#Sagar Patel--------------------------------------------------------------------------------------------#Name #

#ICS 201------------------------------------------------------------------------------------------------#Course #

#October 2, 2019----------------------------------------------------------------------------------------#Date Started #

#Mr. Moore----------------------------------------------------------------------------------------------#Teacher #

#--------------------------------- PACKAGES + SCREEN SPECIFIC + RECURSION LIMIT ------------------------#---------------------------------------------------#

import pygame,os,sys,random,time #imports needed packages #

sys.setrecursionlimit(100000) #Sets recursion limit so game doesn't crash #

os.environ['SDL\_VIDEO\_WINDOW\_POS'] = "%d,%d" % (0,0) #Places pygame window in top left corner #

pygame.init() #Required for all Pygame packages to be active #

#pygame.mixer.pre\_init() #Required for Pygame Music #

#pygame.mixer.init() #Required for Pygame Music #

screen = pygame.display.set\_mode((600, 800), pygame.NOFRAME) #sets the screen size of the game #

#--------------------------------- FONTS + IMAGES + TIME + MOUSE + WINDOW ICON -------------------------#---------------------------------------------------#

fontsize1 = pygame.font.Font('American Captain.ttf', 150) #imports fonts and different sizes #

fontsize2 = pygame.font.Font('American Captain.ttf', 50) #imports fonts and different sizes #

fontsize3 = pygame.font.Font('American Captain.ttf', 100) #imports fonts and different sizes #

fontsize4 = pygame.font.Font('American Captain.ttf', 130) #imports fonts and different sizes #

fontsize5 = pygame.font.Font('American Captain.ttf', 70) #imports fonts and different sizes #

fontsize6 = pygame.font.Font('American Captain.ttf', 95) #imports fonts and different sizes #

fontsize200 = pygame.font.Font('American Captain.ttf', 30) #imports fonts and different sizes #

fontsize300 = pygame.font.Font('American Captain.ttf', 37) #imports fonts and different sizes #

icon1 = pygame.image.load('blackjackicon.png') #imports image for game #

icon2 = pygame.image.load('cardmenusuits.png') #imports image for game #

icon3 = pygame.image.load('mousecursor.png') #imports image for game #

icon4 = pygame.image.load('dealerface.png') #image for dealer icon #

pygame.mouse.set\_visible(False) #This hides the cursor so I can add my custom #

pygame.display.set\_icon(icon1) #This makes the icon of the game #

pygame.display.set\_caption('BlackJack - Sagar') #Changes the name of game #

clock = pygame.time.Clock() #This sets the clock of the game for refresh-rate #

#buttonSound = pygame.mixer.Sound('Button.wav') #Imports sound for button #

#winSound = pygame.mixer.Sound('Congrats.wav') #Imports sound for win screen #

#loseSound = pygame.mixer.Sound('Loser.wav') #Imports sound for lose screen #

#music = pygame.mixer.music.load ('BlackJack Music.mp3') #Imports sound for background music #

#pygame.mixer.music.set\_volume(0.8) #Sets volume for background sound #

#pygame.mixer.music.play(loops=-1) #loops background sound forever #

#--------------------------------- CARDS ---------------------------------------------------------------#---------------------------------------------------#

cardAC = pygame.image.load('AC.png') #imports image for game #

cardAD = pygame.image.load('AD.png') #imports image for game #

cardAH = pygame.image.load('AH.png') #imports image for game #

cardAS = pygame.image.load('AS.png') #imports image for game #

card2C = pygame.image.load('2C.png') #imports image for game #

card2D = pygame.image.load('2D.png') #imports image for game #

card2H = pygame.image.load('2H.png') #imports image for game #

card2S = pygame.image.load('2S.png') #imports image for game #

card3C = pygame.image.load('3C.png') #imports image for game #

card3D = pygame.image.load('3D.png') #imports image for game #

card3H = pygame.image.load('3H.png') #imports image for game #

card3S = pygame.image.load('3S.png') #imports image for game #

card4C = pygame.image.load('4C.png') #imports image for game #

card4D = pygame.image.load('4D.png') #imports image for game #

card4H = pygame.image.load('4H.png') #imports image for game #

card4S = pygame.image.load('4S.png') #imports image for game #

card5C = pygame.image.load('5C.png') #imports image for game #

card5D = pygame.image.load('5D.png') #imports image for game #

card5H = pygame.image.load('5H.png') #imports image for game #

card5S = pygame.image.load('5S.png') #imports image for game #

card6C = pygame.image.load('6C.png') #imports image for game #

card6D = pygame.image.load('6D.png') #imports image for game #

card6H = pygame.image.load('6H.png') #imports image for game #

card6S = pygame.image.load('6S.png') #imports image for game #

card7C = pygame.image.load('7C.png') #imports image for game #

card7D = pygame.image.load('7D.png') #imports image for game #

card7H = pygame.image.load('7H.png') #imports image for game #

card7S = pygame.image.load('7S.png') #imports image for game #

card8C = pygame.image.load('8C.png') #imports image for game #

card8D = pygame.image.load('8D.png') #imports image for game #

card8H = pygame.image.load('8H.png') #imports image for game #

card8S = pygame.image.load('8S.png') #imports image for game #

card9C = pygame.image.load('9C.png') #imports image for game #

card9D = pygame.image.load('9D.png') #imports image for game #

card9H = pygame.image.load('9H.png') #imports image for game #

card9S = pygame.image.load('9S.png') #imports image for game #

card10C = pygame.image.load('10C.png') #imports image for game #

card10D = pygame.image.load('10D.png') #imports image for game #

card10H = pygame.image.load('10H.png') #imports image for game #

card10S = pygame.image.load('10S.png') #imports image for game #

cardJC = pygame.image.load('JC.png') #imports image for game #

cardJD = pygame.image.load('JD.png') #imports image for game #

cardJH = pygame.image.load('JH.png') #imports image for game #

cardJS = pygame.image.load('JS.png') #imports image for game #

cardQC = pygame.image.load('QC.png') #imports image for game #

cardQD = pygame.image.load('QD.png') #imports image for game #

cardQH = pygame.image.load('QH.png') #imports image for game #

cardQS = pygame.image.load('QS.png') #imports image for game #

cardKC = pygame.image.load('KC.png') #imports image for game #

cardKD = pygame.image.load('KD.png') #imports image for game #

cardKH = pygame.image.load('KH.png') #imports image for game #

cardKS = pygame.image.load('KS.png') #imports image for game #

BACKSIDE = pygame.image.load('Back side Of Card.png') #imports image for game #

#--------------------------------- FUNCTIONS -----------------------------------------------------------#---------------------------------------------------#

def quitgame (): #This is the quit game function #

pygame.quit () #Properly quits pygame #

quit () #Quits the window #

#--------------------------------- REUSABLE CODE -------------------------------------------------------#---------------------------------------------------#

def reusesave (): #This is a reuseable code function #

mx,my = pygame.mouse.get\_pos() #This tracks the x and y of the mouse #

screen.blit(icon3, (mx, my)) #This displays the cursor on the mouse #

pygame.display.flip() #This refreshes the screen #

clock.tick(1000000) #This is the refresh rate of game #

def reusesave2 (): #This is a reuseable code function #

screen.fill((30,30,30)) #This fills the screen with a specific colour #

pygame.draw.rect(screen,(79,173,81), [0, 0, 600, 160]) #This draws the top title bar on screen #

pygame.draw.rect(screen,((47,79,79)), [0, 400, 600, 10]) #This draws a diving line (thin rectangle) #

screen.blit(fontsize4.render("INSTRUCTIONS", True, (30,30,30)), [10, 30]) #This prints the text for the instructions #

screen.blit(fontsize4.render("INSTRUCTIONS", True, (255,255,255)), [15, 30]) #This prints the text for the instructions #

