

```

from browser import timer
import random

global running
global runtimer
runtimer = True
running = False
delay = int(random.randint(40, 100))

begining = True

modeRJ = False

cardWidth = 75
cardHeight = 120

xMid = get_width() / 2
yMid = get_height() / 2

cW = cardWidth / 2
cH = cardHeight / 2

cardDelay = 1000

compCardH = 0

cardSlot1H = (get_height() - cardHeight)
cardSlot1 = (0 + 2.25 + 15)

cardSlot2 = (0 + (cardWidth + 2.25)+ (15*2))

cardSlot3 = (0 + (cardWidth * 2 + 2.25)+ (15*3))

cardSlot4 = (0 + (cardWidth * 3 + 2.25)+ (15*4))
#####

def stop_timer():
    if timer_is_active:
        timer.clear_timeout(timeout_id)
        win()

```

```

#Timers for the spoon durratation
def start_timer():
    global timer_is_active, timeout_id, delay
    timer_is_active = True
    timeout_id = timer.set_timeout(loss, delay)

def randomColor():
    choice = random.randint(1,2)
    if choice == 1:
        return "black"
    else:
        return "red"

def spoonChecker(c1, c2, c3, c4): # the math for checking if all your
cards are equill
    global modeRJ
    if c1 == c2 and c3 == c4 and c1 == c3:
        if modeRJ == False:
            spoon()
        else:
            spoonRJ()
    else:
        return False

def deck():
    values = ["2", "3", "4", "5", "6", "7", "8", "9", "10", "Jack",
"Queen", "King", "Ace", "2", "3", "4", "5", "6", "7", "8", "9", "10",
"Jack", "Queen", "King", "Ace", "2", "3", "4", "5", "6", "7", "8", "9",
"10", "Jack", "Queen", "King", "Ace", "2", "3", "4", "5", "6", "7", "8",
"9", "10", "Jack", "Queen", "King", "Ace"]
    random.shuffle(values)
    return values[0]
    values.pop(0)
    if len(values) == 0:
        values.append("2", "3", "4", "5", "6", "7", "8", "9", "10",
"Jack", "Queen", "King", "Ace", "2", "3", "4", "5", "6", "7", "8", "9",
"10", "Jack", "Queen", "King", "Ace", "2", "3", "4", "5", "6", "7", "8",
"9", "10", "Jack", "Queen", "King", "Ace", "2", "3", "4", "5", "6", "7",
"8", "9", "10", "Jack", "Queen", "King", "Ace")

```

```

def refreshCard(card):
    refresh = True
    if card == 2:
        computerCard(cardSlot2, refresh)
    elif card == 3:
        computerCard(cardSlot3, refresh)
    elif card == 4:
        computerCard(cardSlot4, refresh)

def computerHand():
    global cHand
    cHand = []
    cCard1 = deck()
    cCard2 = deck()
    cCard3 = deck()
    cCard4 = deck()
    cHand.append(cCard1)
    cHand.append(cCard2)
    cHand.append(cCard3)
    cHand.append(cCard4)

def playerHand():
    global pHand
    pHand = []
    hCard1 = deck()
    hCard2 = deck()
    hCard3 = deck()
    hCard4 = deck()
    pHand.append(hCard1)
    pHand.append(hCard2)
    pHand.append(hCard3)
    pHand.append(hCard4)

def displayCard(card, slot, color): #Takes players hand and displays it
    if card == "2":
        return card2(slot, color)
    elif card == "3":
        return card3(slot, color)
    elif card == "4":
        return card4(slot, color)

```

```

elif card == "5":
    return card5(slot, color)
elif card == "6":
    return card6(slot, color)
elif card == "7":
    return card7(slot, color)
elif card == "8":
    return card8(slot, color)
elif card == "9":
    return card9(slot, color)
elif card == "10":
    return card10(slot, color)
elif card == "Jack":
    return jack(slot, color)
elif card == "Queen":
    return queen(slot, color)
elif card == "King":
    return king(slot, color)
else:
    return ace(slot, color)

def computerChangeCard(cardNum):
    if cardNum == cHand[1]:
        deck1 = deck()
        cHand[1] = deck1
        refreshCard(2)
        spoonChecker(cHand[0], cHand[1], cHand[2], cHand[3])
    elif cardNum == cHand[2]:
        deck2 = deck()
        cHand[2] = deck2
        refreshCard(3)
        spoonChecker(cHand[0], cHand[1], cHand[2], cHand[3])
    elif cardNum == cHand[3]:
        deck3 = deck()
        cHand[3] = deck3
        refreshCard(3)
        spoonChecker(cHand[0], cHand[1], cHand[2], cHand[3])

def changeCard(x, y):
    global running

