ESTI – ESCOLA SUPERIOR DA TECNOLOGIA DA INFORMAÇÃO
GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO
BLOCO DE DESENVOLVIMENTO ANDROID

Materia
Fundamentos de Programação em Python

LUCAS LANGE BARROZO
PROFESSOR:CASSIUS FIGUEIREDO

RIO DE JANEIRO, 27 de novembro de 2018

1)

```
#a

lista = []


#b

for i in range(1,6):
  lista.append(i)

#c
print(lista)

#d

def remove_da_lista(lista, elemento):

  if elemento in lista:
     lista.remove(elemento)
  else:
     print('Elemento não encontrado')

remove_da_lista(lista, 3)
remove_da_lista(lista,6)

#e

print('lista modificada',lista)

#f
tamanho = len(lista)
print('tamanho da lista',tamanho)

#g

lista[tamanho -1 ] = 6
print('lista com ultimo número modificado', lista)
```

#Print

```
[1, 2, 3, 4, 5]
Elemento não encontrado
lista modificada [1, 2, 4, 5]
tamanho da lista 4
lista com ultimo número modificado [1, 2, 4, 6]

Process finished with exit code 0
```

2)

```
#Escreva um programa em Python que leia um vetor de 5 números inteiros e mostre-os.

vetor = [1,2,3,4,5]

print(vetor)
```

3)

```
vetor = ['um', 'dois', 'tres', 'quatro', 'cinco', 'seis', 'sete', 'oito', 'nove',
'dez']

print(vetor[::-1])
```

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)

vetor = [0,0,10,20,133,220]

t = len(vetor)

print('O vetor é:', vetor)
print('Este vetor tem ',t, 'elementos.')

print('O número 0 aparece', vetor.count(0), '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)

input_aluno = str(input('Qual é o nome do aluno?(digite "sair" para terminar) '))

lista_alunos = []
lista_tamanho = []
while input_aluno != 'sair':

   input_tamanho = float(input('Qual é a altura do aluno em metros? '))
   lista_alunos.append(input_aluno)
   lista_tamanho.append(input_tamanho)

   input_aluno = str(input('Qual é o nome do próximo aluno?(digite "sair" para terminar) '))

print('alunos', lista_alunos, '\ntamanhos', lista_tamanho)

def alunos_maior_media (lista_nomes, lista_tamanho):

   media_de_altura = sum(lista_tamanho)/len(lista_tamanho)
   print('A média de altura é:', media_de_altura)

   for tamanho in lista_tamanho:

       index_tamanho = lista_tamanho.index(tamanho)

       if tamanho > media_de_altura:


          print(lista_nomes[index_tamanho], 'tem altura de', tamanho, 'e por isso é maior que a média')

       else:
           print(lista_nomes[index_tamanho], 'tem altura de', tamanho, 'e por isso NÃO é maior que a média')

alunos_maior_media(lista_alunos, lista_tamanho)
```

6)

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

input_usuario = str(input('Digite sua frase (digite "sair" para terminar): ').lower())

lista_frases = []

while input_usuario != 'sair':

   lista_frases.append(input_usuario)
   input_usuario = str(input('Digite sua próxima frase (digite "sair" para terminar): ').lower())

for frases in lista_frases:

   list = frases.split(' ')

   if "eu" in list:
       print('A frase "', frases, '" possui a palavra  "eu"')
```

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)

list = ['a', '2', 'True']


input_usuario = str(input('\nO que deseja?\nDigite "a": Mostrar lista \n"b":
Adicionar item a lista \n"c": remover elemento \n"d": Apagar todos os elementos da
lista \n"sair": sair do programa'))


while input_usuario != 'sair':

   if input_usuario == 'a':
       #a) Mostrar lista;
       print(list)
       input_usuario = str(input('\nO que deseja?\nDigite "a": Mostrar lista \n"b":
Adicionar item a lista \n"c": remover elemento \n"d": Apagar todos os elementos da
lista \n"sair": sair do programa'))

   elif input_usuario == 'b':
       #b) incluir elemento
       elemento = str(input('Qual novo elemento gostaria de acrescentar? '))
       list.append(elemento)
       print('Elemento adicionado com sucesso:', list)
       input_usuario = str(input(
           '\nO que deseja?\nDigite "a": Mostrar lista \n"b": Adicionar item a
lista \n"c": remover elemento \n"d": Apagar todos os elementos da lista \n"sair":
sair do programa'))

   elif input_usuario == 'c':
       #c) Remover elemento
       elemento = str(input('Qual elemento gostaria de remover? '))
       list.remove(elemento)
       print("Elemento removido com sucesso", list)
       input_usuario = str(input(
           '\nO que deseja?\nDigite "a": Mostrar lista \n"b": Adicionar item a
lista \n"c": remover elemento \n"d": Apagar todos os elementos da lista \n"sair":
sair do programa'))

   elif input_usuario == 'd':
       #d) Apagar todos os elementos da lista
       list =[]
       print('Operação concluída')
       print(list)
       input_usuario = str(input(
           '\nO que deseja?\nDigite "a": Mostrar lista \n"b": Adicionar item a
lista \n"c": remover elemento \n"d": Apagar todos os elementos da lista \n"sair":
sair do programa'))

