LUDO GAME PROJECT

PROGRAMMING FOR DATA SCIENCE

By

(Aryan Suri 21CSU467, Garvit 21CSU449, Swaraj 21CSU451, Devansh 21CSU442, Pratistha 21CSU465)

Under the supervision of

Ms. Amandeep Kaur

Department of Computer Science and Engineering

School of Engineering and Technology

The NorthCap University

HUDA, Sec-23A, Gurugram, Haryana - 122017

Table of Contents

PROJECT DESCRIPTION

Ludo is a strategy board game for two to four players, in which the players race their four tokens from start to finish according to the rolls of a single die. Like other cross and circle games, Ludo is derived from the Indian game Pachisi. The game and its variations are popular in many countries and under various names.

Ludo in python was made using modules like pygame, numpy, sys, random and many other modules.

Pygame is a module that helps in making 2d games using python using object oriented program and data analysis.

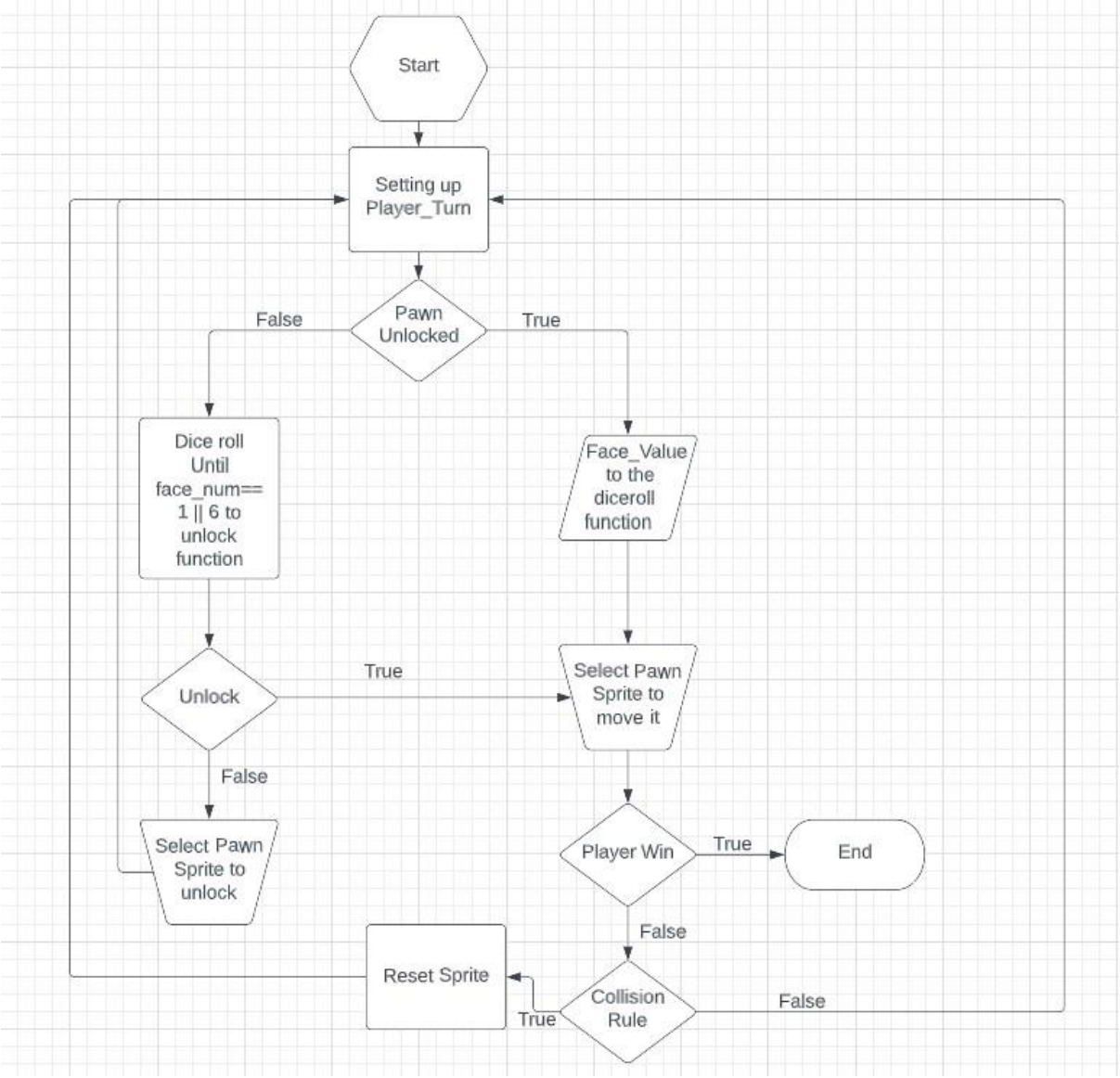
ANALYSIS

HARDWARE REQUIREMENT: Laptop, Desktop, RAM 256 MB or above, Integrated graphics or above