```

```

if running == True:
    if y > cardSlot1H:
        if x > cardSlot1 and x < cardSlot2:
            deck0 = deck()
            displayCard(deck0, cardSlot1, randomColor())
            pHand[0] = deck0
            spoonChecker(pHand[0], pHand[1], pHand[2], pHand[3])
            print (pHand)
        elif x > cardSlot2 and x < cardSlot3:
            deck1 = deck()
            displayCard(deck1, cardSlot2, randomColor())
            pHand[1] = deck1
            spoonChecker(pHand[0], pHand[1], pHand[2], pHand[3])
            print (pHand)
        elif x > cardSlot3 and x < cardSlot4:
            deck2 = deck()
            displayCard(deck2, cardSlot3, randomColor())
            pHand[2] = deck2
            spoonChecker(pHand[0], pHand[1], pHand[2], pHand[3])
            print (pHand)
        else:
            deck3 = deck()
            displayCard(deck3, cardSlot4, randomColor())
            pHand[3] = deck3
            spoonChecker(pHand[0], pHand[1], pHand[2], pHand[3])
            print (pHand)

def cardlogic(c1, c2, c3, c4): # this logic is if/which the card should be
changed
    if c1 == c2:
        if c2 == c3:
            if c3 == c4:
                pass
            else:
                return cHand[3]
        else:
            return cHand[2]
    else:
        return cHand[1]

```

```

def run():
    global runtimer
    if runtimer == True:
        runtimer = False
        computerPlay()

def runTimer():
    global running
    global runtimer
    global cardDelay
    if running == True:
        print("Waitng")
        runtimer = True
        timeout_id = timer.set_timeout(run, cardDelay) # Each 1,000 = 1
second

def playAgain():
    runTimer()

def computerPlay():
    global running
    global runtimer
    if running == True:
        if runtimer == False:
            result = cardlogic(cHand[0], cHand[1], cHand[2], cHand[3])
            if result == cHand[1]:
                computerChangeCard(cHand[1])
                print("Comp " + str(cHand))
                playAgain()
            elif result == cHand[2]:
                computerChangeCard(cHand[2])
                print("Comp " + str(cHand))
                playAgain()
            elif result == cHand[3]:
                computerChangeCard(cHand[3])
                print("Comp " + str(cHand))
                playAgain()

#//////////////////////////////////CARD GRAPHICS//////////////////////////////////

```

```

def spades(x, y):
    pass
def clubs(x, y):
    pass
def hearts(x, y):
    pass
def spades(x, y):
    pass

def spoon():
    global remove, scircle, srect
    if remove == False:
        scircle = Circle(25)
        scircle.set_position(xMid - 45, yMid)
        scircle.set_color(Color.grey)
        add(scircle)
        srect = Rectangle(100, 18)
        srect.set_position(xMid - 45, yMid - 9)
        srect.set_color(Color.grey)
        add(srect)
        running = False
        add_mouse_click_handler(pressSpoon)
        start_timer()
        # X length == 125  Y length == 25

def spoonRJ():
    global remove, scircle, srect, sxMid, syMid
    # Generate random positions within the specified ranges
    sxMid = random.randint(0, 400)
    syMid = random.randint(126, 346)

    if remove == False:
        scircle = Circle(25)
        scircle.set_position(sxMid - 45, syMid)
        scircle.set_color(Color.grey)
        add(scircle)
        srect = Rectangle(100, 18)
        srect.set_position(sxMid - 45, syMid - 9)
        srect.set_color(Color.grey)
        add(srect)