   else:
       input_usuario = str(input(
           '\nO que deseja?\nDigite "a": Mostrar lista \n"b": Adicionar item a
lista \n"c": remover elemento \n"d": Apagar todos os elementos da lista \n"sair":
sair do programa'))
```

8)

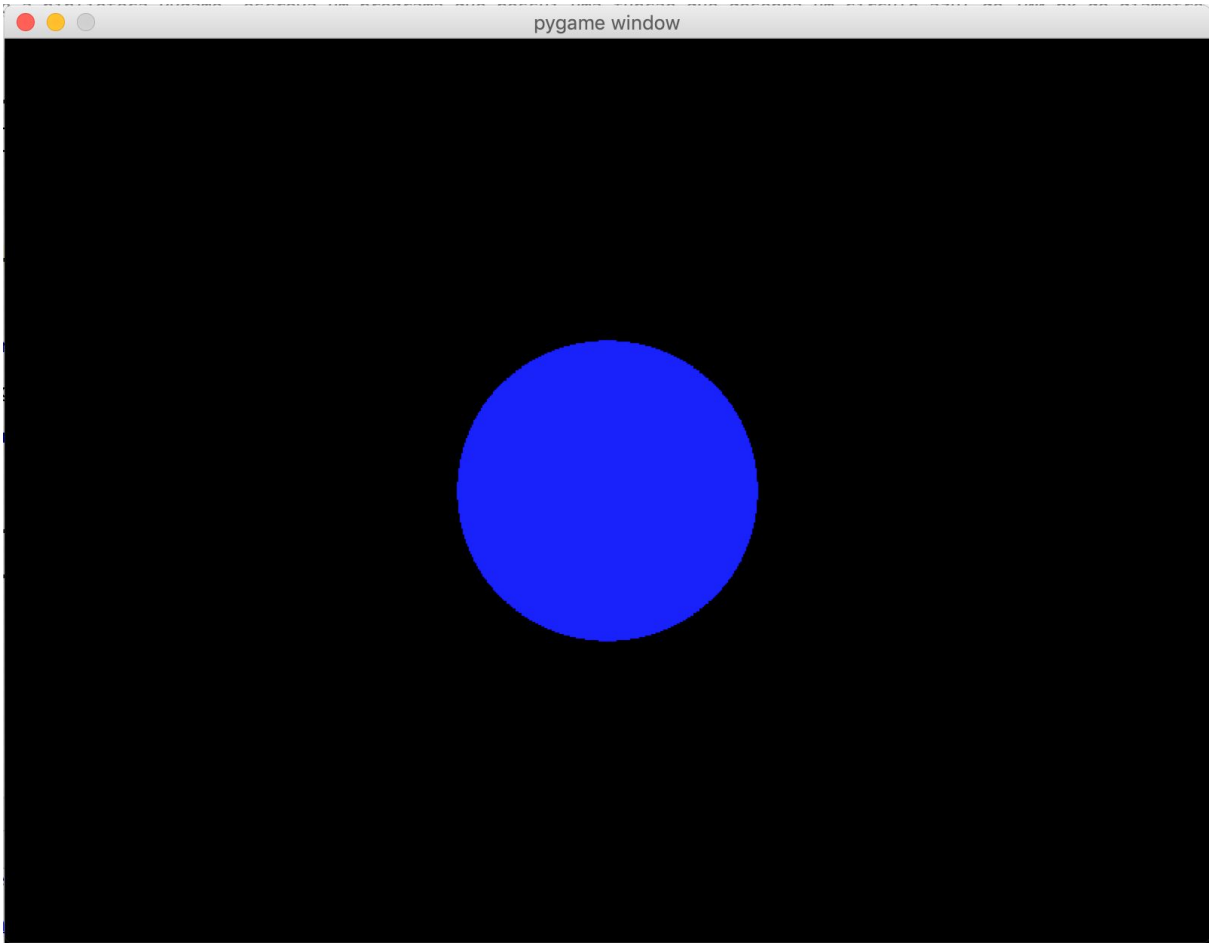
```
#Faça uma funçãoum programa em Python que simula um lançamento de dados. Lance o
dado 100 vezes e armazene os resultados em um vetor. Depois, mostre quantas vezes
cada valor foi conseguido. Dica: use um vetor de contadores (1-6) e uma função do
módulo 'random' de Python para gerar números aleatórios,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

```
import random

def lanca_dado(quantidade_lancamento):
   lista = []

   for lancamento in range(1, quantidade_lancamento +1):
       dado = random.randint(1,6)

       lista.append(dado)

   print('Jogadas individuais', lista)
   print('Tamanho da lista', len(lista))

   quantidade_numeros = {'1': lista.count(1), '2': lista.count(2), '3':
lista.count(3), '4': lista.count(4), '5': lista.count(5), '6': lista.count(6)}

   print(quantidade_numeros)

lanca_dado(100)
```

9)