SOFTWARE REQUIREMENT: Python along with pygame installed, Command Prompt or any text editor to execute the program, OS Linux/Windows/Mac

DESIGN

Module was deeply studied and important functions and attributes were separated to implement in the program.
Detailed design of the program can be seen in the code provided below and output screenshots and flowchart was made to make a staple process of the program and make implementation easy



IMPLEMENTATION

main.py→ Program starts from the main file and executes the sprites from the interface.py file so that we can have a game board set and we can start playing the game. The file contains the diceroll_step which helps in moving the sprites around the board

```
def diceroll_step(number, turn):
    if turn == "player2":
        while interface.click:
            a = interface.player2_a
            b = interface.player2_b
            c = interface.player2_c
            d = interface.player2_d
            if a.unlock is False and b.unlock is False and c.unlock is False and d.unlock is False:
                interface.click = False
            else:
                ev = pygame.event.get()
                for event in ev:
                    if event.type == pygame.MOUSEBUTTONUP:
                        pos = pygame.mouse.get_pos()
                        if pos[0]>=a.x and pos[0]<=a.x+39 and pos[1]>=a.y and pos[1]<=a.y+64 and a.unlock is True:
                            a.map_yellow(number)

                            interface.click = False
                        elif pos[0]>=b.x and pos[0]<=b.x+39 and pos[1]>=b.y and pos[1]<=b.y+64 and b.unlock is True:
                            b.map_yellow(number)

                            interface.click = False
                        elif pos[0]>=c.x and pos[0]<=c.x+39 and pos[1]>=c.y and pos[1]<=c.y+64 and c.unlock is True:
                            c.map_yellow(number)

                            interface.click = False
                        elif pos[0]>=d.x and pos[0]<=d.x+39 and pos[1]>=d.y and pos[1]<=d.y+64 and d.unlock is True:
                            d.map_yellow(number)

                            interface.click = False
                        elif a.unlock is False and b.unlock is False and c.unlock is False and d.unlock is False:
                            interface.click = False
                        else:
                            interface.click = True
                    elif event.type == pygame.QUIT:
                        sys.exit()
```

```
elif turn=="player1":
    while interface.click:
        a = interface.player1_a
        b = interface.player1_b
        c = interface.player1_c
        d = interface.player1_d
        if a.unlock is False and b.unlock is False and c.unlock is False and d.unlock is False:
            interface.click = False
        else:
            ev = pygame.event.get()
            for event in ev:
                if event.type == pygame.MOUSEBUTTONUP:
                    pos = pygame.mouse.get_pos()
                    if pos[0]>=a.x and pos[0]<=a.x+39 and pos[1]>=a.y and pos[1]<=a.y+64 and a.unlock is True:
                        a.map_blue(number)

                        interface.click = False
                    elif pos[0]>=b.x and pos[0]<=b.x+39 and pos[1]>=b.y and pos[1]<=b.y+64 and b.unlock is True:
                        b.map_blue(number)

                        interface.click = False
                    elif pos[0]>=c.x and pos[0]<=c.x+39 and pos[1]>=c.y and pos[1]<=c.y+64 and c.unlock is True:
                        c.map_blue(number)

                        interface.click = False
                    elif pos[0]>=d.x and pos[0]<=d.x+39 and pos[1]>=d.y and pos[1]<=d.y+64 and d.unlock is True:
                        d.map_blue(number)

                        interface.click = False
                    elif a.unlock is False and b.unlock is False and c.unlock is False and d.unlock is False:
                        interface.click = False
                    else:
                        interface.click = True
                elif event.type == pygame.QUIT:
                    sys.exit()
```

```
elif turn=="player3":
    while interface.click:
        a = interface.player4_a
        b = interface.player4_b
        c = interface.player4_c
        d = interface.player4_d
        if a.unlock is False and b.unlock is False and c.unlock is False and d.unlock is False:
            interface.click = False
        else:
            ev = pygame.event.get()
            for event in ev:
                if event.type == pygame.MOUSEBUTTONUP:
                    pos = pygame.mouse.get_pos()
                    if pos[0]>=a.x and pos[0]<=a.x+39 and pos[1]>=a.y and pos[1]<=a.y+64 and a.unlock is True:
                        a.map_green(number)

                        interface.click = False
                    elif pos[0]>=b.x and pos[0]<=b.x+39 and pos[1]>=b.y and pos[1]<=b.y+64 and b.unlock is True:
                        b.map_green(number)

                        interface.click = False
                    elif pos[0]>=c.x and pos[0]<=c.x+39 and pos[1]>=c.y and pos[1]<=c.y+64 and c.unlock is True:
                        c.map_green(number)

                        interface.click = False
                    elif pos[0]>=d.x and pos[0]<=d.x+39 and pos[1]>=d.y and pos[1]<=d.y+64 and d.unlock is True:
                        d.map_green(number)

                        interface.click = False
                    elif a.unlock is False and b.unlock is False and c.unlock is False and d.unlock is False:
                        interface.click = False
                    else:
                        interface.click = True
                elif event.type == pygame.QUIT:
                    sys.exit()
```

```
elif turn=="player4":
    while interface.click:
        a = interface.player3_a
        b = interface.player3_b
        c = interface.player3_c
        d = interface.player3_d
        if a.unlock is False and b.unlock is False and c.unlock is False and d.unlock is False:
            interface.click = False
        else:
            ev = pygame.event.get()
            for event in ev:
                if event.type == pygame.MOUSEBUTTONUP:
                    pos = pygame.mouse.get_pos()
                    if pos[0]>=a.x and pos[0]<=a.x+39 and pos[1]>=a.y and pos[1]<=a.y+64 and a.unlock is True:
                        a.map_red(number)
                        interface.click = False
                    elif pos[0]>=b.x and pos[0]<=b.x+39 and pos[1]>=b.y and pos[1]<=b.y+64 and b.unlock is True:
                        b.map_red(number)
                        interface.click = False
                    elif pos[0]>=c.x and pos[0]<=c.x+39 and pos[1]>=c.y and pos[1]<=c.y+64 and c.unlock is True:
                        c.map_red(number)
                        interface.click = False
                    elif pos[0]>=d.x and pos[0]<=d.x+39 and pos[1]>=d.y and pos[1]<=d.y+64 and d.unlock is True:
                        d.map_red(number)
                        interface.click = False
                    elif a.unlock is False and b.unlock is False and c.unlock is False and d.unlock is False:
                        interface.click = False
                    else:
                        interface.click = True
                elif event.type == pygame.QUIT:
                    sys.exit()
```

