA = 300
# Valores para o desenho das paletas e do fundo
LARGURA_LINHA = 10
PALETA_TAMANHO = 50
PALETAOFFSET = 20
# Cores
PRETO      = (0, 0, 0)
BRANCO = (255,255,255)
#VARIAVEIS
# Será utilizado para a velocidade do jogo
fps = 250
colisao1=0
loop=0

# Função para desenhar a arena
def desenhaArena():
    TELA.fill(BRANCO)
    # Desenha a quadra
    pygame.draw.rect(TELA, PRETO, ((0,0),
(LARGURA_TELA,ALTURA_TELA)), LARGURA_LINHA*2)
    # Desenha a linha no centro
    pygame.draw.line(TELA, PRETO, ((LARGURA_TELA//2),0),
((LARGURA_TELA//2),ALTURA_TELA), (LARGURA_LINHA//4))

def configuracaoSom():
    global pongSom
    global pontoSom
    global ponto10Som
    global derrotaSom
    pongSom = pygame.mixer.Sound("pong.wav")
    pontoSom = pygame.mixer.Sound("ponto.wav")
    ponto10Som = pygame.mixer.Sound("ponto10.wav")
    derrotaSom = pygame.mixer.Sound("derrota.wav")


# Função para desenhar a paleta
def desenhaPaleta(paleta):
    #Impede da paleta ir  além da borda do fundo
    if paleta.bottom > ALTURA_TELA - LARGURA_LINHA:
        paleta.bottom = ALTURA_TELA - LARGURA_LINHA
```

```
    #Impede da paleta ir  além da borda do topo
    elif paleta.top < LARGURA_LINHA:
        paleta.top = LARGURA_LINHA
        #Desenha a paleta
    pygame.draw.rect(TELA, PRETO, paleta)


# Função para desenhar a bola
def desenhaBola(bola):
    pygame.draw.rect(TELA, PRETO, bola)

#altera a direção da bola e retorna ela
def moveBola(bola, bolaDirX, bolaDirY):
        bola.x += bolaDirX
        bola.y += bolaDirY
        return bola

# Verifica por colisão com as bordas
# Retorna uma nova posição caso exista colisão
def verificaColisao(bola, bolaDirX, bolaDirY):
    if bola.top == (LARGURA_LINHA) or bola.bottom ==
(ALTURA_TELA - LARGURA_LINHA):
            bolaDirY = bolaDirY * -1
    if bola.left == (LARGURA_LINHA) or bola.right ==
(LARGURA_TELA - LARGURA_LINHA):
            bolaDirX = bolaDirX * -1
    return bolaDirX, bolaDirY

# Rotina de IA para o NPC.
def inteligenciaArtificial(bola, bolaDirX, paleta2):
# Movimentar a paleta quando a bola vem em direção da paleta
    if bolaDirX == 1:
            if paleta2.centery < bola.centery:
                paleta2.y += 1
            else:
                paleta2.y -=1
    return paleta2

#Verifica a colisão da bola com a paleta1 ou paleta2
def verificaColisaoBola(bola, paleta1, paleta2, bolaDirX):

    if bolaDirX == -1 and paleta1.right == bola.left and
paleta1.top < bola.top and paleta1.bottom > bola.bottom:
        pongSom.play()
        return -1


    elif bolaDirX == 1 and paleta2.left == bola.right and
paleta2.top < bola.top and paleta2.bottom > bola.bottom:
        pongSom.play()
        return -1
    else:
        return 1
```

```
#Verifica se o jogador fez ponto e retorna o novo valor do
placar
def verificaPlacar(paleta1, bola, placar, bolaDirX, speedloop):
    #zera a contagem se a bola acerta a borda do jogador
    if bola.left == LARGURA_LINHA:
        return 0
    #1 ponto por acertar a bola
    elif bolaDirX == 1 and paleta1.right == bola.left and
paleta1.top < bola.top and paleta1.bottom > bola.bottom:
        placar += (1+speedloop)
        return placar
    #10 pontos se vender a paleta do computador
    elif bola.right == LARGURA_TELA - LARGURA_LINHA:
        placar += 10
        return placar
    #retorna o mesmo placar se nenhum ponto foi adicionado
    else: return placar

def verificacolisao(paleta1, bola, colisao1, bolaDirX):

    if bolaDirX == 1 and paleta1.right == bola.left and
paleta1.top < bola.top and paleta1.bottom > bola.bottom:
        colisao1 += 1
        return colisao1
    else: return colisao1