screen.blit(fontsize2.render("SAGAR PATEL - ICS3U1", True, (255,255,255)), [130, 750]) #This prints the text for the instructions #

screen.blit(fontsize3.render("Visit This Link", True, (255,255,255)), [57, 230]) #This prints the text for the instructions #

screen.blit(fontsize5.render("www.blackjack.org", True, (96,94,94)), [47, 330]) #This prints the text for the instructions #

mx,my = pygame.mouse.get\_pos() #This assigns mouse location to variable #

clicked = pygame.mouse.get\_pressed() #This assigns the mouse clicks to variable #

def reusesave3 (): #This is a reuseable code function #

pygame.display.flip() #This refreshes the screen #

clock.tick(1000000) #------------#This is the refresh rate of game #

def button (text,x,y,w,h,r1,g1,b1,r2,g2,b2,fontcolourr1,fontcolourg1,fontcolourb1,xhigh,ylow,font,action = None): #Button function #

mx,my = pygame.mouse.get\_pos() #------------#This tracks x and y mouse #

clicked = pygame.mouse.get\_pressed() #This tracks the clicks of the mouse #

if x+xhigh > mx > x and y+ylow > my > y: #This is essentially cursor collision #

pygame.draw.rect(screen,(r2,g2,b2),(x,y,xhigh,ylow)) #This draws a button on top of the original #

screen.blit(font.render(text, True, (255,255,255)), [w, h]) #This blits the text to the screen #

if clicked [0] == 1 and action != None: #If the button is clicked: #

#buttonSound.play () #Plays the button sound effect #

action () #Do the action that is declared in the button #

else: #Anything else: #

pygame.draw.rect(screen,(r1,g1,b1),(x,y,xhigh,ylow)) #Draw the original button #

menutext3 = font.render(text, True, (fontcolourr1,fontcolourg1,fontcolourb1,30)) #Draws the old text to the screen #

screen.blit(menutext3, [w, h]) #This blits the text ot the screen #

def reusesave7 (): #Reusable code function #

screen.fill((30,30,30)) #Fills the screen with a colour #

pygame.draw.rect(screen,(40,149,80), [0, 0, 1000, 400]) #Draws a rectangle to surface #

pygame.draw.rect(screen,(47,79,79), [800, 0, 1000, 800]) #Draws a rectangle to surface #

pygame.draw.rect(screen,(51,62,63), [600, 0, 200, 800]) #Draws a rectangle to surface #

pygame.draw.rect(screen,(0,0,0), [600, 399, 1000, 2]) #Draws a rectangle to surface #

pygame.draw.rect(screen,(40,40,40), [875, 70, 50, 315]) #Draws a rectangle to surface #

pygame.draw.rect(screen,(40,40,40), [875, 470, 50, 315]) #Draws a rectangle to surface #

#---------------------------------- INTRODUCTION -------------------------------------------------------#---------------------------------------------------#

def introduction (): #Beginning of the intro-screen function #

introduction = True #Loop variable is set to "True" #

while introduction: #Begins loop for intro screen #

screen = pygame.display.set\_mode((600, 800), pygame.NOFRAME) #Sets the screen size of the game #

for event in pygame.event.get(): #Gets the event type for the game #

if event.type == pygame.QUIT: #If the event type is quit: #

introduction = False #Variable will be set to false and game will quit #

screen.fill((30,30,30)) #This fills the screen with a grey colour #

pygame.draw.rect(screen,(79,173,81), [0, 0, 600, 180]) #This draws the title background section #

screen.blit(fontsize1.render("BLACKJACK", True, (30,30,30)), [20, 30]) #This blits text to the screen #

screen.blit(fontsize1.render("BLACKJACK", True, (255,255,255)), [25, 30]) #This blits text to the screen #

screen.blit(fontsize2.render("SAGAR PATEL - ICS3U1", True, (255,255,255)), [130, 750]) #This blits text to the screen #

mx,my = pygame.mouse.get\_pos() #This assigns mouse movement to a variable #

clicked = pygame.mouse.get\_pressed() #This assigns clicks to a variable #

button ("START",50,275,200,285,79,173,81,30,30,30,30,30,30,500,100,fontsize3,premaingame) #This blits a button to the screen #

button ("INSTRUCTIONS",50,425,80,435,79,173,81,30,30,30,30,30,30,500,100,fontsize3,instructions)#This blits a button to the screen #

button ("QUIT",50,575,230,585,79,173,81,30,30,30,30,30,30,500,100,fontsize3,quitgame) #This blits a button to the screen #

screen.blit(icon2, (70, 190)) #This blits the icon2 to the screen #

reusesave () #This calls the reusable code function #

#--------------------------------- GAME FUNCTIONS ------------------------------------------------------#---------------------------------------------------#

def pauseins (): #Pause Instruction Function #

countertest = 0 #Variable for slowing down buttons #

instructions = True #Variable for loop #

for x in range (400): #Loop that repeats 400 times #

screen = pygame.display.set\_mode((1000-x, 800), pygame.NOFRAME) #Sets the original size for the screen #

x = x + 1 #This adds one to the x axis of screen #

while instructions: #Loop for Pause Instructions #

for event in pygame.event.get(): #Gets event so you can exit loop #

if event.type == pygame.QUIT: #If the event type is quit... #

instructions = False #The loop will end #

reusesave2 () #Reusable code function is called #

if countertest >= 50: #If variable is greater than 50... #

button ("GO BACK",50,425,170,435,79,173,81,30,30,30,30,30,30,500,100,fontsize3,pausemenu) #Button will have a function #

button ("QUIT",50,575,230,585,79,173,81,30,30,30,30,30,30,500,100,fontsize3,quitgame) #Button will have a function #

else: #Else... #

button ("GO BACK",50,425,170,435,79,173,81,30,30,30,30,30,30,500,100,fontsize3,None) #utton will not have a function #

button ("QUIT",50,575,230,585,79,173,81,30,30,30,30,30,30,500,100,fontsize3,None) #utton will not have a function #

countertest = countertest + 1 #Add 1 to the slowing down variable #

reusesave () #Calls reusable code #

#----------------------------------- INSTRUCTIONS ------------------------------------------------------#---------------------------------------------------#

def instructions (): #Instructions Function #

countertest = 0 #Variable for slowing down buttons #

instructions = True #Variable to determine if loop is running #

while instructions: #Loop for Pause Instructions #

for event in pygame.event.get(): #Gets event so you can exit loop #

if event.type == pygame.QUIT: #If the event type is quit... #

instructions = False #The loop will end #

reusesave2 () #Reusable code function is called #

if countertest >= 50: #If variable is greater than 50... #

button ("GO BACK",50,425,170,435,79,173,81,30,30,30,30,30,30,500,100,fontsize3,introduction)#Button will have a function #

button ("QUIT",50,575,230,585,79,173,81,30,30,30,30,30,30,500,100,fontsize3,quitgame) #Button will have a function #

else: #Else... #

button ("GO BACK",50,425,170,435,79,173,81,30,30,30,30,30,30,500,100,fontsize3,None) #utton will not have a function #

button ("QUIT",50,575,230,585,79,173,81,30,30,30,30,30,30,500,100,fontsize3,None) #utton will not have a function #

countertest = countertest + 1 #Add 1 to the slowing down variable #

reusesave () #Calls reusable code #

#------------------------------------------- WIN -------------------------------------------------------#---------------------------------------------------#

def winner (): #Function if the player wins #

for x in range (400): #Loop that runs 400 times #

screen = pygame.display.set\_mode((1000-x, 800), pygame.NOFRAME) #Sets the screen size where x increases by 1 #