interface.py→ This module of our project helps us to set a 1375x758 pixel display screen where all sprites dice turn badges and everything that user will be using in the program and looking at the program is programmed.

```
win = pygame.display.set_mode((1375,758))

dice_images = [pygame.image.load('1.png'),pygame.image.load('2.png'),pygame.image.load('3.png'),
               pygame.image.load('4.png'),pygame.image.load('5.png'),pygame.image.load('6.png')]
turn_list=['player1','player2','player3','player4']
turn_sprites = [pygame.image.load('Blue Turn.png'),pygame.image.load('Yellow Turn.png'), pygame.image.load('Green Turn.png'),py
pawn_sprites = [pygame.image.load('Red.png'), pygame.image.load('Blue.png'), pygame.image.load('Yellow.png'), pygame.image.loa
bg = pygame.image.load('Background.png')
img = pygame.image.load('Game Logo.png')
pygame.display.set_caption('LUDO')
pygame.display.set_icon(img)
clock = pygame.time.Clock()
class Players():

    def __init__(self, x, y, width, height):
        self.x = x
        self.y = y
        self.og_x = x
        self.og_y = y
        self.width = width
        self.height = height
        self.unlock = False
        self.step = -1
        self.check_home = -1
```

```
def draw_blue(self, win):
    win.blit(pawn_sprites[1],(self.x,self.y))
def draw_yellow(self,win):
    win.blit(pawn_sprites[2],(self.x,self.y))
def draw_red(self,win):
    win.blit(pawn_sprites[0],(self.x,self.y))
def draw_green(self,win):
    win.blit(pawn_sprites[3],(self.x,self.y))

def unlocked_blue(self):
    self.x = 91-(39//2)
    self.y = 332-(64//2)
def unlocked_yellow(self):
    self.x = 428-(39//2)
    self.y = 94-(64//2)
def unlocked_green(self):
    self.x = 666-(39//2)
    self.y = 426-(64//2)
def unlocked_red(self):
    self.x = 330-(39//2)
    self.y = 666-(64//2)
```

```
def locked(self):
    self.x = self.og_x
    self.y = self.og_y

def map_blue(self, movespace):
    self.check_home += movespace
    if self.check_home >=57:
        self.check_home -= movespace
    else:
        self.step += movespace
        self.x = sprite_movement.map_blue[self.step][0] - (39//2)
        self.y = sprite_movement.map_blue[self.step][1] - (64//2)

def map_yellow(self, movespace):
    self.check_home += movespace
    if self.check_home >=57:
        self.check_home -= movespace
    else:
        self.step += movespace
        self.x = sprite_movement.map_yellow[self.step][0] - (39//2)
        self.y = sprite_movement.map_yellow[self.step][1] - (64//2)

def map_green(self, movespace):
    self.check_home += movespace
    if self.check_home >=57:
        self.check_home -= movespace
    else:
        self.step += movespace
        self.x = sprite_movement.map_green[self.step][0] - (39//2)
        self.y = sprite_movement.map_green[self.step][1] - (64//2)
```

```
def redrawGameWindow(num, face_number):
    win.blit(turn_sprites[num],(759,0))
    win.blit(dice_images[face_number],(759+250,383))
    win.blit(bg,(0,0))
    player1_a.draw_blue(win)
    player1_b.draw_blue(win)
    player1_c.draw_blue(win)
    player1_d.draw_blue(win)
    player2_a.draw_yellow(win)
    player2_b.draw_yellow(win)
    player2_c.draw_yellow(win)
    player2_d.draw_yellow(win)
    player3_a.draw_red(win)
    player3_b.draw_red(win)
    player3_c.draw_red(win)
    player3_d.draw_red(win)
    player4_a.draw_green(win)
    player4_b.draw_green(win)
    player4_c.draw_green(win)
    player4_d.draw_green(win)
    pygame.display.update()
```

```
#Mainloop
run = True

player1_a = Players(101-(39//2), 158-(64//2), 39, 64)
player1_b = Players(162-(39//2), 104-(64//2), 39, 64)
player1_c = Players(220-(39//2), 158-(64//2), 39, 64)
player1_d = Players(163-(39//2), 220-(64//2), 39, 64)
player2_a = Players(596-(39//2), 102-(64//2), 39, 64)
player2_b = Players(534-(39//2), 166-(64//2), 39, 64)
player2_c = Players(595-(39//2), 223-(64//2), 39, 64)
player2_d = Players(651-(39//2), 165-(64//2), 39, 64)
player3_a = Players(167-(39//2), 535-(64//2), 39, 64)
player3_b = Players(103-(39//2), 592-(64//2), 39, 64)
player3_c = Players(167-(39//2), 654-(64//2), 39, 64)
player3_d = Players(220-(39//2), 594-(64//2), 39, 64)
player4_a = Players(590-(39//2), 534-(64//2), 39, 64)
player4_b = Players(532-(39//2), 596-(64//2), 39, 64)
player4_c = Players(591-(39//2), 653-(64//2), 39, 64)
player4_d = Players(652-(39//2), 597-(64//2), 39, 64)
i=0
while run:
    click = True
    if i>3:
        i=0
    clock.tick(100)
    face_num = random.randint(0,5)
    number = face_num+1
    redrawGameWindow(i,face_num)
    if number==6 or number==1:
        unlock_sprite.unlock(turn_list[i])
    else:
        main.diceroll_step(number, turn_list[i])
    collisions.lock_case(turn_player=turn_list[i])
    i+=1

pygame.quit()
```