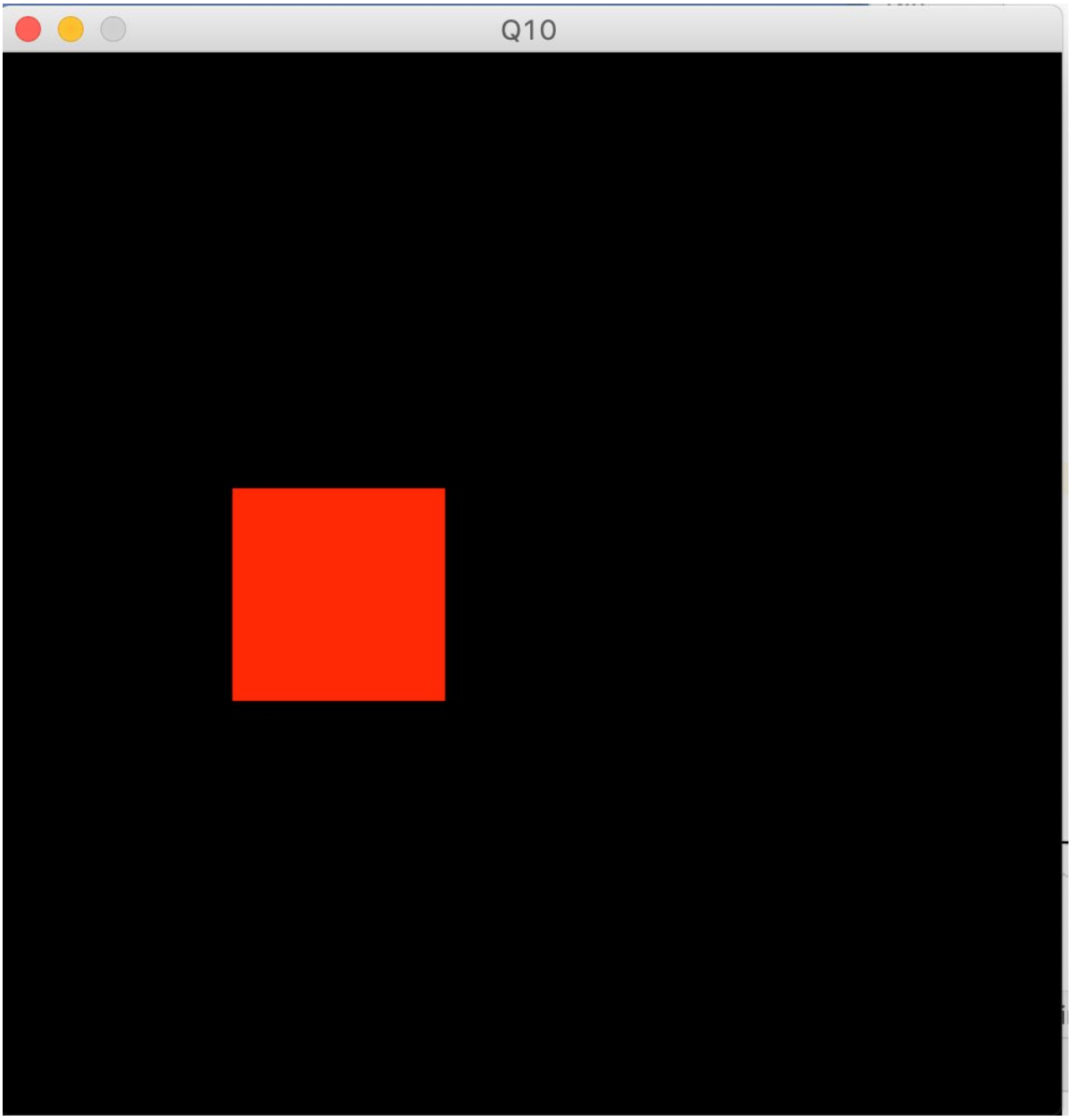
```
#Usando a biblioteca Pygame, escreva um programa que possui uma função que desenha
um círculo azul de 100 px de diâmetro no centro da tela. (código e printscreen)
```

```
import pygame

pygame.init()
screen_width = 800
screen_height = 600
screen = pygame.display.set_mode((screen_width,screen_height))

blue = (0,0,255)

pygame.draw.circle(screen,blue, (int(screen_width/2),int(screen_height/2)), 100)

done = False

while not done:

   pygame.display.update()

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

pygame.display.quit()

pygame.quit()
```


pygame window

10)