winner = True #Variable for loop #

exitloop2 = True #Variable for loop #

x = 0 #Variable to slow down box animation #

while exitloop2: #Loop #

for event in pygame.event.get(): #This gets the event type #

if event.type == pygame.QUIT: #If the event type is quit... #

exitloop2 = False #Variable is set to false and loop ends #

if x >= 200: #If variable is greater than 200... #

exitloop2 = False #Exit the loop because variable is false #

else: #Else... #

x = x + 1 #Add 1 to the x value #

pygame.draw.rect(screen,((26,146,75)), [0, 0, x\*3, 800]) #Draw a rectangle [increases in size by 1px] #

reusesave3 () #Calls reusable code for use #

#winSound.play () #Plays winning sound effect #

while winner: #Loop begins #

for event in pygame.event.get(): #This gets the event type #

if event.type == pygame.QUIT: #If the event type is quit... #

winner = False #Variable is assigned to false #

mx,my = pygame.mouse.get\_pos() #Assigns x and y of mouse to 2 variables #

clicked = pygame.mouse.get\_pressed() #Assigns all mouse presses onto variable #

screen.fill((26,146,75)) #Fills the screen with a colour #

pygame.draw.rect(screen,(255,255,255), [0, 0, 600, 160]) #Draws a rectangle onto screen #

button ("RETRY",50,425,200,435,0,0,0,26,146,75,26,146,75,500,100,fontsize3,introduction) #Button for game [Function: go back to introduction]#

button ("QUIT",50,575,230,585,0,0,0,26,146,75,26,146,75,500,100,fontsize3,quitgame) #Button for game [Function: quit tge game] #

screen.blit(icon2, (70, 280)) #Blits icon to screen [card suits for design] #

screen.blit(fontsize5.render("THANK YOU FOR PLAYING!", True, (0,0,0)), [20, 170]) #This blits text/icon to the surface #

screen.blit(fontsize1.render("WINNER!", True, (0,0,0)), [110, 20]) #This blits text/icon to the surface #

screen.blit(fontsize2.render("SAGAR PATEL - ICS3U1", True, (0,0,0)), [130, 750]) #This blits text/icon to the surface #

reusesave () #Calls reusable code for use #

#----------------------------------------- LOSE --------------------------------------------------------#---------------------------------------------------#

def loser (): #Function for if player loses #

for x in range (400): #Loop that runs 400 times #

screen = pygame.display.set\_mode((1000-x, 800), pygame.NOFRAME) #Screen size is set [increases x by 1 each time] # #

loser = True #Variable is set to True for loop use #

exitloop3 = True #Variable is set to True for loop use #

x = 0 #Variable is set to 0 [used for increasing screen x]#

while exitloop3: #While loop #

for event in pygame.event.get(): #Required to exit loop #

if event.type == pygame.QUIT: #If event is quit pygame... #

exitloop3 = False #Variable becomes false and loop ends #

if x >= 200: #If variable is greater or equal to 200... #

exitloop3 = False #Variable becomes false and loop ends #

else: #Else... #

x = x + 1 #Variable increases by 1 each time for screensize #

pygame.draw.rect(screen,((141,29,46)), [0, 0, x\*3, 800]) #Rectangle that increases in x is drawn #

reusesave3 () #Calls function to reuse code #

#loseSound.play () #Loser sound is played #

while loser: #While loop #

for event in pygame.event.get(): #Required to exit loop #

if event.type == pygame.QUIT: #If event is quit pygame... #

loser = False #Variable becomes False and loop ends #

mx,my = pygame.mouse.get\_pos() #Assigns x and y of mouse to 2 variables #

clicked = pygame.mouse.get\_pressed() #Assigns mouse presses to variable #

screen.fill((141,29,46)) #Sets the background screen colour #

pygame.draw.rect(screen,(255,255,255), [0, 0, 600, 160]) #Draws a white rectangle #

screen.blit(icon2, (70, 280)) #Blits icon2 to the screen/surface #

button ("RETRY",50,425,200,435,0,0,0,141,29,46,141,29,46,500,100,fontsize3,introduction) #This is a button that takes you to intro #

button ("QUIT",50,575,230,585,0,0,0,141,29,46,141,29,46,500,100,fontsize3,quitgame) #This is a button that quits the game #

screen.blit(fontsize5.render("THANK YOU FOR PLAYING!", True, (0,0,0)), [20, 170]) #This blits text/icon to the surface #

screen.blit(fontsize1.render("LOSER!", True, (0,0,0)), [140, 20]) #This blits text/icon to the surface #

screen.blit(fontsize2.render("SAGAR PATEL - ICS3U1", True, (0,0,0)), [130, 750]) #This blits text/icon to the surface #

reusesave () #Calls function to reuse code #

#---------------------------------------- TIE ----------------------------------------------------------#---------------------------------------------------#

def tie (): #Function if Player and Dealer Tie #

for x in range (400): #Loop that repeats 400 times #

screen = pygame.display.set\_mode((1000-x, 800), pygame.NOFRAME) #Sets the screen size where x decreases by one #

tie = True #Variable is set to True for loop #

exitloop4 = True #Variable is set to True for loop #

x = 0 #Variable to slow down animation is set to 0 #

while exitloop4: #While loop #

for event in pygame.event.get(): #Required to exit loop #

if event.type == pygame.QUIT: #If event is quit pygame... #

exitloop4 = False #Variable is False and loop ends #

if x >= 200: #If variable is greater or equal to 200... #

exitloop4 = False #Variable is false and loop ends #

else: #Else... #

x = x + 1 #Variable increases by 1 each time #

pygame.draw.rect(screen,((0,112,188)), [0, 0, x\*3, 800]) #Draws a growing rectangle (+= 1) #

reusesave3 () #Calls function to reuse code #

while tie: #Loop #

for event in pygame.event.get(): #Required to exit loop #

if event.type == pygame.QUIT: #If event is quit pygame... #

tie = False #Loop ends and variable becomes False #

mx,my = pygame.mouse.get\_pos() #Assigns x and y of mouse to 2 variables #

clicked = pygame.mouse.get\_pressed() #Assigns mouse presses to variable #

screen.fill((0,112,188)) #Fills the screen with a background colour #

pygame.draw.rect(screen,(255,255,255), [0, 0, 600, 160]) #Draws a rectangle onto the screen #

screen.blit(icon2, (70, 280)) #Blits icon to the screen #

button ("RETRY",50,425,200,435,0,0,0,0,112,188,0,112,188,500,100,fontsize3,introduction) #Button for game [Function: Go back to introduction]#

button ("QUIT",50,575,230,585,0,0,0,0,112,188,0,112,188,500,100,fontsize3,quitgame) #Button for game [Function: quit game] #

screen.blit(fontsize5.render("THANK YOU FOR PLAYING!", True, (0,0,0)), [20, 170]) #This blits text/icon to the surface #

screen.blit(fontsize1.render("TIE!", True, (0,0,0)), [210, 20]) #This blits text/icon to the surface #

screen.blit(fontsize2.render("SAGAR PATEL - ICS3U1", True, (0,0,0)), [130, 750]) #This blits text/icon to the surface #

reusesave () #Calls function to reuse code #

#----------------------------------- PAUSE/UNPAUSE MENU ------------------------------------------------#---------------------------------------------------#

def unpausemenu (): #Function for Unpause Menu #

pause = False #pause is set to False #

mainmainmaingame () #main game is called #

def pausemenu (): #Function for Pause Menu #

pause = True #Variable used for loop is set to True #

countertest2 = 0 #Variable to slow down button click is set to 0 #

while pause: #While loop #

screen = pygame.display.set\_mode((1000, 800), pygame.NOFRAME) #Screen size is set #

for event in pygame.event.get(): #Required to exit loop #

if event.type == pygame.QUIT: #If event is quit pygame... #

pause = False #Variable is false and loop ends #

mx,my = pygame.mouse.get\_pos() #Assigns x and y of mouse to 2 variables #

clicked = pygame.mouse.get\_pressed() #Assigns mouse presses to variable #

pygame.draw.rect(screen,(140,149,80), [0, 0, 1000, 400]) #Draws rectangle onto screen #