def desenhaPlacar(placar):
    resultadoSurf = BASICFONT.render('placar = %s' %(placar),
True, PRETO)
    resultadoRect = resultadoSurf.get_rect()
    resultadoRect.topleft = (LARGURA_TELA - 150, 25)
    TELA.blit(resultadoSurf, resultadoRect)

def desenhaColisoes(colisao):
    resultadoSurf = BASICFONT.render('Colisão = %s' %(colisao),
True, PRETO)
    resultadoRect = resultadoSurf.get_rect()
    resultadoRect.topleft = (LARGURA_TELA - 350, 25)
    TELA.blit(resultadoSurf, resultadoRect)

def desenhavelocidade(fps):
    resultadoSurf = BASICFONT.render('Velocidade = %s' %(fps),
True, PRETO)
    resultadoRect = resultadoSurf.get_rect()
    resultadoRect.topleft = (LARGURA_TELA - 380, 265)
    TELA.blit(resultadoSurf, resultadoRect)



#funcao Principal
def main():
    pygame.init()

    #Informação da fonte
```

```
    global BASICFONT, BASICFONTSIZE
    BASICFONTSIZE = 20
    BASICFONT = pygame.font.Font('freesansbold.ttf',
BASICFONTSIZE)

    global TELA

    FPSCLOCK = pygame.time.Clock()
    TELA = pygame.display.set_mode((LARGURA_TELA,ALTURA_TELA))
    pygame.display.set_caption('PongNet')
    colisao10=False

    #Iniciando as variáveis nas posições iniciais
    #Estas variáveis serão alteradas ao longo da execução
    bolaX = LARGURA_TELA//2 - LARGURA_LINHA//2
    bolaY = ALTURA_TELA//2 - LARGURA_LINHA//2
    jogadorUm_posicao = (ALTURA_TELA - PALETA_TAMANHO) //2
    jogadorDois_posicao = (ALTURA_TELA - PALETA_TAMANHO) //2
    placar = 0
    colisao1 = 0
    fps = 250
    speedloop=0


    #altera a posição da bola
    bolaDirX = -1
    bolaDirY = -1

    #Criando os retangulos para a bola e paletas.
    paleta1 =
pygame.Rect(PALETAOFFSET,jogadorUm_posicao,LARGURA_LINHA,PALETA_
TAMANHO)
    paleta2 = pygame.Rect(LARGURA_TELA - PALETAOFFSET -
LARGURA_LINHA, jogadorDois_posicao,
LARGURA_LINHA,PALETA_TAMANHO)
    bola = pygame.Rect(bolaX, bolaY, LARGURA_LINHA,
LARGURA_LINHA)

    #Desenhando as posições iniciais da Arena
    desenhaArena()
    desenhaPaleta(paleta1)
    desenhaPaleta(paleta2)
    desenhaBola(bola)

    pygame.mouse.set_visible(0)

    while True: #Loop principal
        for event in pygame.event.get():
                if event.type == QUIT:
                   pygame.quit()
                   sys.exit()
                elif event.type == MOUSEMOTION:
                    mouseX, mouseY = event.pos
                    paleta1.y = mouseY

        if colisao1 == 10:
```

```
            colisao10=True



        if colisao10 == True:
            fps+=10
            speedloop+=1
            colisao10=False
            colisao1=0

        desenhaArena()
        desenhaPaleta(paleta1)
        desenhaPaleta(paleta2)
        desenhaBola(bola)
        configuracaoSom()


        bola = moveBola(bola, bolaDirX, bolaDirY)
        bolaDirX, bolaDirY = verificaColisao(bola, bolaDirX,
bolaDirY)
        bolaDirX = bolaDirX * verificaColisaoBola(bola, paleta1,
paleta2, bolaDirX)
        paleta2 = inteligenciaArtificial(bola, bolaDirX,
paleta2)

        placar = verificaPlacar(paleta1, bola, placar, bolaDirX,
speedloop)
        colisao1 = verificacolisao(paleta1, bola, colisao1,
bolaDirX)
        desenhaPlacar(placar)
        desenhaColisoes(colisao1)
        desenhavelocidade(fps)


        pygame.display.update()
        FPSCLOCK.tick(fps)


if __name__=='__main__':
 main()
```

#21.
Usando a biblioteca Pygame, escreva um programa que implemente o jogo “Pong” alterado na questão anterior e que adicione uma nova modificação. Tal modificação consiste em inserir um som (derrota) quando a bola bate na borda atrás da paleta do jogador1. (código)

```
import pygame, sys
from pygame.locals import *

# CONSTANTES
# Constantes para o tamanho da tela
LARGURA_TELA = 400
ALTURA_TELA = 300
# Valores para o desenho das paletas e do fundo
LARGURA_LINHA = 10
PALETA_TAMANHO = 50
PALETAOFFSET = 20
# Cores
PRETO      = (0, 0, 0)
BRANCO = (255,255,255)
#VARIAVEIS
# Será utilizado para a velocidade do jogo
fps = 250
colisao1=0
loop=0