```

```

        running = False
        add_mouse_click_handler(pressRJSpoon)
        start_timer()

def card2(slot, color):
    outline = Rectangle(cardWidth + 5, cardHeight + 5)
    outline.set_position(slot - 2.25, cardSlot1H - 2.25)
    outline.set_color(Color.black)
    add(outline)
    rect2 = Rectangle(cardWidth, cardHeight)
    rect2.set_position(slot, cardSlot1H)
    rect2.set_color(Color.white)
    add(rect2)

    txt2 = Text("2")
    txt2.set_position(slot + 25, cardSlot1H + 60)
    if "r" in color:
        txt2.set_color(Color.red)
    else:
        txt2.set_color(Color.black)
    txt2.set_font("30pt Arial")
    add(txt2)

def card3(slot, color):
    outline = Rectangle(cardWidth + 5, cardHeight + 5)
    outline.set_position(slot - 2.25, cardSlot1H - 2.25)
    outline.set_color(Color.black)
    add(outline)
    rect3 = Rectangle(cardWidth, cardHeight)
    rect3.set_position(slot, cardSlot1H)
    rect3.set_color(Color.white)
    add(rect3)
    txt3 = Text("3")
    txt3.set_position(slot + 25, cardSlot1H + 60)
    if "r" in color:
        txt3.set_color(Color.red)
    else:
        txt3.set_color(Color.black)
    txt3.set_font("30pt Arial")
    add(txt3)

def card4(slot, color):

```



```

outline = Rectangle(cardWidth + 5, cardHeight + 5)
outline.set_position(slot - 2.25, cardSlot1H - 2.25)
outline.set_color(Color.black)
add(outline)
rect4 = Rectangle(cardWidth, cardHeight)
rect4.set_position(slot, cardSlot1H)
rect4.set_color(Color.white)
add(rect4)
txt4 = Text("4")
txt4.set_position(slot + 25, cardSlot1H + 60)
if "r" in color:
    txt4.set_color(Color.red)
else:
    txt4.set_color(Color.black)
txt4.set_font("30pt Arial")
add(txt4)
def card5(slot, color):
    outline = Rectangle(cardWidth + 5, cardHeight + 5)
    outline.set_position(slot - 2.25, cardSlot1H - 2.25)
    outline.set_color(Color.black)
    add(outline)
    rect5 = Rectangle(cardWidth, cardHeight)
    rect5.set_position(slot, cardSlot1H)
    rect5.set_color(Color.white)
    add(rect5)
    txt5 = Text("5")
    txt5.set_position(slot + 25, cardSlot1H + 60)
    if "r" in color:
        txt5.set_color(Color.red)
    else:
        txt5.set_color(Color.black)
    txt5.set_font("30pt Arial")
    add(txt5)
def card6(slot, color):
    outline = Rectangle(cardWidth + 5, cardHeight + 5)
    outline.set_position(slot - 2.25, cardSlot1H - 2.25)
    outline.set_color(Color.black)
    add(outline)
    rect6 = Rectangle(cardWidth, cardHeight)
    rect6.set_position(slot, cardSlot1H)

```

```

rect6.set_color(Color.white)
add(rect6)
txt6 = Text("6")
txt6.set_position(slot + 25, cardSlot1H + 60)
if "r" in color:
    txt6.set_color(Color.red)
else:
    txt6.set_color(Color.black)
txt6.set_font("30pt Arial")
add(txt6)
def card7(slot, color):
    outline = Rectangle(cardWidth + 5, cardHeight + 5)
    outline.set_position(slot - 2.25, cardSlot1H - 2.25)
    outline.set_color(Color.black)
    add(outline)
    rect7 = Rectangle(cardWidth, cardHeight)
    rect7.set_position(slot, cardSlot1H)
    rect7.set_color(Color.white)
    add(rect7)
    txt7 = Text("7")
    txt7.set_position(slot + 25, cardSlot1H + 60)
    if "r" in color:
        txt7.set_color(Color.red)
    else:
        txt7.set_color(Color.black)
    txt7.set_font("30pt Arial")
    add(txt7)
def card8(slot, color):
    outline = Rectangle(cardWidth + 5, cardHeight + 5)
    outline.set_position(slot - 2.25, cardSlot1H - 2.25)
    outline.set_color(Color.black)
    add(outline)
    rect8 = Rectangle(cardWidth, cardHeight)
    rect8.set_position(slot, cardSlot1H)
    rect8.set_color(Color.white)
    add(rect8)
    txt8 = Text("8")
    txt8.set_position(slot + 25, cardSlot1H + 60)
    if "r" in color:
        txt8.set_color(Color.red)