if countertest2 >= 50: #If variable is greater or equal to 50... #

button ("RETURN",0,0,330,85,79,173,81,30,30,30,30,30,30,1000,267,fontsize1,unpausemenu) #Button for game [Function: resume game] #

button ("INSTRUCTIONS",0,267,200,355,53,94,59,30,30,30,30,30,30,1000,267,fontsize1,pauseins) #Button for game [Function: calls pauseins function]#

button ("QUIT",0,533,410,635,0,50,50,30,30,30,30,30,30,1000,267,fontsize1,quitgame) #Button for game [Function: quits game] #

else: #Else... #

button ("RETURN",0,0,330,85,79,173,81,30,30,30,30,30,30,1000,267,fontsize1,None) #Button for game [Function: None] #

button ("INSTRUCTIONS",0,267,200,355,53,94,59,30,30,30,30,30,30,1000,267,fontsize1,None) #Button for game [Function: None] #

button ("QUIT",0,533,410,635,0,50,50,30,30,30,30,30,30,1000,267,fontsize1,None) #Button for game [Function: None] #

countertest2 = countertest2 + 1 #Variable increases by 1 #

reusesave () #Reusable code is called for use #

#-------------------------------------- RESULTS PAGE FUNCTION ------------------------------------------#---------------------------------------------------#

def results (): #Results Function #

counter90 = 0 #Variable is set to 0 for slowing down processes #

for x in range (400): #Loop that repeats 400 times #

screen = pygame.display.set\_mode((1000-x, 800), pygame.NOFRAME) #Screen size is set and decreases by 1 each time #

global totalplayervalue #Variable is global for use #

global totaldealervalue #Variable is global for use #

results = True #Variable is true to begin loop #

while results: #While loop #

for event in pygame.event.get(): #Required to exit loop #

if event.type == pygame.QUIT: #If event is quit pygame... #

results = False #Variable is false and loop ends #

mx,my = pygame.mouse.get\_pos() #Assigns x and y of mouse to 2 variables #

clicked = pygame.mouse.get\_pressed() #Assigns mouse presses to variable #

screen.fill((30,30,30)) #This fills the screen with a grey colour #

pygame.draw.rect(screen,(79,173,81), [0, 0, 600, 180]) #This draws the title background section #

screen.blit(fontsize1.render("RESULTS", True, (30,30,30)), [80, 30]) #This blits text/icon to the surface #

screen.blit(fontsize1.render("RESULTS", True, (255,255,255)), [85, 30]) #---#This blits text/icon to the surface #

screen.blit(fontsize3.render("PLAYER: " + str(totalplayervalue), True, (255,255,255)), [105, 300]) #This blits text/icon to the surface #

screen.blit(fontsize3.render("DEALER: " + str(totaldealervalue), True, (255,255,255)), [105, 500]) #This blits text/icon to the surface #

screen.blit(fontsize2.render("SAGAR PATEL - ICS3U1", True, (230,230,230)), [130, 750]) #---#This blits text/icon to the surface #

if counter90 >= 200: #If counter is greater or equal to 200... #

if totalplayervalue > totaldealervalue and totaldealervalue < 21: #If these requirements are met... #

winner () #winner function is called #

elif totalplayervalue == 21 and totaldealervalue < 21: #If these requirements are met... #

tie () #tie function is called #

elif totaldealervalue > 21 and totalplayervalue <= 21: #If these requirements are met... #

winner () #winner function is called #

elif totaldealervalue > totalplayervalue and totaldealervalue <= 21: #If these requirements are met... #

loser () #loser function is called #

elif totalplayervalue > totaldealervalue and totalplayervalue <= 21: #If these requirements are met... #

winner () #winner function is called #

elif totalplayervalue == totaldealervalue and totalplayervalue <= 21: #If these requirements are met... #

tie () #tie function is called #

else: #Else... #

counter90 = counter90 + 1 #Variable is increased by 1 #

reusesave () #Calls function to reuse code #

#----------------------------------- DEALER TURN FUNCTION ----------------------------------------------#---------------------------------------------------#

def dealer (): #Function to chose the dealers cards #

global totalplayervalue #Variable is global for use #

global totaldealervalue #Variable is global for use #

global seconddealercard #Variable is global for use #

global carddeck #Variable is global for use #

global playerdeck #Variable is global for use #

global dealerdeck #Variable is global for use #

global firstdealercard #Variable is global for use #

global firstplayercard #Variable is global for use #

global secondplayercard #Variable is global for use #

totaldealervalue = totaldealervalue + seconddealercard [1] #Second dealer card is now added to totaldealervalue#

counter9 = 0 #Variable to slow down processes in loop #

counter10 = 0 #Variable to slow down processes in loop #

counter80 = 0 #Variable to slow down processes in loop #

counter800 = 0 #Variable to slow down processes in loop #

counter70000 = 0 #Variable to slow down processes in loop #

counter10000000 = 0 #Variable to slow down processes in loop #

maingame = True #Variable is used for loop and is True #

while maingame: #While loop #

for event in pygame.event.get(): #Required to exit loop #

if event.type == pygame.QUIT: #If event is quit pygame... #

maingame = False #Variable is false and loop ends #

mx,my = pygame.mouse.get\_pos() #Assigns mouse position to 2 variables #

clicked = pygame.mouse.get\_pressed() #Assigns mouse presses to variable #

reusesave7 () #Calls function to reuse code #

if counter10 >= 200: #If variable is greater or equal to 200... #

dealerxvalue = 5 #Variable is assigned the value of 5 #

dealeryvalue = 405 #Variable is assigned the value of 405 #

lengthdeck = len(dealerdeck) #This process blits the cards with an algorithm #

dealerimagelist = [item[0] for item in dealerdeck] #This process blits the cards with an algorithm #

if lengthdeck >= 5: #This process blits the cards with an algorithm #

for x in range (0,5): #This process blits the cards with an algorithm #

screen.blit(dealerimagelist[x], [dealerxvalue, dealeryvalue]) #This process blits the cards with an algorithm #

dealerxvalue = dealerxvalue + 110 #This process blits the cards with an algorithm #

dealerxvalue = 5 #This process blits the cards with an algorithm #

for x in range (5,lengthdeck): #This process blits the cards with an algorithm #

dealeryvalue = 565 #This process blits the cards with an algorithm #

dealerxvalue = 5 #This process blits the cards with an algorithm #

screen.blit(dealerimagelist[x], [dealerxvalue, dealeryvalue]) #This process blits the cards with an algorithm #

dealerxvalue = dealerxvalue + 110 #This process blits the cards with an algorithm #

else: #Else... #

for x in range(len(dealerimagelist)): #This process blits the cards with an algorithm #

screen.blit(dealerimagelist[x], [dealerxvalue, dealeryvalue]) #This process blits the cards with an algorithm #

dealerxvalue = dealerxvalue + 110 #This process blits the cards with an algorithm #

counter10 = counter10 + 1 #Variable is increased by 1 #

else: #Else... #

if counter9 >= 50: #If counter is greater or equal to 50... #

screen.blit(firstdealercard[0],[5, 405]) #Blit the first dealer card to the screen #

screen.blit(seconddealercard[0], [115, 405]) #Blit the second dealer card to the screen #

else: #Else... #

screen.blit(firstdealercard[0],[5, 405]) #The first dealer card is blitted to the screen #

screen.blit(BACKSIDE, [115, 405]) #The backside of the card is blitted to the screen #

counter9 = counter9 + 1 #Variable is increased by 1 #

counter10 = counter10 + 1 #Variable is increased by 1 #

if totaldealervalue < 17 and counter10 >= 200: #If these requirements are met... #