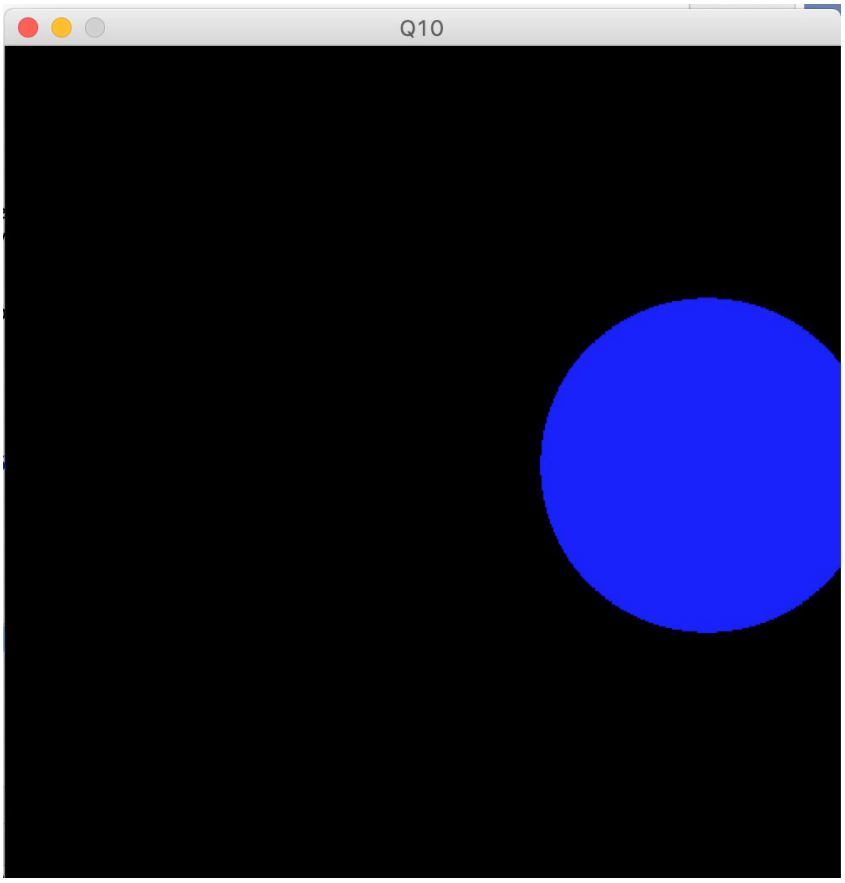
```
import pygame



SCREEN_WIDTH = 500
SCREEN_HEIGHT= 500

width = 100
height= 100

X = SCREEN_WIDTH//2 - width//2
Y = SCREEN_HEIGHT//2 - height//2
RED = (255,0,0)
BLACK = (0,0,0)
pygame.init()



win = pygame.display.set_mode((SCREEN_WIDTH,SCREEN_HEIGHT))

pygame.display.set_caption('Q10')



vel = 5

run = True
while run:

   pygame.time.delay(100)

   for event in pygame.event.get():
       if event.type == pygame.QUIT:
           run = False

   keys = pygame.key.get_pressed()

   if keys[pygame.K_w]:
       Y -= vel
   if keys[pygame.K_s]:
       Y += vel
   if keys[pygame.K_a]:
       X -= vel
   if keys[pygame.K_d]:
       X += vel

   win.fill(BLACK)
   pygame.draw.rect(win,RED,(X, Y,width,height ) )
   pygame.display.update()