# Função para desenhar a arena
def desenhaArena():
    TELA.fill(BRANCO)
    # Desenha a quadra
    pygame.draw.rect(TELA, PRETO, ((0,0),
(LARGURA_TELA,ALTURA_TELA)), LARGURA_LINHA*2)
    # Desenha a linha no centro
    pygame.draw.line(TELA, PRETO, ((LARGURA_TELA//2),0),
((LARGURA_TELA//2),ALTURA_TELA), (LARGURA_LINHA//4))

def configuracaoSom():
    global pongSom
    global pontoSom
    global ponto10Som
    global derrotaSom
    pongSom = pygame.mixer.Sound("pong.wav")
    pontoSom = pygame.mixer.Sound("ponto.wav")
    ponto10Som = pygame.mixer.Sound("ponto10.wav")
    derrotaSom = pygame.mixer.Sound("derrota.wav")


# Função para desenhar a paleta
def desenhaPaleta(paleta):
    #Impede da paleta ir  além da borda do fundo
    if paleta.bottom > ALTURA_TELA - LARGURA_LINHA:
        paleta.bottom = ALTURA_TELA - LARGURA_LINHA
    #Impede da paleta ir  além da borda do topo
    elif paleta.top < LARGURA_LINHA:
        paleta.top = LARGURA_LINHA
```

```
        #Desenha a paleta
    pygame.draw.rect(TELA, PRETO, paleta)


# Função para desenhar a bola
def desenhaBola(bola):
    pygame.draw.rect(TELA, PRETO, bola)

#altera a direção da bola e retorna ela
def moveBola(bola, bolaDirX, bolaDirY):
        bola.x += bolaDirX
        bola.y += bolaDirY
        return bola

# Verifica por colisão com as bordas
# Retorna uma nova posição caso exista colisão
def verificaColisao(bola, bolaDirX, bolaDirY):
    if bola.top == (LARGURA_LINHA) or bola.bottom ==
(ALTURA_TELA - LARGURA_LINHA):
            bolaDirY = bolaDirY * -1

    if bola.left == (LARGURA_LINHA) or bola.right ==
(LARGURA_TELA - LARGURA_LINHA):
            bolaDirX = bolaDirX * -1
            derrotaSom.play()
    return bolaDirX, bolaDirY

# Rotina de IA para o NPC.
def inteligenciaArtificial(bola, bolaDirX, paleta2):
# Movimentar a paleta quando a bola vem em direção da paleta
    if bolaDirX == 1:
            if paleta2.centery < bola.centery:
                paleta2.y += 1
            else:
                paleta2.y -=1
    return paleta2

#Verifica a colisão da bola com a paleta1 ou paleta2
def verificaColisaoBola(bola, paleta1, paleta2, bolaDirX):

    if bolaDirX == -1 and paleta1.right == bola.left and
paleta1.top < bola.top and paleta1.bottom > bola.bottom:
        pongSom.play()
        return -1


    elif bolaDirX == 1 and paleta2.left == bola.right and
paleta2.top < bola.top and paleta2.bottom > bola.bottom:
        pongSom.play()
        return -1
    else:
        return 1


#Verifica se o jogador fez ponto e retorna o novo valor do
placar
```

```
def verificaPlacar(paleta1, bola, placar, bolaDirX, speedloop):
    #zera a contagem se a bola acerta a borda do jogador
    if bola.left == LARGURA_LINHA:
        return 0
    #1 ponto por acertar a bola
    elif bolaDirX == 1 and paleta1.right == bola.left and
paleta1.top < bola.top and paleta1.bottom > bola.bottom:
        placar += (1+speedloop)

        return placar
    #10 pontos se vender a paleta do computador
    elif bola.right == LARGURA_TELA - LARGURA_LINHA:
        placar += 10
        pontoSom.play()
        return placar
    #retorna o mesmo placar se nenhum ponto foi adicionado
    else: return placar

def verificacolisao(paleta1, bola, colisao1, bolaDirX):

    if bolaDirX == 1 and paleta1.right == bola.left and
paleta1.top < bola.top and paleta1.bottom > bola.bottom:
        colisao1 += 1
        return colisao1
    else: return colisao1



def desenhaPlacar(placar):
    resultadoSurf = BASICFONT.render('placar = %s' %(placar),
True, PRETO)
    resultadoRect = resultadoSurf.get_rect()
    resultadoRect.topleft = (LARGURA_TELA - 150, 25)
    TELA.blit(resultadoSurf, resultadoRect)