unlock.py→ This module is used to unlock sprite and let user choose which sprite to choose according to the basic

rules of ludo i.e upon getting either 1 or 6 a pawn/token can unlock

```
def unlock(turn):
    if turn=='player2':
        while interface.click:
            a = interface.player2_a
            b = interface.player2_b
            c = interface.player2_c
            d = interface.player2_d

            ev = pygame.event.get()
            for event in ev:
                if event.type == pygame.MOUSEBUTTONUP:
                    pos = pygame.mouse.get_pos()
                    if pos[0]>=a.x and pos[0]<=a.x+39 and pos[1]>=a.y and pos[1]<=a.y+64 and a.unlock is False:
                        a.unlocked_yellow()
                        a.unlock = True
                        interface.click = False

                    elif pos[0]>=a.x and pos[0]<=a.x+39 and pos[1]>=a.y and pos[1]<=a.y+64 and a.unlock is True:
                        if pos[0]>=a.x and pos[0]<=a.x+39 and pos[1]>=a.y and pos[1]<=a.y+64:
                            a.map_yellow(interface.number)
                            interface.click = False

                    elif pos[0]>=b.x and pos[0]<=b.x+39 and pos[1]>=b.y and pos[1]<=b.y+64 and b.unlock is False:
                        b.unlocked_yellow()
                        b.unlock = True
                        interface.click = False

                    elif pos[0]>=b.x and pos[0]<=b.x+39 and pos[1]>=b.y and pos[1]<=b.y+64 and b.unlock is True:
                        if pos[0]>=b.x and pos[0]<=b.x+39 and pos[1]>=b.y and pos[1]<=b.y+64:
                            b.map_yellow(interface.number)
                            interface.click = False
```

```
elif turn=="player1":
    while interface.click:
        a = interface.player1_a
        b = interface.player1_b
        c = interface.player1_c
        d = interface.player1_d

        ev = pygame.event.get()
        for event in ev:
            if event.type == pygame.MOUSEBUTTONUP:
                pos = pygame.mouse.get_pos()
                if pos[0]>=a.x and pos[0]<=a.x+39 and pos[1]>=a.y and pos[1]<=a.y+64 and a.unlock is False:
                    a.unlocked_blue()
                    a.unlock = True
                    interface.click = False

                elif pos[0]>=a.x and pos[0]<=a.x+39 and pos[1]>=a.y and pos[1]<=a.y+64 and a.unlock is True:
                    if pos[0]>=a.x and pos[0]<=a.x+39 and pos[1]>=a.y and pos[1]<=a.y+64:
                        a.map_blue(interface.number)
                        interface.click = False

                elif pos[0]>=b.x and pos[0]<=b.x+39 and pos[1]>=b.y and pos[1]<=b.y+64 and b.unlock is False:
                    b.unlocked_blue()
                    b.unlock = True
                    interface.click = False

                elif pos[0]>=b.x and pos[0]<=b.x+39 and pos[1]>=b.y and pos[1]<=b.y+64 and b.unlock is True:
                    if pos[0]>=b.x and pos[0]<=b.x+39 and pos[1]>=b.y and pos[1]<=b.y+64:
                        b.map_blue(interface.number)
                        interface.click = False


                elif pos[0]>=c.x and pos[0]<=c.x+39 and pos[1]>=c.y and pos[1]<=c.y+64 and c.unlock is False:
                    c.unlocked_blue()
                    c.unlock = True
                    interface.click = False
```

```
elif turn=="player3":
    while interface.click:
        a = interface.player4_a
        b = interface.player4_b
        c = interface.player4_c
        d = interface.player4_d

        ev = pygame.event.get()
        for event in ev:
            if event.type == pygame.MOUSEBUTTONUP:
                pos = pygame.mouse.get_pos()
                if pos[0]>=a.x and pos[0]<=a.x+39 and pos[1]>=a.y and pos[1]<=a.y+64 and a.unlock is False:
                    a.unlocked_green()
                    a.unlock = True
                    interface.click = False

                elif pos[0]>=a.x and pos[0]<=a.x+39 and pos[1]>=a.y and pos[1]<=a.y+64 and a.unlock is True:
                    if pos[0]>=a.x and pos[0]<=a.x+39 and pos[1]>=a.y and pos[1]<=a.y+64:
                        a.map_green(interface.number)
                        interface.click = False

                elif pos[0]>=b.x and pos[0]<=b.x+39 and pos[1]>=b.y and pos[1]<=b.y+64 and b.unlock is False:
                    b.unlocked_green()
                    b.unlock = True
                    interface.click = False

                elif pos[0]>=b.x and pos[0]<=b.x+39 and pos[1]>=b.y and pos[1]<=b.y+64 and b.unlock is True:
                    if pos[0]>=b.x and pos[0]<=b.x+39 and pos[1]>=b.y and pos[1]<=b.y+64:
                        b.map_green(interface.number)
                        interface.click = False


                elif pos[0]>=c.x and pos[0]<=c.x+39 and pos[1]>=c.y and pos[1]<=c.y+64 and c.unlock is False:
                    c.unlocked_green()
                    c.unlock = True
                    interface.click = False
```