pygame.display.quit()

pygame.quit()
```



11)

```
import pygame


SCREEN_WIDTH = 500
SCREEN_HEIGHT= 500


RADIUS= 100

X = SCREEN_WIDTH//2
Y = SCREEN_HEIGHT//2
BLUE = (0,0,255)
BLACK = (0,0,0)
pygame.init()



win = pygame.display.set_mode((SCREEN_WIDTH,SCREEN_HEIGHT))
```

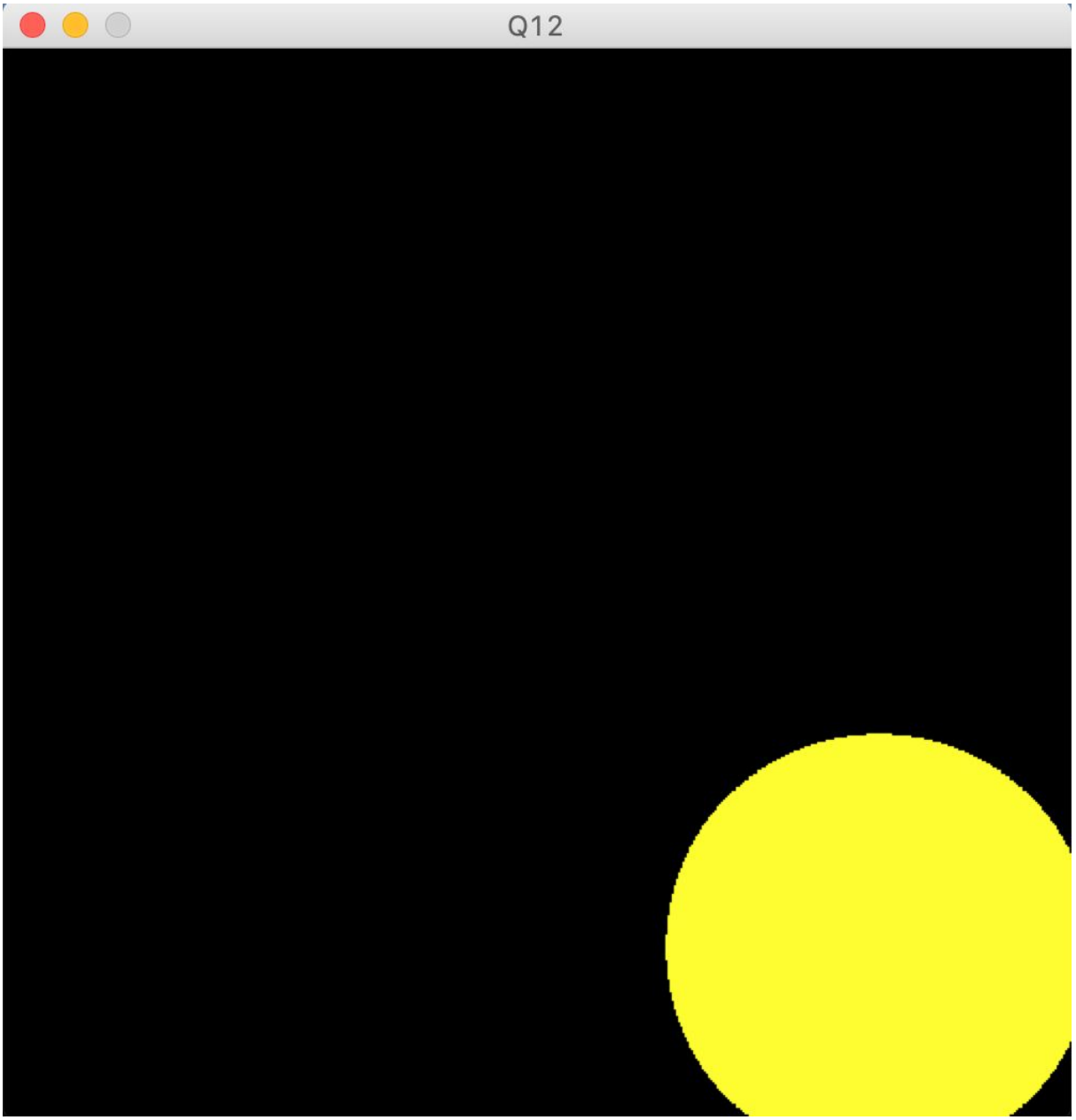
```
pygame.display.set_caption('Q11')


vel = 20

run = True
while run:

   pygame.time.delay(100)

   for event in pygame.event.get():
       if event.type == pygame.QUIT:
           run = False

   X += vel

   #controla colisao

   if X > SCREEN_WIDTH - 50 :
       X = 0
   if X < 0 :
       X = SCREEN_WIDTH - 50

   win.fill(BLACK)
   pygame.draw.circle(win,BLUE,(X, Y ), RADIUS )
   pygame.display.update()

pygame.display.quit()

pygame.quit()
```