def desenhaColisoes(colisao):
    resultadoSurf = BASICFONT.render('Colisão = %s' %(colisao),
True, PRETO)
    resultadoRect = resultadoSurf.get_rect()
    resultadoRect.topleft = (LARGURA_TELA - 350, 25)
    TELA.blit(resultadoSurf, resultadoRect)

def desenhavelocidade(fps):
    resultadoSurf = BASICFONT.render('Velocidade = %s' %(fps),
True, PRETO)
    resultadoRect = resultadoSurf.get_rect()
    resultadoRect.topleft = (LARGURA_TELA - 380, 265)
    TELA.blit(resultadoSurf, resultadoRect)



#funcao Principal
def main():
    pygame.init()

    #Informação da fonte
```

```
    global BASICFONT, BASICFONTSIZE
    BASICFONTSIZE = 20
    BASICFONT = pygame.font.Font('freesansbold.ttf',
BASICFONTSIZE)

    global TELA

    FPSCLOCK = pygame.time.Clock()
    TELA = pygame.display.set_mode((LARGURA_TELA,ALTURA_TELA))
    pygame.display.set_caption('PongNet')
    colisao10=False

    #Iniciando as variáveis nas posições iniciais
    #Estas variáveis serão alteradas ao longo da execução
    bolaX = LARGURA_TELA//2 - LARGURA_LINHA//2
    bolaY = ALTURA_TELA//2 - LARGURA_LINHA//2
    jogadorUm_posicao = (ALTURA_TELA - PALETA_TAMANHO) //2
    jogadorDois_posicao = (ALTURA_TELA - PALETA_TAMANHO) //2
    placar = 0
    colisao1 = 0
    fps = 250
    speedloop=0


    #altera a posição da bola
    bolaDirX = -1
    bolaDirY = -1

    #Criando os retangulos para a bola e paletas.
    paleta1 =
pygame.Rect(PALETAOFFSET,jogadorUm_posicao,LARGURA_LINHA,PALETA_
TAMANHO)
    paleta2 = pygame.Rect(LARGURA_TELA - PALETAOFFSET -
LARGURA_LINHA, jogadorDois_posicao,
LARGURA_LINHA,PALETA_TAMANHO)
    bola = pygame.Rect(bolaX, bolaY, LARGURA_LINHA,
LARGURA_LINHA)

    #Desenhando as posições iniciais da Arena
    desenhaArena()
    desenhaPaleta(paleta1)
    desenhaPaleta(paleta2)
    desenhaBola(bola)

    pygame.mouse.set_visible(0)

    while True: #Loop principal
        for event in pygame.event.get():
                if event.type == QUIT:
                   pygame.quit()
                   sys.exit()
                elif event.type == MOUSEMOTION:
                    mouseX, mouseY = event.pos
                    paleta1.y = mouseY

        if colisao1 == 10:
```

```
            colisao10=True


        if colisao10 == True:
            fps+=10
            speedloop+=1
            colisao10=False
            colisao1=0

        desenhaArena()
        desenhaPaleta(paleta1)
        desenhaPaleta(paleta2)
        desenhaBola(bola)
        configuracaoSom()


        bola = moveBola(bola, bolaDirX, bolaDirY)
        bolaDirX, bolaDirY = verificaColisao(bola, bolaDirX,
bolaDirY)
        bolaDirX = bolaDirX * verificaColisaoBola(bola, paleta1,
paleta2, bolaDirX)
        paleta2 = inteligenciaArtificial(bola, bolaDirX,
paleta2)

        placar = verificaPlacar(paleta1, bola, placar, bolaDirX,
speedloop)
        colisao1 = verificacolisao(paleta1, bola, colisao1,
bolaDirX)
        desenhaPlacar(placar)
        desenhaColisoes(colisao1)
        desenhavelocidade(fps)


        pygame.display.update()
        FPSCLOCK.tick(fps)


if __name__=='__main__':
 main()
```

#22.
Usando a biblioteca Pygame, escreva um programa que implemente o jogo “Pong” alterado na questão anterior e que adicione uma nova modificação. Tal modificação consiste em inserir um som a cada 10 pontos feitos pelo jogador. (código)

```
import pygame, sys
from pygame.locals import *

# CONSTANTES
# Constantes para o tamanho da tela
LARGURA_TELA = 400
ALTURA_TELA = 300
# Valores para o desenho das paletas e do fundo
LARGURA_LINHA = 10
PALETA_TAMANHO = 50
PALETAOFFSET = 20
# Cores
PRETO      = (0, 0, 0)
BRANCO = (255,255,255)
#VARIAVEIS
# Será utilizado para a velocidade do jogo
fps = 250
colisao1=0
loop=0