dealercardselect = "HIT" #variable is set to "HIT" #

if counter10000000 >= 100: #If variable is greater or equal to 100... #

newdealercard = random.choice(carddeck) #A random card is selected from the deck #

dealerdeck.append(newdealercard) #The card is appended to the dealer deck #

carddeck.remove(newdealercard) #The card is removed from the card deck #

lengthdeck = len(dealerdeck) #This process blits the cards with an algorithm #

dealerxvalue = 5 #Variable is assigned the value of 5 #

dealeryvalue = 405 #Variable is assigned the value of 405 #

cardimagelist = [item[0] for item in dealerdeck] #This process blits the cards with an algorithm #

if lengthdeck >= 5: #This process blits the cards with an algorithm #

for x in range (0,5): #This process blits the cards with 1an algorithm #

screen.blit(cardimagelist[x], [dealerxvalue, dealeryvalue]) #This process blits the cards with an algorithm #

dealerxvalue = dealerxvalue + 110 #This process blits the cards with an algorithm #

dealerxvalue = 5 #This process blits the cards with an algorithm #

for x in range (5,lengthdeck): #This process blits the cards with an algorithm #

dealeryvalue = 565 #This process blits the cards with an algorithm #

screen.blit(cardimagelist[x], [dealerxvalue, dealeryvalue]) #This process blits the cards with an algorithm #

dealerxvalue = dealerxvalue + 110 #This process blits the cards with an algorithm #

else: #Else... #

for x in range(len(cardimagelist)): #This process blits the cards with an algorithm #

screen.blit(cardimagelist[x], [dealerxvalue, dealeryvalue]) #This process blits the cards with an algorithm #

dealerxvalue = dealerxvalue + 110 #This process blits the cards with an algorithm #

if newdealercard[1] == 0: #If the new dealer card 1st index is 0... #

totaldealervalue = totaldealervalue + 1 #Add one to the totaldealer value [selected] #

else: #Else... #

totaldealervalue = totaldealervalue + newdealercard[1] #The newdealercar is added to the totaldealervalue #

pygame.time.wait(400) #Time delay of 0.4 seconds #

counter10000000 = 0 #Variable is set to the value 0 for time delay #

else: #Else... #

counter10000000 = counter10000000 + 1 #Variable is increased by 1 #

pygame.time.wait(1) #Time is delayed by 0.001 seconds #

elif totaldealervalue >= 17 and totaldealervalue <= 21 and counter10 >= 200: #If these requirements are met... #

dealercardselect = "STAND" #Variable is set to "STAND" #

if totaldealervalue > 21: #If these requirements are met... #

if counter80 >=200: #If these requirements are met... #

pygame.draw.rect(screen,(202,52,51), [875, 470, 50, 21\*15]) #Dealer bar is filled to 21 and is coloured red #

dealercardselect = "BUST" #Blits text to screen #

results () #Results function is called #

else: #Else... #

pygame.draw.rect(screen,(202,52,51), [875, 470, 50, 21\*15]) #Dealer bar is filled to 21 and is coloured red #

counter80 = counter80 + 1 #Variable is increased by 1 #

reusesave3 () #This calls reusable code for use #

elif totaldealervalue == 21: #If these requirements are met... #

if counter800 >= 1000: #If variable is greater or equal to 1000... #

pygame.draw.rect(screen,(255,215,0), [875, 470, 50, 21\*15]) #Score bar is 21 and coloured yellow #

results () #Results function is called #

else: #Else... #

pygame.draw.rect(screen,(255,215,0), [875, 470, 50, 21\*15]) #Score bar is 21 and coloured yellow #

counter800 = counter800 + 1 #Variable is increased by 1 #

else: #Else... #

pygame.draw.rect(screen,(255,255,255), [875, 470, 50, totaldealervalue\*15]) #Score of dealer is blitted to the screen #

screen.blit(fontsize2.render("YOU - " + str(totalplayervalue), True, (0,0,0)), [836, 20]) #Text is blitted to the screen #

screen.blit(fontsize2.render("DEALER - " + str(totaldealervalue), True, (0,0,0)), [810, 420]) #Text is blitted to the screen #

lengthdeck = len(playerdeck) #This process blits the cards with an algorithm #

xvalue = 5 #Variable is assigned the value of 5 #

yvalue = 5 #Variable is assigned the value of 5 #

cardimagelist = [item[0] for item in playerdeck] #This process blits the cards with an algorithm #

if lengthdeck >= 5: #This process blits the cards with an algorithm #

for x in range (0,5): #This process blits the cards with an algorithm #

screen.blit(cardimagelist[x], [xvalue, yvalue]) #This process blits the cards with an algorithm #

xvalue = xvalue + 110 #This process blits the cards with an algorithm #

xvalue = 5 #This process blits the cards with an algorithm #

for x in range (5,lengthdeck): #This process blits the cards with an algorithm #

yvalue = 165 #This process blits the cards with an algorithm #

screen.blit(cardimagelist[x], [xvalue, yvalue]) #This process blits the cards with an algorithm #

xvalue = xvalue + 110 #This process blits the cards with an algorithm #

else: #Else... #

for x in range(len(cardimagelist)): #This process blits the cards with an algorithm #

screen.blit(cardimagelist[x], [xvalue, yvalue]) #This process blits the cards with an algorithm #

xvalue = xvalue + 110 #Variable is increased by 110 #

screen.blit(fontsize300.render("DEALER SELECTS:", True, (255,255,255)), [603, 420]) #This blits text to the screen #

if counter10 >= 200: #If variable is greater or equal to 200... #

if dealercardselect == "HIT": #If variable is equal to "HIT"... #

screen.blit(fontsize3.render(dealercardselect, True, (144,238,144)), [651, 480]) #This blits text to the screen #

else: #Else... #

if counter70000 >= 400: #If variable is greater or equal to 400... #

screen.blit(fontsize3.render(dealercardselect, True, (144,238,144)), [601, 480]) #This blits text to the screen #

results () #Results function is called #

elif counter70000 >= 100: #Other condition #

screen.blit(fontsize3.render(dealercardselect, True, (144,238,144)), [601, 480]) #Print dealer option #

counter70000 = counter70000 + 1 #Variable increases by 1 #

else: #Else... #

counter70000 = counter70000 + 1 #Add 1 to variable #

if totalplayervalue == 21: #If the totalplayervalue is 21... #

pygame.draw.rect(screen,(255,215,0), [875, 70, 50, 21\*15]) #This draws a yellow bar for the player at 21 value #

else: #Else... #

pygame.draw.rect(screen,(255,255,255), [875, 70, 50, totalplayervalue\*15]) #Draws a player bar with totalplayervalue #

reusesave3 () #Calls reusable code function for use #

#-------------------------------------- MAIN GAME FOR PLAYER -------------------------------------------#---------------------------------------------------#

def mainmainmaingame (): #Main function for player card turn #

global totalplayervalue #Variable is global for use #

global totaldealervalue #Variable is global for use #

global carddeck #Variable is global for use #

global playerdeck #Variable is global for use #

global dealerdeck #Variable is global for use #

global firstdealercard #Variable is global for use #

global seconddealercard #Variable is global for use #

global firstplayercard #Variable is global for use #

global secondplayercard #Variable is global for use #

counter7 = 0 #Variable to slow down some processes of game #

counter8 = 0 #Variable to slow down some processes of game #

maingame = True #Variable to determine if loop is runnning #

while maingame: #Loop #

for event in pygame.event.get(): #Gets events for loop so player can quit #

if event.type == pygame.QUIT: #If event type is quit... #

maingame = False #Variable becomes false and exits loop #

mx,my = pygame.mouse.get\_pos() #Assigns x and y of mouse to 2 variables #

clicked = pygame.mouse.get\_pressed() #Assigns mouse presses to variable #

reusesave7 () #Calls function to reuse code #

pygame.draw.rect(screen,(255,255,255), [875, 470, 50, totaldealervalue\*15]) #This draws a dealerscore bar #