```

```

else:
    txt8.set_color(Color.black)
    txt8.set_font("30pt Arial")
    add(txt8)
def card9(slot, color):
    outline = Rectangle(cardWidth + 5, cardHeight + 5)
    outline.set_position(slot - 2.25, cardSlot1H - 2.25)
    outline.set_color(Color.black)
    add(outline)
    rect9 = Rectangle(cardWidth, cardHeight)
    rect9.set_position(slot, cardSlot1H)
    rect9.set_color(Color.white)
    add(rect9)
    txt9 = Text("9")
    txt9.set_position(slot + 25, cardSlot1H + 60)
    if "r" in color:
        txt9.set_color(Color.red)
    else:
        txt9.set_color(Color.black)
    txt9.set_font("30pt Arial")
    add(txt9)
def card10(slot, color):
    outline = Rectangle(cardWidth + 5, cardHeight + 5)
    outline.set_position(slot - 2.25, cardSlot1H - 2.25)
    outline.set_color(Color.black)
    add(outline)
    rect10 = Rectangle(cardWidth, cardHeight)
    rect10.set_position(slot, cardSlot1H)
    rect10.set_color(Color.white)
    add(rect10)
    txt10 = Text("10")
    txt10.set_position(slot + 15, cardSlot1H + 60)
    if "r" in color:
        txt10.set_color(Color.red)
    else:
        txt10.set_color(Color.black)
    txt10.set_font("30pt Arial")
    add(txt10)
def jack(slot, color):
    outline = Rectangle(cardWidth + 5, cardHeight + 5)

```

```

outline.set_position(slot - 2.25, cardSlot1H - 2.25)
outline.set_color(Color.black)
add(outline)

rectJ = Rectangle(cardWidth, cardHeight)
rectJ.set_position(slot, cardSlot1H)
rectJ.set_color(Color.white)
add(rectJ)

txtJ = Text("J")
txtJ.set_position(slot + 25, cardSlot1H + 60)
if "r" in color:
    txtJ.set_color(Color.red)
else:
    txtJ.set_color(Color.black)
txtJ.set_font("30pt Arial")
add(txtJ)

def queen(slot, color):
    outline = Rectangle(cardWidth + 5, cardHeight + 5)
    outline.set_position(slot - 2.25, cardSlot1H - 2.25)
    outline.set_color(Color.black)
    add(outline)

    rectQ = Rectangle(cardWidth, cardHeight)
    rectQ.set_position(slot, cardSlot1H)
    rectQ.set_color(Color.white)
    add(rectQ)

    txtQ = Text("Q")
    txtQ.set_position(slot + 25, cardSlot1H + 60)
    if "r" in color:
        txtQ.set_color(Color.red)
    else:
        txtQ.set_color(Color.black)
    txtQ.set_font("30pt Arial")
    add(txtQ)

def king(slot, color):
    outline = Rectangle(cardWidth + 5, cardHeight + 5)
    outline.set_position(slot - 2.25, cardSlot1H - 2.25)
    outline.set_color(Color.black)
    add(outline)

    rectK = Rectangle(cardWidth, cardHeight)
    rectK.set_position(slot, cardSlot1H)
    rectK.set_color(Color.white)

```

```

add(rectK)
txtK = Text("K")
txtK.set_position(slot + 25, cardSlot1H + 60)
if "r" in color:
    txtK.set_color(Color.red)
else:
    txtK.set_color(Color.black)
txtK.set_font("30pt Arial")
add(txtK)
def ace(slot, color):
    outline = Rectangle(cardWidth + 5, cardHeight + 5)
    outline.set_position(slot - 2.25, cardSlot1H - 2.25)
    outline.set_color(Color.black)
    add(outline)
    rectA = Rectangle(cardWidth, cardHeight)
    rectA.set_position(slot, cardSlot1H)
    rectA.set_color(Color.white)
    add(rectA)
    txtA = Text("A")
    txtA.set_position(slot + 25, cardSlot1H + 60)
    if "r" in color:
        txtA.set_color(Color.red)
    else:
        txtA.set_color(Color.black)
    txtA.set_font("30pt Arial")
    add(txtA)
def computerCard(slot, refresh):
    outline = Rectangle(cardWidth + 5, cardHeight + 5)
    outline.set_position(slot - 2.25, compCardH - 2.25)
    outline.set_color(Color.black)
    add(outline)
    rectA = Rectangle(cardWidth, cardHeight)
    rectA.set_position(slot, compCardH)
    rectA.set_color(Color.red)
    add(rectA)
    crossline = Line(slot, compCardH, slot + cardWidth, compCardH +
cardHeight)
    crossline.set_color(Color.white)
    add(crossline)