# Função para desenhar a arena
def desenhaArena():
    TELA.fill(BRANCO)
    # Desenha a quadra
    pygame.draw.rect(TELA, PRETO, ((0,0),
(LARGURA_TELA,ALTURA_TELA)), LARGURA_LINHA*2)
    # Desenha a linha no centro
    pygame.draw.line(TELA, PRETO, ((LARGURA_TELA//2),0),
((LARGURA_TELA//2),ALTURA_TELA), (LARGURA_LINHA//4))

def configuracaoSom():
    global pongSom
    global pontoSom
    global ponto10Som
    global derrotaSom
    pongSom = pygame.mixer.Sound("pong.wav")
    pontoSom = pygame.mixer.Sound("ponto.wav")
    ponto10Som = pygame.mixer.Sound("ponto10.wav")
    derrotaSom = pygame.mixer.Sound("derrota.wav")


# Função para desenhar a paleta
def desenhaPaleta(paleta):
    #Impede da paleta ir  além da borda do fundo
    if paleta.bottom > ALTURA_TELA - LARGURA_LINHA:
        paleta.bottom = ALTURA_TELA - LARGURA_LINHA
    #Impede da paleta ir  além da borda do topo
    elif paleta.top < LARGURA_LINHA:
        paleta.top = LARGURA_LINHA
        #Desenha a paleta
```

```
    pygame.draw.rect(TELA, PRETO, paleta)

# Função para desenhar a bola
def desenhaBola(bola):
    pygame.draw.rect(TELA, PRETO, bola)

#altera a direção da bola e retorna ela
def moveBola(bola, bolaDirX, bolaDirY):
        bola.x += bolaDirX
        bola.y += bolaDirY
        return bola

# Verifica por colisão com as bordas
# Retorna uma nova posição caso exista colisão
def verificaColisao(bola, bolaDirX, bolaDirY):
    if bola.top == (LARGURA_LINHA) or bola.bottom ==
(ALTURA_TELA - LARGURA_LINHA):
            bolaDirY = bolaDirY * -1

    if bola.left == (LARGURA_LINHA) or bola.right ==
(LARGURA_TELA - LARGURA_LINHA):
            bolaDirX = bolaDirX * -1
            derrotaSom.play()
    return bolaDirX, bolaDirY

# Rotina de IA para o NPC.
def inteligenciaArtificial(bola, bolaDirX, paleta2):
# Movimentar a paleta quando a bola vem em direção da paleta
    if bolaDirX == 1:
            if paleta2.centery < bola.centery:
                paleta2.y += 1
            else:
                paleta2.y -=1
    return paleta2

#Verifica a colisão da bola com a paleta1 ou paleta2
def verificaColisaoBola(bola, paleta1, paleta2, bolaDirX):

    if bolaDirX == -1 and paleta1.right == bola.left and
paleta1.top < bola.top and paleta1.bottom > bola.bottom:
        pongSom.play()
        return -1


    elif bolaDirX == 1 and paleta2.left == bola.right and
paleta2.top < bola.top and paleta2.bottom > bola.bottom:
        pongSom.play()
        return -1
    else:
        return 1


#Verifica se o jogador fez ponto e retorna o novo valor do
placar
def verificaPlacar(paleta1, bola, placar, bolaDirX, speedloop):
    #zera a contagem se a bola acerta a borda do jogador
```

```
    if bola.left == LARGURA_LINHA:
        return 0
    #1 ponto por acertar a bola
    elif bolaDirX == 1 and paleta1.right == bola.left and
paleta1.top < bola.top and paleta1.bottom > bola.bottom:
        placar += (1+speedloop)

        return placar
    #10 pontos se vender a paleta do computador
    elif bola.right == LARGURA_TELA - LARGURA_LINHA:
        placar += 10
        pontoSom.play()
        return placar
    #retorna o mesmo placar se nenhum ponto foi adicionado
    else: return placar

def verificacolisao(paleta1, bola, colisao1, bolaDirX):

    if bolaDirX == 1 and paleta1.right == bola.left and
paleta1.top < bola.top and paleta1.bottom > bola.bottom:
        colisao1 += 1
        return colisao1
    else: return colisao1

def desenhaPlacar(placar):
    resultadoSurf = BASICFONT.render('placar = %s' %(placar),
True, PRETO)
    resultadoRect = resultadoSurf.get_rect()
    resultadoRect.topleft = (LARGURA_TELA - 150, 25)
    TELA.blit(resultadoSurf, resultadoRect)