button ("",960,755,230,585,255,255,255,30,30,30,255,255,255,30,35,fontsize3,pausemenu) #Button for game [Function: calls pause function] #

button ("",970,755,230,585,47,79,79,47,79,79,255,255,255,10,35,fontsize3,None) #Button for game [Function: None] #

screen.blit(firstdealercard[0],[5, 405]) #Blits the first dealer card to screen #

screen.blit(BACKSIDE, [115, 405]) #Blits the backside of the dealer card to screen #

screen.blit(icon4, [580, 505]) #Blits icon of dealer's face in dealer's section #

button ("HIT",622,10,675,25,79,173,81,30,30,30,30,30,30,155,70,fontsize2,None) #-----------#Button for game [Function: None] #

if clicked [0] == 1 and 622+155 > mx > 622 and 10+70 > my > 10 and totalplayervalue <= 21 and counter8 <= 0 and counter7 <= 0:#requirements are met #

buttonSound.play () #-----------#Plays button sound effect #

newplayercard = random.choice(carddeck) #Picks a random card from the card deck #

playerdeck.append(newplayercard) #This appends a card into the player deck #

carddeck.remove(newplayercard) #This removes the card from the card deck #

if newplayercard[1] == 0: #If the 1st index in the new card is 0... #

acedrawn = True #Variable used to enter loop #

while acedrawn: #While loop #

for event in pygame.event.get(): #This is required to exit the loop #

if event.type == pygame.QUIT: #If event type is quit... #

acedrawn = False #Variable becomes false and loop ends #

mx,my = pygame.mouse.get\_pos() #Assigns x and y of mouse to 2 variables #

clicked = pygame.mouse.get\_pressed() #Assigns mouse presses to variable #

reusesave7 () #Calls function to reuse code #

pygame.draw.rect(screen,(255,255,255), [875, 470, 50, totaldealervalue\*15]) #--#Draws a rectangle for dealer bar #

button ("",960,755,230,585,255,255,255,30,30,30,255,255,255,30,35,fontsize3,pausemenu) #This button is a pause button [pause menu] #

button ("",970,755,230,585,47,79,79,47,79,79,255,255,255,10,35,fontsize3,None) #--#This button has no function, used for design #

screen.blit(firstdealercard[0],[5, 405]) #This blits the first dealer card to the screen #

screen.blit(BACKSIDE, [115, 405]) #This blits an image to the screen #

screen.blit(icon4, [580, 505]) #--#This blits and image to the screen #

screen.blit(fontsize200.render("YOU DREW AN ACE!", True, (255,255,255)), [612, 20]) #This blits text/icon to the surface #

screen.blit(fontsize200.render("CHOOSE ITS VALUE:", True, (255,255,255)), [615, 50]) #This blits text/icon to the surface #

button ("1",622,110+70,692,125+70,0,166,140,30,30,30,30,30,30,155,70,fontsize2,None) #Button for game [Function: None] #

button ("11",622,195+70,685,210+70,0,166,140,30,30,30,30,30,30,155,70,fontsize2,None) #Button for game [Function: None] #

if 622+155 > mx > 622 and 110+70+70 > my > 110+70 and clicked [0] == 1: #--#If these conditions are met... #

buttonSound.play () #Plays button sound effect #

firstacevalue = 1 #Variable value becomes 1 #

totalplayervalue = totalplayervalue + firstacevalue #--------#1 is added to total player value #

screen.blit(fontsize2.render("YOU - " + str(totalplayervalue), True, (0,0,0)), [836, 20])#This blits text/icon to the surface #

if totalplayervalue > 21: #If total player value is above 21... #

pygame.draw.rect(screen,(202,52,51), [875, 70, 50, 21\*15]) #Draws a filled player score bar with red #

elif totalplayervalue == 21: #If the totalplayer value is 21... #

pygame.draw.rect(screen,(255,215,0), [875, 70, 50, 21\*15]) #Draws a filled player score bar with yellow #

else: #Else... #

pygame.draw.rect(screen,(255,255,255), [875, 70, 50, totalplayervalue\*15]) #Draws a white bar with totalplayer value score #

break #This exits from loop #

elif 622+155 > mx > 622 and 195+70+70 > my > 195+70 and clicked [0] == 1: #If the requirements are met... #

buttonSound.play () #Plays button sound effect #

firstacevalue = 11 #Variable is set to "11" #

totalplayervalue = totalplayervalue + firstacevalue #--------#Firstacevalue is added to totaplayervalue #

screen.blit(fontsize2.render("YOU - " + str(totalplayervalue), True, (0,0,0)), [836, 20])#This blits text/icon to the surface #

if totalplayervalue > 21: #If variable is greater than 21... #

pygame.draw.rect(screen,(202,52,51), [875, 70, 50, 21\*15]) #Draws a player score bar at 21 thats red #

elif totalplayervalue == 21: #If variable is equal to 21... #

pygame.draw.rect(screen,(255,215,0), [875, 70, 50, 21\*15]) #Draws a player score bar at 21 thats yellow #

else: #Else... #

pygame.draw.rect(screen,(255,255,255), [875, 70, 50, totalplayervalue\*15]) #Exit loop #

break #---------#This exits from loop #

screen.blit(fontsize2.render("YOU - " + str(totalplayervalue), True, (0,0,0)), [836, 20]) #This blits text/icon to the surface #

screen.blit(fontsize2.render("DEALER - " + str(totaldealervalue), True, (0,0,0)), [810, 420]) #This blits text/icon to the surface #

if totalplayervalue > 21: #---------#If variable is greater than 21... #

pygame.draw.rect(screen,(202,52,51), [875, 70, 50, 21\*15]) #Draws player bar that red and is 21 #

elif totalplayervalue == 21: #If variable is equal to 21... #

pygame.draw.rect(screen,(255,215,0), [875, 70, 50, 21\*15]) #Draws a player score bar at 21 thats yellow #

else: #Else... #

pygame.draw.rect(screen,(255,255,255), [875, 70, 50, totalplayervalue\*15]) #Draw the player's total score bar in white #

xvalue = 5 #Variable is assigned to 5 #

yvalue = 5 #Variable is assigned to 5 #

lengthdeck = len(playerdeck) #Variable is assigned the # of card in playerdeck #

cardimagelist = [item[0] for item in playerdeck] #Variable is assigned the first index in each list #

if lengthdeck >= 5: #If the variable is greater or equal to 5: #

for x in range (0,5): #This process blits the cards with an algorithm #

screen.blit(cardimagelist[x], [xvalue, yvalue]) #This process blits the cards with an algorithm #

xvalue = xvalue + 110 #This process blits the cards with an algorithm #

xvalue = 5 #This process blits the cards with an algorithm #

for x in range (5,lengthdeck): #This process blits the cards with an algorithm #

yvalue = 165 #This process blits the cards with an algorithm #

screen.blit(cardimagelist[x], [xvalue, yvalue]) #This process blits the cards with an algorithm #

xvalue = xvalue + 110 #This process blits the cards with an algorithm #

else: #Else... #

for x in range(len(cardimagelist)): #For every card in the players card #

screen.blit(cardimagelist[x], [xvalue, yvalue]) #Blit each card with x value changing by 110 #

xvalue = xvalue + 110 #variable for x value increases by 110 #

else: #Else... #

totalplayervalue = totalplayervalue + newplayercard[1] #New selected card is added to the totalplayervalue #