```

```

        for i in range(random.randint(1, cardWidth)):
            crossline = Line(slot + (i+1), compCardH, slot + cardWidth,
compCardH + cardHeight)
            crossline.set_color(Color.white)
            add(crossline)

def loss():
    global remove
    global running
    if running == True:
        remove = True
        running = False
        txtL = Text("YOU LOSE :(")
        txtL.set_position(xMid - 200, yMid)
        txtL.set_color(Color.black)
        txtL.set_font("50pt Arial")
        add(txtL)

def win():
    global remove
    global running
    if running == True:
        remove = True
        running = False
        txtL = Text("YOU WIN :)")
        txtL.set_position(xMid - 180, yMid)
        txtL.set_color(Color.black)
        txtL.set_font("50pt Arial")
        add(txtL)
        spoon()

def gamemode(done):
    global Eoutline, Erect, txtE, Moutline, Mrect, txtM, Houtline, Hrect,
txtH, Ioutline, Irect, txtI, RJoutline, RJrect, RJtxt
    if done == "no" or done == "No":
        Eoutline = Rectangle(cardHeight + 5, cardWidth + 5)
        Eoutline.set_position(10 - 2.5, yMid - (cardWidth / 2) - 2.5)
        Eoutline.set_color(Color.black)
        add(Eoutline)
        Erect = Rectangle(cardHeight, cardWidth)

```

```

Erect.set_position(10, yMid - cardWidth / 2)
Erect.set_color(Color.green)
add(Erect)
txtE = Text("Easy")
txtE.set_position((10 - 2.5) + 25, yMid + 12.5)
txtE.set_color(Color.black)
txtE.set_font("25pt Arial")
add(txtE)
#####
Moutline = Rectangle(cardHeight + 5, cardWidth + 5)
Moutline.set_position((20 - 2.5) + cardHeight, yMid - (cardWidth /
2) - 2.5)
Moutline.set_color(Color.black)
add(Moutline)
Mrect = Rectangle(cardHeight, cardWidth)
Mrect.set_position((20) + cardHeight, yMid - cardWidth / 2)
Mrect.set_color(Color.yellow)
add(Mrect)
txtM = Text("Medium")
txtM.set_position((20 - 2.5) + cardHeight + 2.5, yMid + 12.5)
txtM.set_color(Color.black)
txtM.set_font("25pt Arial")
add(txtM)
#####
Houtline = Rectangle(cardHeight + 5, cardWidth + 5)
Houtline.set_position((20 - 2.5) + (cardHeight * 2) + 10, yMid -
(cardWidth / 2) - 2.5)
Houtline.set_color(Color.black)
add(Houtline)
Hrect = Rectangle(cardHeight, cardWidth)
Hrect.set_position((20) + (cardHeight * 2) + 10, yMid - cardWidth
/ 2)
Hrect.set_color(Color.red)
add(Hrect)
txtH = Text("Hard")
txtH.set_position((20 - 2.5) + (cardHeight * 2) + 25 + 10, yMid +
12.5)
txtH.set_color(Color.black)
txtH.set_font("25pt Arial")
add(txtH)

```

```

#####
Ioutline = Rectangle((cardHeight * 3) + 25, cardWidth + 5)
Ioutline.set_position(10 - 2.5, (yMid - 2.5) + cardWidth - 25)
Ioutline.set_color(Color.black)
add(Ioutline)

Irect = Rectangle((cardHeight * 3) + 20, cardWidth)
Irect.set_position(10, yMid + cardWidth - 25)
Irect.set_color(Color.purple)
add(Irect)

txtI = Text("IMPOSSIBLE")
txtI.set_position((xMid - 100), yMid + 22.5 + cardWidth)
txtI.set_color(Color.black)
txtI.set_font("25pt Arial")
add(txtI)

#####
RJoutline = Rectangle(cardHeight + 5, cardWidth + 5)
RJoutline.set_position(10,10)
RJoutline.set_color(Color.black)
add(RJoutline)

RJrect = Rectangle(cardHeight, cardWidth)
RJrect.set_position(12.5, 12.5)
RJrect.set_color('#00A67E')
add(RJrect)

RJtxt = Text("RJ")
RJtxt.set_position(cardWidth - 25, cardHeight / 2)
RJtxt.set_color(Color.black)
RJtxt.set_font("25pt Arial")
add(RJtxt)

elif done.lower() == "yes":
    print("Removing")
    #####
    remove(Eoutline)
    remove(Erect)
    remove(txtE)
    #####
    remove(Moutline)
    remove(Mrect)
    remove(txtM)
    #####
    remove(Houtline)