def desenhaColisoes(colisao):
    resultadoSurf = BASICFONT.render('Colisão = %s' %(colisao),
True, PRETO)
    resultadoRect = resultadoSurf.get_rect()
    resultadoRect.topleft = (LARGURA_TELA - 350, 25)
    TELA.blit(resultadoSurf, resultadoRect)

def desenhavelocidade(fps):
    resultadoSurf = BASICFONT.render('Velocidade = %s' %(fps),
True, PRETO)
    resultadoRect = resultadoSurf.get_rect()
    resultadoRect.topleft = (LARGURA_TELA - 380, 265)
    TELA.blit(resultadoSurf, resultadoRect)

def desenhaplacarreal(placarreal):
    resultadoSurf = BASICFONT.render('Placar real = %s'
%(placarreal), True, PRETO)
    resultadoRect = resultadoSurf.get_rect()
    resultadoRect.topleft = (LARGURA_TELA - 180, 265)
    TELA.blit(resultadoSurf, resultadoRect)



#funcao Principal
def main():
```

```
    pygame.init()

    #Informação da fonte
    global BASICFONT, BASICFONTSIZE
    BASICFONTSIZE = 20
    BASICFONT = pygame.font.Font('freesansbold.ttf',
BASICFONTSIZE)

    global TELA

    FPSCLOCK = pygame.time.Clock()
    TELA = pygame.display.set_mode((LARGURA_TELA,ALTURA_TELA))
    pygame.display.set_caption('PongNet')
    colisao10=False
    ponto10=False

    #Iniciando as variáveis nas posições iniciais
    #Estas variáveis serão alteradas ao longo da execução
    bolaX = LARGURA_TELA//2 - LARGURA_LINHA//2
    bolaY = ALTURA_TELA//2 - LARGURA_LINHA//2
    jogadorUm_posicao = (ALTURA_TELA - PALETA_TAMANHO) //2
    jogadorDois_posicao = (ALTURA_TELA - PALETA_TAMANHO) //2
    placar = 0
    colisao1 = 0
    fps = 250
    speedloop=0
    placarreal=0

    #altera a posição da bola
    bolaDirX = -1
    bolaDirY = -1

    #Criando os retangulos para a bola e paletas.
    paleta1 =
pygame.Rect(PALETAOFFSET,jogadorUm_posicao,LARGURA_LINHA,PALETA_
TAMANHO)
    paleta2 = pygame.Rect(LARGURA_TELA - PALETAOFFSET -
LARGURA_LINHA, jogadorDois_posicao,
LARGURA_LINHA,PALETA_TAMANHO)
    bola = pygame.Rect(bolaX, bolaY, LARGURA_LINHA,
LARGURA_LINHA)

    #Desenhando as posições iniciais da Arena
    desenhaArena()
    desenhaPaleta(paleta1)
    desenhaPaleta(paleta2)
    desenhaBola(bola)

    pygame.mouse.set_visible(0)