time.sleep (0.5) #This causes the game to wait 0.5 seconds #

xvalue = 5 #Variable is assigned value of 5 #

yvalue = 5 #Variable is assigned value of 5 #

lengthdeck = len(playerdeck) #This process blits the cards with an algorithm #

cardimagelist = [item[0] for item in playerdeck] #This process blits the cards with an algorithm #

if lengthdeck >= 5: #This process blits the cards with an algorithm #

for x in range (0,5): #This process blits the cards with an algorithm #

screen.blit(cardimagelist[x], [xvalue, yvalue]) #This process blits the cards with an algorithm #

xvalue = xvalue + 110 #Variable increases by 110 #

xvalue = 5 #Variable is reset to the beginning x of the row #

for x in range (5,lengthdeck): #For every card from 5 - # in player's deck... #

yvalue = 165 #The y value for the cards are increased #

screen.blit(cardimagelist[x], [xvalue, yvalue]) #Each card after index 5 is blitted to second row #

xvalue = xvalue + 110 #Variable is increased by 110 #

else: #Else... #

for x in range(len(cardimagelist)): #For however many cards there are in playerdeck #

screen.blit(cardimagelist[x], [xvalue, yvalue]) #Each card in the list is blitted to the screen #

xvalue = xvalue + 110 #Variable is increased by 110 #

if totalplayervalue > 21: #If variable is greater or equal to 21... #

if counter7 >= 600: #If variable is greater or equal to 600... #

screen.blit(fontsize3.render("BUST!", True, (255,255,255)), [606, 220]) #This blits text/icon to the surface #

pygame.draw.rect(screen,(202,52,51), [875, 70, 50, 21\*15]) #This draws a playerscore bar #

loser () #Calls the loser function #

else: # Else... #

screen.blit(fontsize3.render("BUST!", True, (255,255,255)), [606, 220]) #This blits text/icon to the surface #

pygame.draw.rect(screen,(202,52,51), [875, 70, 50, 21\*15]) #This draws a rectangle on the screen #

counter7 = counter7 + 10 #Variable increases by 10 #

elif totalplayervalue == 21: #If totalplayervalue is equal to 21... #

if counter8 >= 2000: #If variable is greater or equal to 2000... #

pygame.draw.rect(screen,(255,215,0), [875, 70, 50, 21\*15]) #This draws a playerscore bar #

screen.blit(fontsize2.render("BLACKJACK!", True, (255,255,255)), [604, 200]) #This blits text/icon to the surface #

screen.blit(fontsize200.render("DEALER'S TURN!", True, (252,252,252)), [626, 250]) #This blits text/icon to the surface #

dealer () #Dealer function is called #

else: #Else... #

pygame.draw.rect(screen,(255,215,0), [875, 70, 50, 21\*15]) #This draws a playerscore bar #

screen.blit(fontsize2.render("BLACKJACK!", True, (255,255,255)), [604, 200]) #This blits text/icon to the surface #

screen.blit(fontsize200.render("DEALER'S TURN!", True, (252,252,252)), [626, 250]) #This blits text/icon to the surface #

counter8 = counter8 + 10 #Variable increases by 10 #

else: #Else... #

pygame.draw.rect(screen,(255,255,255), [875, 70, 50, totalplayervalue\*15]) #This draws a playerscore bar #

screen.blit(fontsize2.render("YOU - " + str(totalplayervalue), True, (0,0,0)), [836, 20]) #This blits text/icon to the surface #

screen.blit(fontsize2.render("DEALER - " + str(totaldealervalue), True, (0,0,0)), [810, 420]) #This blits text/icon to the surface #

if totalplayervalue <= 21 and counter8 <= 0 and counter7 <= 0: #If requirements are met... #

button ("STAND",622,95,650,110,79,173,81,30,30,30,30,30,30,155,70,fontsize2,dealer) #This is a button for display (no function) #

else: #Else... #

button ("STAND",622,95,650,110,79,173,81,30,30,30,30,30,30,155,70,fontsize2,None) #This is a button that calls dealer function #

reusesave () #Calls function to resuse code #

def reusesave9 (): #Another reuse code function to save code #

pygame.draw.rect(screen,(255,255,255), [875, 70, 50, totalplayervalue\*15]) #This draws a playerscore bar #

pygame.draw.rect(screen,(255,255,255), [875, 470, 50, totaldealervalue\*15]) #This draws a dealerscore bar #

screen.blit(fontsize2.render("YOU - " + str(totalplayervalue), True, (0,0,0)), [836, 20]) #This blits text/icon to the surface #

screen.blit(fontsize2.render("DEALER - " + str(totaldealervalue), True, (0,0,0)), [810, 420]) #This blits text/icon to the surface #

screen.blit(BACKSIDE, [115, 405]) #Backside of 2nd dealer card is blitted #

screen.blit(firstdealercard[0],[5, 405]) #First dealer card is blitted to the screen #

screen.blit(firstplayercard[0], [5, 5]) #First player card is blitted to the screen #

screen.blit(icon4, [580, 505]) #Icon is blitted to screen #

screen.blit(fontsize200.render("YOU DREW AN ACE!", True, (255,255,255)), [612, 50]) #This blits text/icon to the surface #

screen.blit(fontsize200.render("CHOOSE ITS VALUE:", True, (255,255,255)), [615, 80]) #This blits text/icon to the surface #

button ("1",622,110+70,692,125+70,0,166,140,30,30,30,30,30,30,155,70,fontsize2,None) #This is a button for display (no function) #

button ("11",622,195+70,685,210+70,0,166,140,30,30,30,30,30,30,155,70,fontsize2,None) #This is a button for display (no function) #

def maingame (): #Main game function for card selection #

firstmaingame = True #Variable for loop is assigned True #

global totalplayervalue #Variable is global for use #

global totaldealervalue #Variable is global for use #

global carddeck #Variable is global for use #

global playerdeck #Variable is global for use #

global dealerdeck #Variable is global for use #

global firstdealercard #Variable is global for use #

global seconddealercard #Variable is global for use #

global firstplayercard #Variable is global for use #

global secondplayercard #Variable is global for use #

totaldealervalue = 0 #Resets variables for use #

totalplayervalue = 0 #------------------------#Resets variables for use #

carddeck = [[cardAC,0],[cardAD,0],[cardAH,0],[cardAS,0],[card2C,2],[card2D,2],[card2H,2],[card2S,2],[card3C,3],[card3D,3], #List of cards #

[card3H,3],[card3S,3],[card4C,4],[card4D,4],[card4H,4],[card4S,4],[card5C,5],[card5D,5],[card5H,5],[card5S,5],[cardKC,10], #List of cards #

[card6C,6],[card6D,6],[card6H,6],[card6S,6],[card7C,7],[card7D,7],[card7H,7],[card7S,7],[card8C,8],[card8D,8],[cardKD,10], #List of cards #

[card8H,8],[card8S,8],[card9C,9],[card9D,9],[card9H,9],[card9S,9],[card10C,10],[card10D,10],[card10H,10],[cardKH,10], #List of cards #