12)

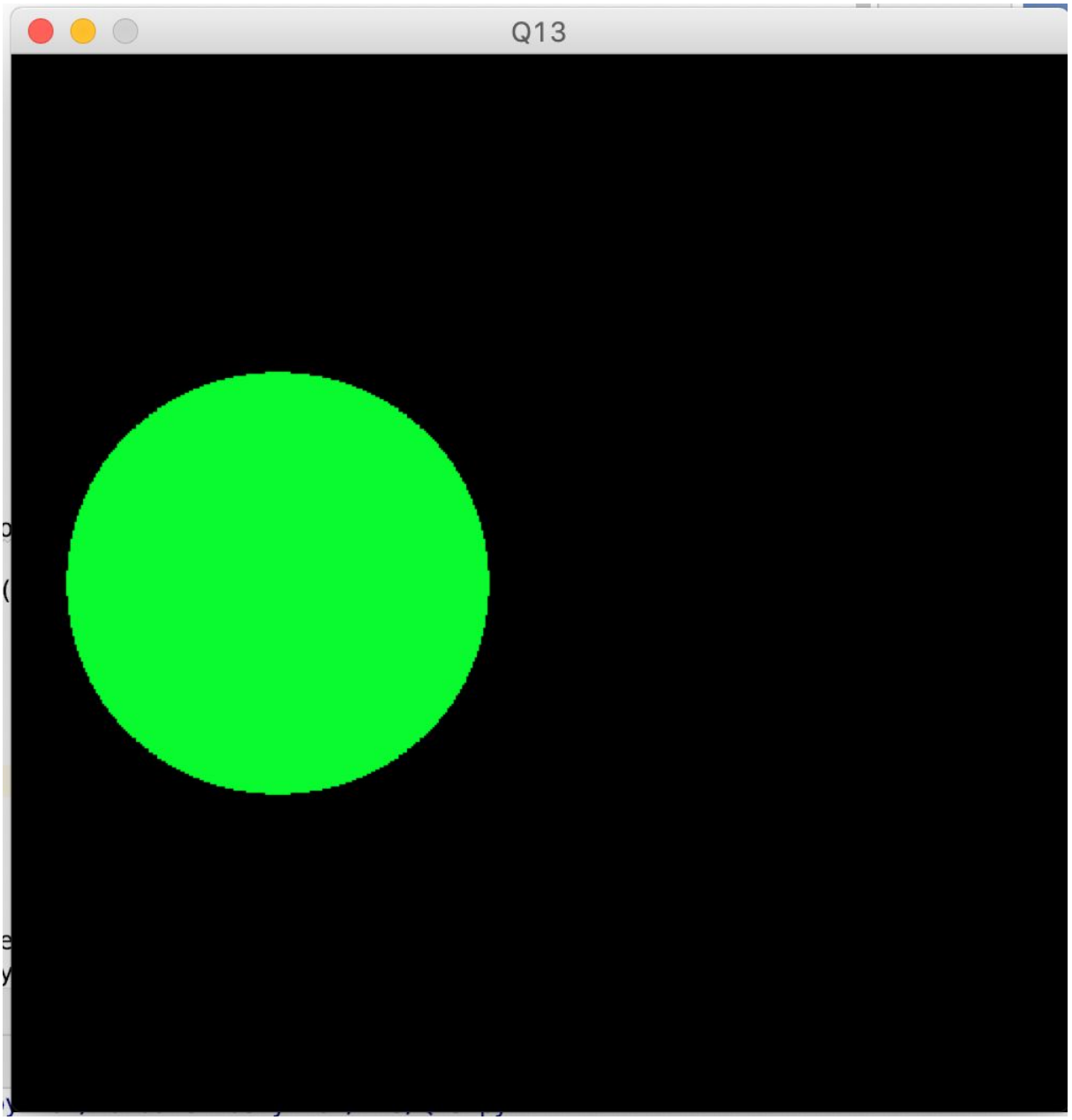
```
import pygame


SCREEN_WIDTH = 500
SCREEN_HEIGHT= 500


RADIUS= 100

X = SCREEN_WIDTH//2
Y = SCREEN_HEIGHT//2
YELLOW = (255,255,0)
BLACK = (0,0,0)
pygame.init()


direction = ''
win = pygame.display.set_mode((SCREEN_WIDTH,SCREEN_HEIGHT))

pygame.display.set_caption('Q12')



vel = 20

run = True
while run:

   pygame.time.delay(100)

   for event in pygame.event.get():
       if event.type == pygame.QUIT:
           run = False


   #input
   keys = pygame.key.get_pressed()

   if keys[pygame.K_w]:
      direction = 'up'

   if keys[pygame.K_s]:
       direction = 'down'

   if keys[pygame.K_a]:
       direction = 'left'

   if keys[pygame.K_d]:
       direction = 'right'

   #faz o circulo se mover constantemente
   if direction == 'up':
       Y -= vel
   if direction == 'down':
       Y += vel
   if direction == 'left':
       X -= vel
   if direction == 'right':
       X += vel



   #controla colisao
```

```
    if X > SCREEN_WIDTH - 50 :
        X = 0

    if X < 0 :
        X = SCREEN_WIDTH - 50

    if Y > SCREEN_HEIGHT - 50:
        Y = 0

    if Y < 0:
        Y = SCREEN_HEIGHT - 50

    win.fill(BLACK)
    pygame.draw.circle(win,YELLOW,(X, Y ), RADIUS )
    pygame.display.update()

pygame.display.quit()

pygame.quit()
```