```



```

        remove(Hrect)
        remove(txtH)
        #////////////////////////////////////
        remove(Ioutline)
        remove(Irect)
        remove(txtI)
        #////////////////////////////////////
        remove(RJoutline)
        remove(RJrect)
        remove(RJtxt)
        #////////////////////////////////////
        print("Removed")
        play()

#////////////////////////////////////
def rounds():
    if spoonChecker(cHand[0], cHand[1], cHand[2], cHand[3]) == False and
    spoonChecker(pHand[0], pHand[1], pHand[2], pHand[3]) == False:
        displayCard(pHand[0], cardSlot1, randomColor())
        displayCard(pHand[1], cardSlot2, randomColor())
        displayCard(pHand[2], cardSlot3, randomColor())
        displayCard(pHand[3], cardSlot4, randomColor())

def pressSpoon(x, y): #This checks if you have pressed the spoon
    if (y > (210)) and (y < (260)):
        if (x > (135)) and (x < (270)):
            stop_timer()

def pressRJSpoon(x, y):
    global sxMid, syMid
    if (y > (syMid - 50)) and (y < (syMid + 50)):
        if (x > (syMid - 135)) and (x < (syMid + 135)):
            stop_timer()

def testPress(x, y):
    if (y > (210)) and (y < (260)):
        if (x > (135)) and (x < (270)):
            print("Yep!")

def wherePress(x, y):

```

```

    print(str(x), str(y))

def testSquare():
    trect = Circle(5)
    trect.set_position(xMid - 65, yMid - 25)
    trect.set_color(Color.red)
    add(trect)

    t2rect = Circle(5)
    t2rect.set_position(xMid - 65, yMid + 25)
    t2rect.set_color(Color.red)
    add(t2rect)

    t3rect = Circle(5)
    t3rect.set_position(xMid + 65, yMid - 25)
    t3rect.set_color(Color.red)
    add(t3rect)

    t4rect = Circle(5)
    t4rect.set_position(xMid + 65, yMid + 25)
    t4rect.set_color(Color.red)
    add(t4rect)

def play():
    global remove
    global running
    print("Playing")
    remove = False
    running = True
    add_mouse_click_handler(changeCard)
    playerHand()
    computerHand()
    rounds()
    runTimer()

    computerCard(cardSlot1, False)
    computerCard(cardSlot2, False)
    computerCard(cardSlot3, False)
    computerCard(cardSlot4, False)

```

```

def removeGame(): # removes the game selector menu
    global Eoutline
    global Erect
    global txtE
    global Moutline
    global Mrect
    global txtM
    global Houtline
    global Hrect
    global txtH
    print("Removing")
    #////////////////////
    remove(Eoutline)
    remove(Erect)
    remove(txtE)
    #////////////////////
    remove(Moutline)
    remove(Mrect)
    remove(txtM)
    #////////////////////
    remove(Houtline)
    remove(Hrect)
    remove(txtH)
    #////////////////////
    print("Removed")
    play()

def selector(x, y):
    global cardDelay, begining, delay, modeRJ
    if begining == True:
        if y > 200 and y < 280: # Paramaters for click selectors
            if x > 10 and x < 130:
                begining = False
                print("E")
                cardDelay = 750
                delay = int(random.randint(500, 1000))
                gamemode("yes")
            elif x > 140 and x < 260:
                begining = False
                print("M")

```

```
        cardDelay = 500
        delay = int(random.randint(300, 500))
        gamemode("yes")
    elif x > 270 and x < 400:
        begining = False
        print("H")
        cardDelay = 250
        delay = int(random.randint(250, 300))
        gamemode("yes")
elif y > 280 and y < 360:
    begining = False
    print("I")
    cardDelay = 100
    delay = int(random.randint(0, 100))
    gamemode("yes")
elif y > 0 and y < 80:
    if x > 0 and x < 135:
        begining = False
        modeRJ = True
        print("RJ")
        cardDelay = 50
        delay = 0
        gamemode("yes")

#////////////////////////////////////
gamemode("no")
add_mouse_click_handler(selector)
#//////////////////////////////////// END //////////////////////////////////
```