[card10S,10],[cardJC,10],[cardJD,10],[cardJH,10],[cardJS,10],[cardQC,10],[cardQD,10],[cardQH,10],[cardQS,10],[cardKS,10]] #List of cards #

playerdeck = [] #------------------------#List of player's cards #

dealerdeck = [] #List of dealer's cards #

global countertest4 #Variable is global for use #

countertest4 = 0 #Variable to slow down process #

while firstmaingame: #While loop begins #

for event in pygame.event.get(): #Required to exit loop #

if event.type == pygame.QUIT: #If event is quit pygame... #

firstmaingame = False #Variable becomes false and loop ends #

if countertest4 <= 0: #If countertest4 is less or equal to 0... #

firstdealercard = random.choice(carddeck) #Variable is assigned a card from deck #

dealerdeck.append(firstdealercard) #Card is appended to the dealer's deck #

carddeck.remove(firstdealercard) #Card is removed from the card deck #

seconddealercard = random.choice(carddeck) #Variable is assigned a card from deck #

dealerdeck.append(seconddealercard) #Card is appended to the dealer's deck #

carddeck.remove(seconddealercard) #Card is removed from the card deck #

if seconddealercard[1] == 0: #If the seconddealercard is 0.... #

seconddealercard.insert(1,1) #1 is inserted to the card value #

if firstdealercard[1] == 0: #If the seconddealercard is 0.... #

totaldealervalue = totaldealervalue + 11 #11 is added to totaldealervalue #

else: #Else... #

totaldealervalue = totaldealervalue + firstdealercard[1] #First dealer card is added to total dealer value #

countertest4 = countertest4 + 10 #Variable is increased by 10 #

screen.blit(firstdealercard[0],[5, 405]) #The first dealer card is blitted to screen #

firstplayercard = random.choice(carddeck) #A random card is selected from deck #

playerdeck.append(firstplayercard) #The card is added to the player deck #

carddeck.remove(firstplayercard) #The card is removed from the deck of cards #

screen.blit(firstplayercard[0], [5, 5]) #Blit the firstplayer card to the screen #

if firstplayercard[1] != 0: #If the 1st index of firstplayercar is 0... #

totalplayervalue = totalplayervalue + firstplayercard[1] #Firstplayer card is added to totalplayervalue #

else: #Else... #

global firstacevalue #Variable is global for use #

firstacevalue = 0 #Variable is declared before use #

while firstmaingame: #While loop #

for event in pygame.event.get(): #Required to exit loop #

if event.type == pygame.QUIT: #If event is quit pygame... #

firstmaingame = False #Variable is false and loop ends #

mx,my = pygame.mouse.get\_pos() #Assigns x and y of mouse to 2 variables #

clicked = pygame.mouse.get\_pressed() #Assigns mouse presses to variable #

reusesave7 () #Calls function to reuse code #

reusesave9 () #Calls function to reuse code #

if 622+155 > mx > 622 and 110+70+70 > my > 110+70 and clicked [0] == 1: #If these requirements are met... #

buttonSound.play () #Plays button sound effect #

firstacevalue = 1 #firstacevalue is assigned the value "1" #

totalplayervalue = totalplayervalue + firstacevalue #firstacevalue is added to totalplayervalue #

pygame.draw.rect(screen,(255,255,255), [875, 70, 50, totalplayervalue\*15]) #----#This draws a playerscore bar #

screen.blit(fontsize2.render("YOU - " + str(totalplayervalue), True, (0,0,0)), [836, 20])#This blits text/icon to the surface #

reusesave () #----# #

break #This exits from loop #

elif 622+155 > mx > 622 and 195+70+70 > my > 195+70 and clicked [0] == 1: #If these requirements are met... #

buttonSound.play () #Plays button sound effect #

firstacevalue = 11 #firstacevalue is assigned the value "11" #

totalplayervalue = totalplayervalue + firstacevalue #The firstacevalue is added to totalplayervalue #

pygame.draw.rect(screen,(255,255,255), [875, 70, 50, totalplayervalue\*15]) #----#This draws a playerscore bar #

screen.blit(fontsize2.render("YOU - " + str(totalplayervalue), True, (0,0,0)), [836, 20])#This blits text/icon to the surface #

reusesave () #----#Reusable code is called for use #

break #This exits from loop #

reusesave () #Calls function to reuse code #

pygame.time.wait(400) #This causes the game to wait 0.4 seconds #

secondplayercard = random.choice(carddeck) #This selects a random card from carddeck #

playerdeck.append(secondplayercard) #The appends the card to the player's hand #

carddeck.remove(secondplayercard) #This removes the card from the deck #

screen.blit(secondplayercard[0], [115, 5]) #This blits the second player card to the screen #

reusesave () #Reusable code function is called for use #

if secondplayercard[1] != 0: #If the value of the second player card is not 0... #

totalplayervalue = totalplayervalue + secondplayercard[1] #The secondcard is added to totalplayervalue #

break #Exit loop #

else: #Else... #

global secondacevalue #Variable is global for use #

secondacevalue = 0 #Variable is declared for later use #

while firstmaingame: #While loop begins #

for event in pygame.event.get(): #Required to exit loop #

if event.type == pygame.QUIT: #If event is quit pygame... #

firstmaingame = False #Variable becomes false and loop ends #

mx,my = pygame.mouse.get\_pos() #Assigns x and y of mouse to 2 variables #

clicked = pygame.mouse.get\_pressed() #Assigns mouse presses to variable #

reusesave7 () #Calls function to reuse code #

reusesave9 () #Calls function to reuse code #

if 622+155 > mx > 622 and 110+70+70 > my > 110+70 and clicked [0] == 1: #If requirements for button are met... #

buttonSound.play () #Plays button sound effect #

firstacevalue = 1 #Player selected the Button "1" #

totalplayervalue = totalplayervalue + firstacevalue #Adds the selected ace value to the bar #

pygame.draw.rect(screen,(255,255,255), [875, 70, 50, totalplayervalue\*15]) #----#Draws a rectangle for player score #

screen.blit(fontsize2.render("YOU - " + str(totalplayervalue), True, (0,0,0)), [836, 20])#This blits text/icon to the surface #

reusesave () #----#Calls function to reuse code #

break #Break from loop #

elif 622+155 > mx > 622 and 195+70+70 > my > 195+70 and clicked [0] == 1: #If requirements for button are met... #

buttonSound.play () #Plays button sound effect #

firstacevalue = 11 #Assigns variable to selected value by player #

totalplayervalue = totalplayervalue + firstacevalue #Adds the selected ace value to player value #

pygame.draw.rect(screen,(255,255,255), [875, 70, 50, totalplayervalue\*15]) #----#Draws the bar for the player score #

screen.blit(fontsize2.render("YOU - " + str(totalplayervalue), True, (0,0,0)), [836, 20])#This blits text/icon to the surface #

reusesave () #----#Calls function to reuse code #

break #Breaks from the loop #

reusesave () #Calls reusable code for use #

break #Breaks from the loop #

mainmainmaingame () #This calls the "actual game" for player #

#-------------------------------------- FUNCTION FOR ANIMATION -----------------------------------------#---------------------------------------------------#

def premaingame (): #Function for before the game starts #

for x in range (400): #Loop that runs 400 times #

screen = pygame.display.set\_mode((600+x, 800), pygame.NOFRAME) #Sets the screen size, increases by 1 each run #

exitloop4 = True #Variable for exiting loop #

x = 0 #Variable for graphical animation #

while exitloop4: #Loop for premainmenu begins #

for event in pygame.event.get(): #Required to exit loop #

if event.type == pygame.QUIT: #If event is quit pygame... #

exitloop4 = False #Loop will end #

if x >= 200: #If the increasing variable is 200... #

exitloop4 = False #Loop will end #

else: #Else... #

x = x + 1 #Increases the variable by 1 to change object shape #

pygame.draw.rect(screen,((40,149,80)), [0, 0, x\*3, 400]) #Draws a growing rectangle #

pygame.draw.rect(screen,((30,30,30)), [0, 400, x\*3, 800]) #Draws a growing rectangle #

pygame.draw.rect(screen,(47,79,79), [800, 0, 200, 4\*x]) #Draws a growing rectangle #

pygame.draw.rect(screen,(51,62,63), [600, 0, 200, 4\*x]) #Draws a growing rectangle #

reusesave3 () #Calls the reusable code for use #

maingame () #Calls the actual game menu for card selection #

#--------------------------------- MAIN GAME -----------------------------------------------------------#---------------------------------------------------#

introduction() #This calls the function for the game #

#--------------------------------- END OF GAME ---------------------------------------------------------#---------------------------------------------------#