13)

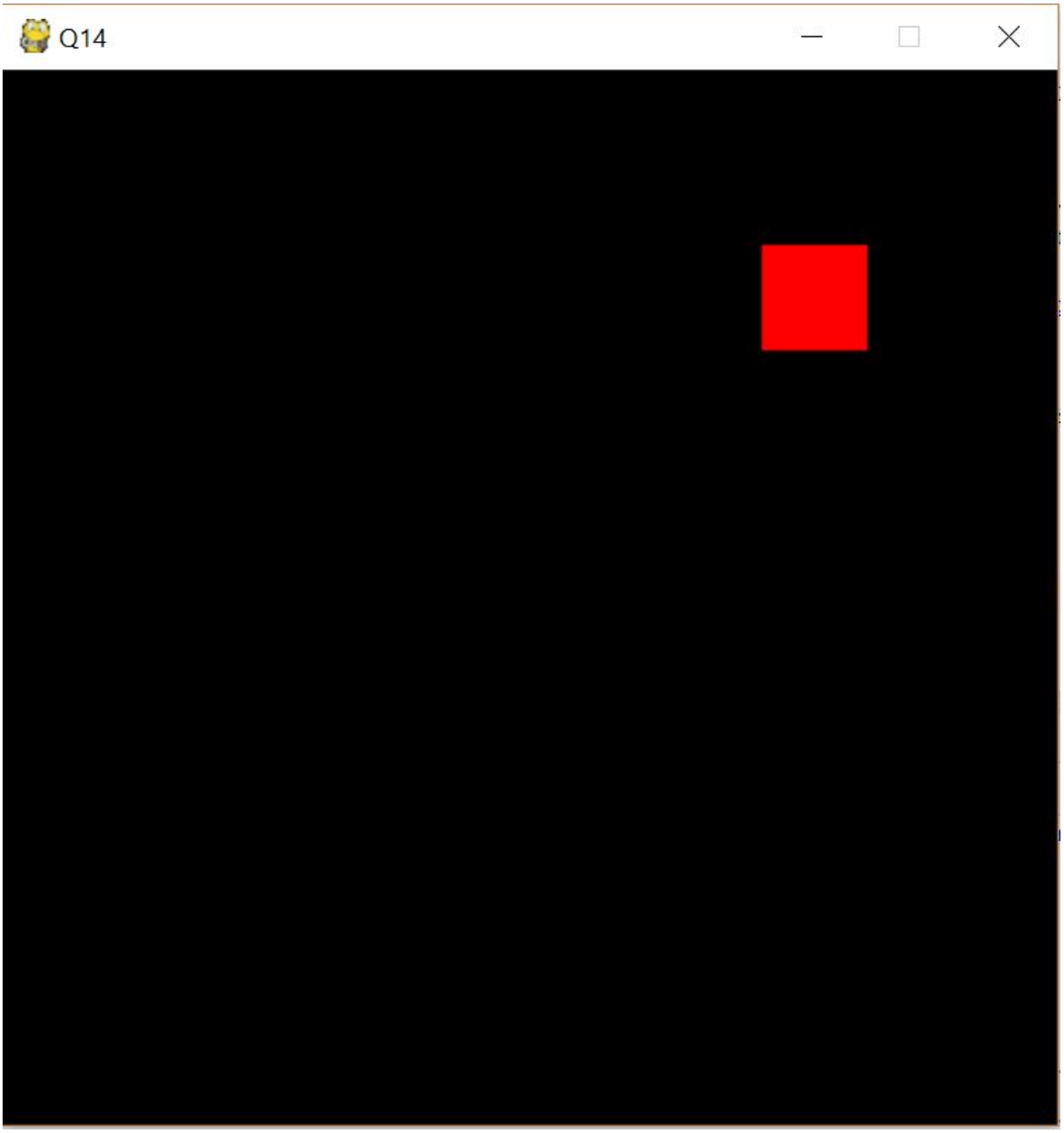
```
import pygame


SCREEN_WIDTH = 500
SCREEN_HEIGHT= 500


RADIUS= 100

X = SCREEN_WIDTH//2
Y = SCREEN_HEIGHT//2
GREEN = (0,255,0)
BLACK = (0,0,0)
pygame.init()



win = pygame.display.set_mode((SCREEN_WIDTH,SCREEN_HEIGHT))

pygame.display.set_caption('Q13')



vel = 20

run = True
while run:

   pygame.time.delay(100)

   for event in pygame.event.get():
       if event.type == pygame.QUIT:
           run = False

   X += vel

   #controla colisao

   if X > SCREEN_WIDTH - 50 :
       vel = (vel + 1) * -1
   if X < 0 :
       vel = (vel - 1) * -1

   win.fill(BLACK)
   pygame.draw.circle(win,GREEN,(X, Y ), RADIUS )
   pygame.display.update()

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



SCREEN_WIDTH = 500
SCREEN_HEIGHT= 500

width = 50
height= 50

X = SCREEN_WIDTH//2 - width//2
Y = SCREEN_HEIGHT//2 - height//2
```

```
RED = (255,0,0)
BLACK = (0,0,0)
pygame.init()



win = pygame.display.set_mode((SCREEN_WIDTH,SCREEN_HEIGHT))

pygame.display.set_caption('Q14')



vel = 5

run = True
while run:

   pygame.time.delay(100)

   for event in pygame.event.get():
       if event.type == pygame.QUIT:
           run = False

       elif event.type == pygame.MOUSEBUTTONDOWN:
           mouseX, mouseY = event.pos
           X = mouseX - width//2
           Y = mouseY - height//2

   win.fill(BLACK)
   pygame.draw.rect(win,RED,(X, Y,width,height ) )
   pygame.display.update()

pygame.display.quit()

pygame.quit()
```



15)

```
import pygame


SCREEN_WIDTH = 500
SCREEN_HEIGHT= 500

width = 100
height= 100

X = SCREEN_WIDTH//2 - width//2
Y = SCREEN_HEIGHT//2 - height//2
RED = (255,0,0)
BLACK = (0,0,0)
pygame.init()
```

```
win = pygame.display.set_mode((SCREEN_WIDTH,SCREEN_HEIGHT))

pygame.display.set_caption('Q15')



vel = 5

run = True
while run:

   pygame.time.delay(100)

   for event in pygame.event.get():
       if event.type == pygame.QUIT:
           run = False



   win.fill(BLACK)

   pygame.draw.polygon(win, RED, [(40, 40), (60, 40), (65, 50), (50, 60),(35,50)])

   pygame.draw.polygon(win,RED, [(40,40),(60,40),(50, 20)])
   pygame.draw.polygon(win, RED, [(60, 40), (65, 50), (80, 45)])
   pygame.draw.polygon(win, RED, [(65, 50), (50, 60), (62, 70)])
   pygame.draw.polygon(win, RED, [(50, 60), (35, 50), (32, 70)])
   pygame.draw.polygon(win, RED, [(35, 50), (40, 40), (20, 45)])


   pygame.display.update()

pygame.display.quit()

pygame.quit()
```

16)

17)

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

#CONSTANTES
#constantes para tamanho de tela

SCREEN_WIDTH = 400
SCREEN_HEIGHT = 300

#será utilizada para velocidade do jogo

FPS = 200

#Valores para o desenho das paletas e do fundo
LINE_WIDTH = 10
BAR_HEIGHT = 50
```

```
BAR_PADDING = 20
score = 0
#CORES
BLACK = (0,0,0)
WHITE = (255,255,255)

def arena_draw():
   DISPLAYSURF.fill(BLACK)

   #desenha a quadra

   pygame.draw.rect(DISPLAYSURF, WHITE,((0,0),(SCREEN_WIDTH,SCREEN_HEIGHT)),
LINE_WIDTH*2)
   pygame.draw.line(DISPLAYSURF, WHITE, ((SCREEN_WIDTH // 2), 0), ((SCREEN_WIDTH //
2), SCREEN_HEIGHT),(LINE_WIDTH // 4))

def draw_bar(bar):