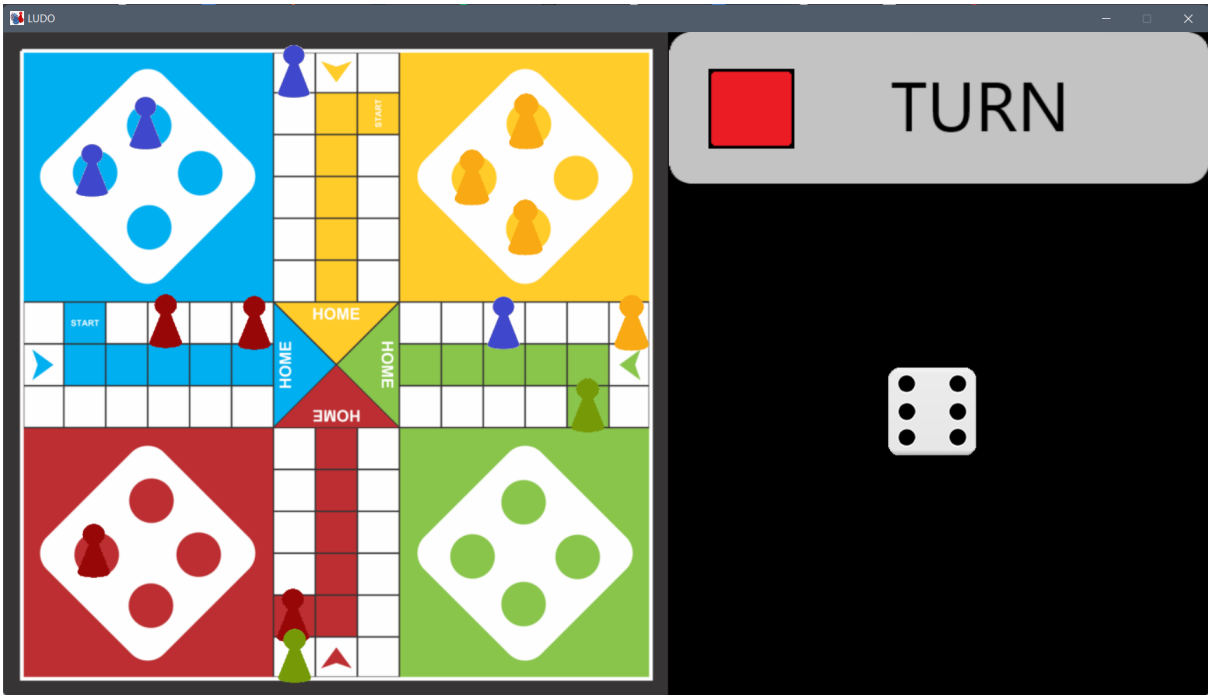
sprite_movement.py→ This file contains numpy arrays consisting of the points on the cartesian plane so that sprites can move around the board.

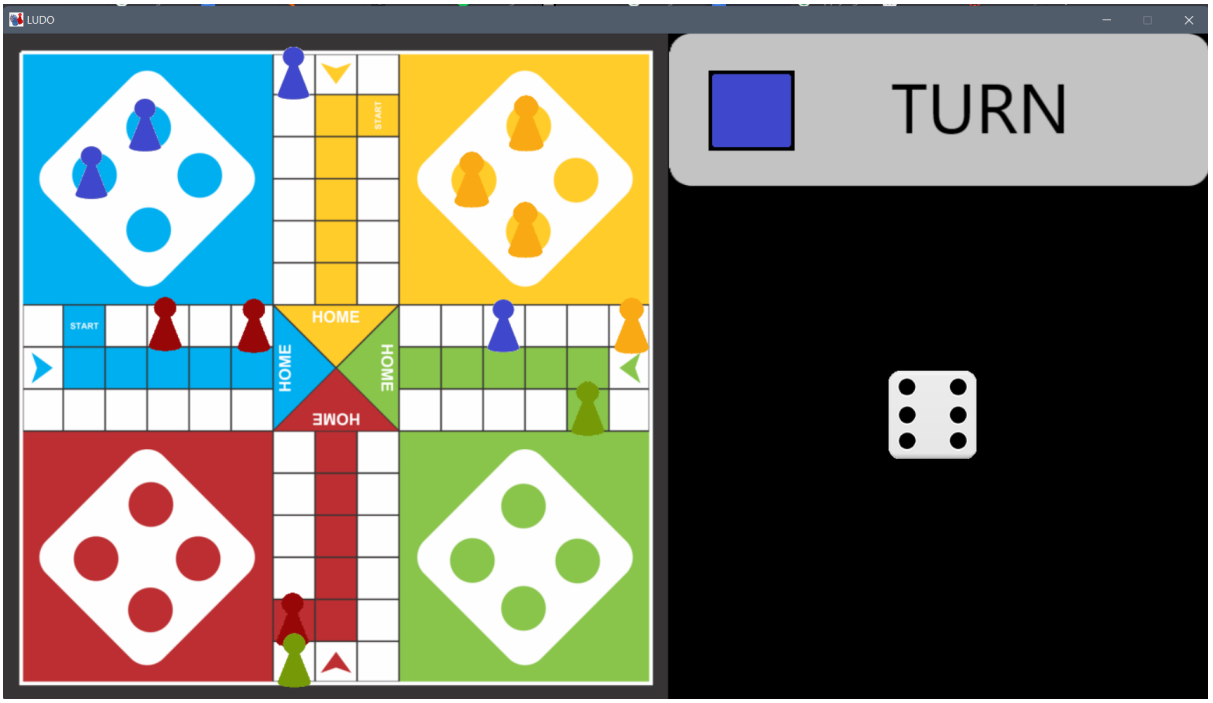
collisions.py → This file is used to check the rule in the game of ludo where a pawn/token strikes out another token on the account of when both of them are in the same box