    while True: #Loop principal
        for event in pygame.event.get():
                if event.type == QUIT:
                   pygame.quit()
                   sys.exit()
                elif event.type == MOUSEMOTION:
```

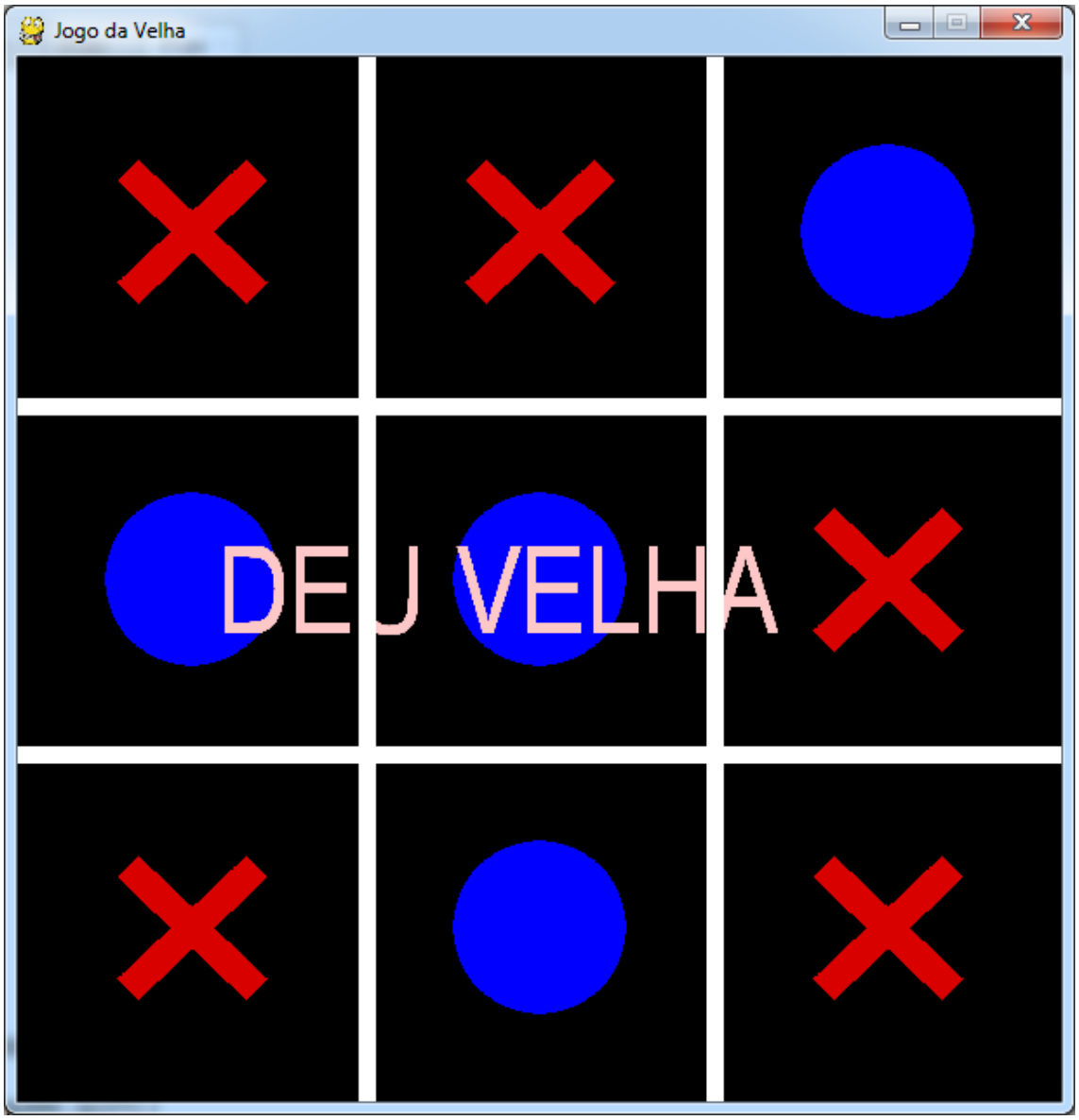
```
                    mouseX, mouseY = event.pos
                    paleta1.y = mouseY

        configuracaoSom()

        if colisao1 == 10:
            colisao10=True

        while placar >= 10:
            ponto10=True
            placarreal+=placar
            placar=0

        if ponto10==True:
            ponto10Som.play()
            ponto10=False



        if colisao10 == True:
            fps+=10
            speedloop+=1
            colisao10=False
            colisao1=0

        desenhaArena()
        desenhaPaleta(paleta1)
        desenhaPaleta(paleta2)
        desenhaBola(bola)
        configuracaoSom()


        bola = moveBola(bola, bolaDirX, bolaDirY)
        bolaDirX, bolaDirY = verificaColisao(bola, bolaDirX,
bolaDirY)
        bolaDirX = bolaDirX * verificaColisaoBola(bola, paleta1,
paleta2, bolaDirX)
        paleta2 = inteligenciaArtificial(bola, bolaDirX,
paleta2)

        placar = verificaPlacar(paleta1, bola, placar, bolaDirX,
speedloop)
        colisao1 = verificacolisao(paleta1, bola, colisao1,
bolaDirX)
        desenhaPlacar(placar)
        desenhaColisoes(colisao1)
        desenhavelocidade(fps)
        desenhaplacarreal(placarreal)


        pygame.display.update()
        FPSCLOCK.tick(fps)


if __name__=='__main__':
 main()
```

#23.
(Desafio) Usando a biblioteca Pygame, escreva um programa que implemente o jogo da velha para dois jogadores (ambos usuários). (código e printscreen