   #impede da paleta ir além da borda do fundo
   if bar.bottom > SCREEN_HEIGHT - LINE_WIDTH:
       bar.bottom = SCREEN_HEIGHT - LINE_WIDTH

   #impede da paleta ir além da borda do topo

   if bar.top < LINE_WIDTH:
       bar.top = LINE_WIDTH

   #desenha a paleta

   pygame.draw.rect(DISPLAYSURF,WHITE, bar)

def draw_ball(ball):
   pygame.draw.rect(DISPLAYSURF, WHITE, ball)



def move_ball(ball, ballDirX, ballDirY):
   ball.x += ballDirX
   ball.y += ballDirY

   return ball

#verifica se existe colisão com as bordas
#Retorna uma nova posição caso exista colisão

def colision_detection(ball, ballDirX, ballDirY):
   if ball.top <= (LINE_WIDTH) or ball.bottom >= (SCREEN_HEIGHT - LINE_WIDTH):
       ballDirY *= -1

   if ball.left <= (LINE_WIDTH) or ball.right >= (SCREEN_WIDTH - LINE_WIDTH):
       ballDirX *= -1

   return ballDirX,ballDirY

def artificial_inteligency(ball, ballDirX, bar2):
   #movimentar a paleta quando a bola vem em direção da paleta
   if ballDirX >= 1:
       if bar2.centery < ball.centery:
           bar2.y += 1
       else:
           bar2.y -= 1

   return bar2

#Verifica a colisão da bola com a paleta 1 e 2
def ball_collision(ball, bar1, bar2, ballDirX):
```

```
    if ballDirX < 0 and  bar1.right == ball.left and bar1.top <= ball.top and
bar1.bottom >= ball.bottom:
        return -1
    elif ballDirX == 1 and bar2.left == ball.right and bar2.top <= ball.top and
bar2.bottom >= ball.bottom:
        return -1

    else: return 1

#verifica se um jogador fez ponto e retorna o novo valor do placar
def score_check(bar1, ball, score, ballDirX):
    #zera a contagem se a bola acerta a borda do jogador
    if ball.left == LINE_WIDTH:
        return 0
    elif ball.right == SCREEN_WIDTH - LINE_WIDTH:
        score += 10
        return score
    elif ballDirX > 0 and bar1.right == ball.left and bar1.top < ball.top and
bar1.bottom > ball.bottom:
        score += 1
        return score

    else: return score

def draw_score(score):
    resultSurf = BASICFONT.render('Score = %s' %(score), True, WHITE)
    resultRect = resultSurf.get_rect()
    resultRect.topleft = (SCREEN_WIDTH - 150, 25)
    DISPLAYSURF.blit(resultSurf, resultRect)

def main():

    pygame.init()
    global DISPLAYSURF

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

    FPSCLOCK = pygame.time.Clock()
    DISPLAYSURF = pygame.display.set_mode((SCREEN_WIDTH,SCREEN_HEIGHT))
    pygame.display.set_caption('PONG')

    #iniciando as variáveis nas posições iniciais
    #Estas variáveis serão alteradas ao longo da execução
    bolaX = SCREEN_WIDTH//2 - LINE_WIDTH//2
    bolaY = SCREEN_HEIGHT//2 - LINE_WIDTH//2


    playerOne_position =(SCREEN_HEIGHT-BAR_HEIGHT)//2
    playerTwo_position = (SCREEN_HEIGHT - BAR_HEIGHT)//2

    score = int(0)

    #criando rentagulos para a bola e paletas
    bar1 = pygame.Rect(BAR_PADDING, playerOne_position,LINE_WIDTH, BAR_HEIGHT)
    bar2 = pygame.Rect(SCREEN_WIDTH - BAR_PADDING - LINE_WIDTH,playerTwo_position,
LINE_WIDTH, BAR_HEIGHT)
    ball = pygame.Rect(bolaX,bolaY,LINE_WIDTH,LINE_WIDTH)

    #altera posição da bola
    ballDirX = -1
    ballDirY = -1

    #desenhando as posições iniciais da arena
```

```
    arena_draw()
    draw_bar(bar1)
    draw_bar(bar2)
    draw_ball(ball)

    done = False

    while not done:


        #checa os eventos do mouse
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                done = True

            #move o jogador
            if event.type == MOUSEMOTION:
                mouseX, mouseY = event.pos
                bar1.y = mouseY

        arena_draw()
        draw_bar(bar1)
        draw_bar(bar2)
        draw_ball(ball)

        ball = move_ball(ball, ballDirX, ballDirY)
        ballDirX, ballDirY = colision_detection(ball, ballDirX, ballDirY)
        ballDirX = ballDirX * ball_collision(ball, bar1, bar2, ballDirX)
        bar2 = artificial_inteligency(ball, ballDirX, bar2)
        score = score_check(bar1, ball, score , ballDirX)
        draw_score(score)

        if score % 10 == 0 and score > 0:
            ballDirX += 1
            ballDirY += 1
        # atualiza o desenho na tela

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

    #finaliza o display
    pygame.display.quit()

    #finaliza o pygame

    pygame.quit()
    sys.exit()

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

18)

19)

20)

21)

22)

23)

24)