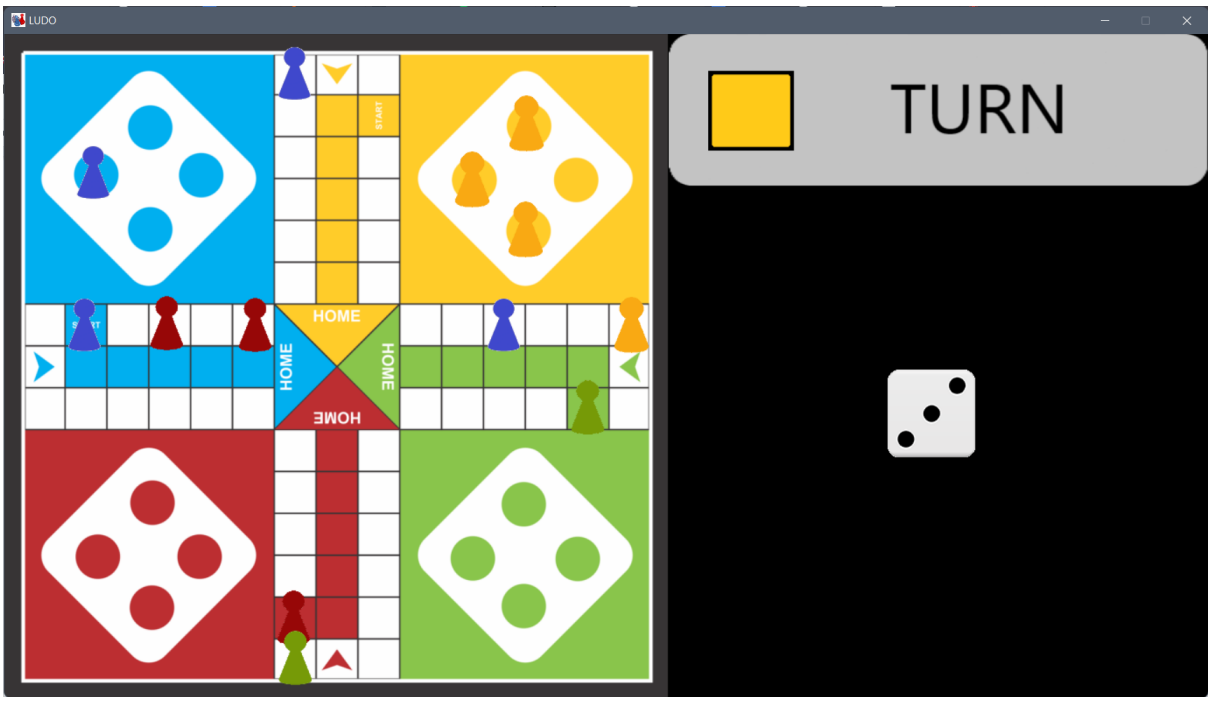
```
def lock_case(turn_player):
    l1 = [interface.player1_a, interface.player1_b, interface.player1_c, interface.player1_d]
    l2 = [interface.player2_a, interface.player2_b, interface.player2_c, interface.player2_d]
    l3 = [interface.player3_a, interface.player3_b, interface.player3_c, interface.player3_d]
    l4 = [interface.player4_a, interface.player4_b, interface.player4_c, interface.player4_d]
    if turn_player=='player1':
        for i in l1:
            for j in l2:
                if i.x==j.x and i.y==j.y:
                    j.unlock=False
                    j.locked()
                    break
                else:
                    continue

            for k in l3:
                if i.x==k.x and i.y==k.y:
                    k.unlock=False
                    k.locked()
                    break
                else:
                    continue
            for l in l4:
                if i.x==l.x and i.y==l.y:
                    l.unlock=False
                    l.locked()
                    break
                else:
                    continue

    elif turn_player=='player2':
        for i in l2:
            for j in l1:
                if i.x==j.x and i.y==j.y:
                    j.unlock=False
                    j.locked()
```