```
import pygame

vermelho = (255,0,0)
verde = (0,255,0)
azul = (0,0,255)
azulescuro = (0,0,128)
branco = (255,255,255)
preto = (0,0,0)
rosa = (255,200,200)

pygame.init()
tela = pygame.display.set_mode((600, 600))
pygame.display.set_caption('Jogo da Velha')
clock = pygame.time.Clock()
terminou = False

estado = 'jogando'
vez = 'jogador1'
escolha= 'X'
espaco=0

marca_tabu = [
    0, 1, 2,
    3, 4, 5,
    6, 7, 8
    ]

rect1 = pygame.Rect ((0, 0), (200, 200))
rect2 = pygame.Rect ((200, 0), (200, 200))
rect3 = pygame.Rect ((400, 0), (200, 200))
rect4 = pygame.Rect ((0, 200), (200, 200))
rect5 = pygame.Rect ((200, 200), (200, 200))
rect6 = pygame.Rect ((400, 200), (200, 200))
rect7 = pygame.Rect ((0, 400), (200, 200))
rect8 = pygame.Rect ((200, 400), (200, 200))
rect9 = pygame.Rect ((400, 400), (200, 200))

rec = [ rect1, rect2, rect3,
        rect4, rect5, rect6,
        rect7, rect8, rect9
        ]

def desenhar_tabu():
    pygame.draw.line(tela, branco, (200, 0), (200, 600), 10)
    pygame.draw.line(tela, branco, (400, 0), (400, 600), 10)
    pygame.draw.line(tela, branco, (0, 200), (600, 200), 10)
    pygame.draw.line(tela, branco, (0, 400), (600, 400), 10)


def desenhar_peca(pos):
    global vez
```

```
    x, y = pos
    if vez == 'jogador2':
        pygame.draw.circle(tela, azul, pos, 50)
    else:
        img = pygame.image.load('x.png').convert_alpha()
        imgR = pygame.transform.scale(img, (100, 100))
        tela.blit(imgR, (x - 50, y - 50))

def testa_pos():
    for p in rec:
        if event.type == pygame.MOUSEBUTTONDOWN and
p.collidepoint(mouse_pos):
            if p == rect1:
                confirmar(0, [100, 100])
            if p == rect2:
                confirmar(1, [300, 100])
            if p == rect3:
                confirmar(2, [500, 100])
            if p == rect4:
                confirmar(3, [100, 300])
            if p == rect5:
                confirmar(4, [300, 300])
            if p == rect6:
                confirmar(5, [500, 300])
            if p == rect7:
                confirmar(6, [100, 500])
            if p == rect8:
                confirmar(7, [300, 500])
            if p == rect9:
                confirmar(8, [500, 500])

def confirmar(indice, pos):
    global escolha, vez, espaco
    if marca_tabu[indice] == 'X':
        print('X')
    elif marca_tabu[indice] == 'O':
        print('O')
    else:
        marca_tabu[indice] = escolha
        desenhar_peca(pos)
        print(marca_tabu)
        if vez == 'jogador1':
            vez = 'jogador2'
        else:
            vez = 'jogador1'
        espaco+=1

def teste_vitoria(l):
    return ((marca_tabu[0] == l and marca_tabu[1] == l and
marca_tabu[2] == l) or
            (marca_tabu[3] == l and marca_tabu[4] == l and
marca_tabu[5] == l) or
            (marca_tabu[6] == l and marca_tabu[7] == l and
marca_tabu[8] == l) or
            (marca_tabu[0] == l and marca_tabu[3] == l and
marca_tabu[6] == l) or
```

```
            (marca_tabu[1] == l and marca_tabu[4] == l and
marca_tabu[7] == l) or
            (marca_tabu[2] == l and marca_tabu[1] == l and
marca_tabu[8] == l) or
            (marca_tabu[0] == l and marca_tabu[4] == l and
marca_tabu[8] == l) or
            (marca_tabu[2] == l and marca_tabu[5] == l and
marca_tabu[6] == l))

def texto_vitoria(v):
    arial= pygame.font.SysFont('arial', 70)
    mensagem = 'Jogador {} Venceu'.format(v)

    if v =='Empate':
        mens_vitoria = arial.render('DEU VELHA', True, rosa, 0)
        tela.blit(mens_vitoria, (115, 265))
    else:
        mens_vitoria = arial.render(mensagem, True, verde, 0)
        tela.blit(mens_vitoria, (0, 265))

while not terminou:
    mouse_pos = pygame.mouse.get_pos()
    if estado == 'jogando':
        desenhar_tabu()

    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            terminou = True
        if event.type == pygame.MOUSEBUTTONDOWN:
            if vez == 'jogador1':
                escolha = 'X'
                testa_pos()
            else:
                escolha = 'O'
                testa_pos()

    if teste_vitoria('X'):
         print('X Venceu')
         texto_vitoria('X')

    elif teste_vitoria('O'):
         print('O Venceu')
         texto_vitoria('O')

    elif espaco >= 9:
         print('Empate')
         texto_vitoria('Empate')


    desenhar_tabu()

    pygame.display.update()

pygame.display.quit()

pygame.quit()
```



#24.
Usando a biblioteca Pygame, escreva um programa que implemente o jogo da memória para um jogador. (código e *printscreen*)

Não consegui fazer a tempo.