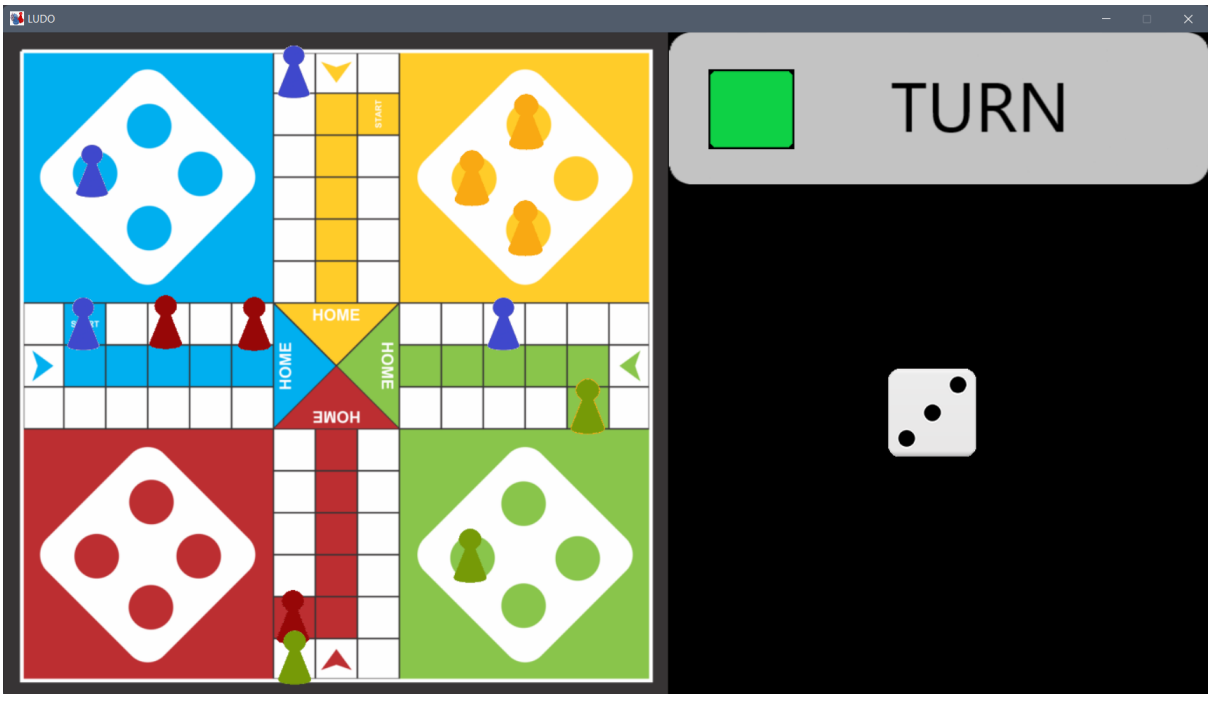
```
            for k in l3:
                if i.x==k.x and i.y==k.y:
                    k.unlock=False
                    k.locked()
                    break
                else:
                    continue
            for l in l4:
                if i.x==l.x and i.y==l.y:
                    l.unlock=False
                    l.locked()
                    break
                else:
                    continue

    elif turn_player=='player4':
        for i in l3:
            for j in l1:
                if i.x==j.x and i.y==j.y:
                    j.unlock=False
                    j.locked()
                    break
                else:
                    continue
            for k in l2:
                if i.x==k.x and i.y==k.y:
                    k.unlock=False
                    k.locked()
                    break
                else:
                    continue
            for l in l4:
                if i.x==l.x and i.y==l.y:
                    l.unlock=False
                    l.locked()
                    break
```

OUTPUT SCREENSHOTS






LUDO
START
HOME
HOME
HOME
HOME
TURN


LUDO
START
HOME
HOME
HOME
HOME
TURN

FUTURE SCOPE

This project can be enhanced in future via many different ways. A menu can be added so that the user can select the number of players. The project can also add bots so that users can play against bots also. Different types of gaming boards can be setup and players can get after achieving a certain amount of score which can be displayed to everyone through web development