```
import pygame

pygame.init()

width = 800
height = 600

black = (0, 0, 0)
white = (255, 255, 255)
grey = (100, 100, 100)
red = (120, 0, 0)
light_green = (0, 255, 0)
light_red = (255, 0, 0)
blue = (0, 0, 255)
background = (54, 54, 54)
board_color = (0, 31, 0)

display = pygame.display.set_mode((800, 600))
pygame.display.set_caption('Jogo de Damas')
pygame.font.init()
clock = pygame.time.Clock()
class Game:
    def __init__(self):
        self.status = 'Jogando'
        self.round = 1
        self.players = ('x', 'o')
        self.selected_spaces = None
        self.jumping = False
        self.matriz_players = [['x', '-', 'x', '-', 'x', '-', 'x', '-'],
                               ['-', 'x', '-', 'x', '-', 'x', '-',
'x'],
                               ['x', '-', 'x', '-', 'x', '-', 'x', '-
'],
                               ['-', '-', '-', '-', '-', '-', '-', '-
'],
                               ['-', '-', '-', '-', '-', '-', '-', '-
'],
                               ['-', 'o', '-', 'o', '-', 'o', '-',
'o'],
                               ['o', '-', 'o', '-', 'o', '-', 'o', '-
'],
                               ['-', 'o', '-', 'o', '-', 'o', '-',
'o']]
    def check_click(self, pos):
        round = self.round % 2
        if self.status == "Jogando":
            row, column = row_clicked(pos), column_clicked(pos)
            if self.selected_spaces:
                movement = self.is_movement_valid(self.players[round],
self.selected_spaces, row, column)
                if movement[0]:
                    self.play(self.players[round], self.selected_spaces,
row, column, movement[1])
                elif row == self.selected_spaces[0] and column ==
self.selected_spaces[1]:
                    move =
self.obligatories_movement(self.selected_spaces)
                    if move[0] == []:
```

```
                        if self.jumping:
                            self.jumping = False
                            self.next_round()
                    self.selected_spaces = None
            else:
                if self.matriz_players[row][column].lower() ==
self.players[round]:
                    self.selected_spaces = [row, column]
    def is_movement_valid(self, player, cedula_location, row_destiny,
column_destiny):

        row_origin = cedula_location[0]
        column_origin = cedula_location[1]

        obligatories = self.all_obligatories()

        if obligatories != {}:
            if (row_origin, column_origin) not in obligatories:
                return False, None
            elif [row_destiny, column_destiny] not in
obligatories[(row_origin, column_origin)]:
                return False, None

        movement, jump = self.possibly_movements(cedula_location)

        if [row_destiny, column_destiny] in movement:
            if jump:
                if len(jump) == 1:
                    return True, jump[0]
                else:
                    for i in range(len(jump)):
                        if abs(jump[i][0] - row_destiny) == 1 and
abs(jump[i][1] - column_destiny) == 1:
                            return True, jump[i]

            if self.jumping:
                return False, None

            return True, None

        return False, None
    def all_obligatories(self):
        all = {}

        for r in range(len(self.matriz_players)):
            for c in range(len(self.matriz_players[r])):
                ob, jumps = self.obligatories_movement((r, c))
                if ob != []:
                    all[(r, c)] = ob
        return all
    def existy_possibly(self):
        for l in range(len(self.matriz_players)):
            for c in range(len(self.matriz_players[l])):
                if self.possibly_movements((l, c))[0]:
                    return True
        return False
    def obligatories_movement(self, cedula_location):
        obligatories = []
```

```
        position_cedula_jumped = []

        l = cedula_location[0]
        c = cedula_location[1]

        player = self.players[self.round % 2]
        index = self.players.index(player)

        array = [player.lower(), player.upper(), '-']

        if self.matriz_players[l][c].islower() and
self.matriz_players[l][c] == player and \
                self.round % 2 == index:
            if l > 0:
                if c < 7:
                    if self.matriz_players[l - 1][c + 1].lower() not in
array:
                        l_x = l - 1
                        l_c = c + 1

                        if l_x - 1 >= 0 and l_c + 1 <= 7:
                            if self.matriz_players[l_x - 1][l_c + 1] ==
'-':
                                obligatories.append([l_x - 1, l_c + 1])
                                position_cedula_jumped.append((l_x, l_c))
                if c > 0:
                    if self.matriz_players[l - 1][c - 1].lower() not in
array:
                        l_x = l - 1
                        l_c = c - 1

                        if l_x - 1 >= 0 and l_c - 1 >= 0:
                            if self.matriz_players[l_x - 1][l_c - 1] ==
'-':
                                obligatories.append([l_x - 1, l_c - 1])
                                position_cedula_jumped.append((l_x, l_c))
            if l < 7:
                if c < 7:
                    if self.matriz_players[l + 1][c + 1].lower() not in
array:
                        l_x = l + 1
                        l_c = c + 1

                        if l_x + 1 <= 7 and l_c + 1 <= 7:
                            if self.matriz_players[l_x + 1][l_c + 1] ==
'-':
                                obligatories.append([l_x + 1, l_c + 1])
                                position_cedula_jumped.append((l_x, l_c))
                if c > 0:
                    if self.matriz_players[l + 1][c - 1].lower() not in
array:
                        l_x = l + 1
                        l_c = c - 1

                        if l_x + 1 <= 7 and l_c - 1 >= 0:
                            if self.matriz_players[l_x + 1][l_c - 1] ==
'-':
                                obligatories.append([l_x + 1, l_c - 1])
```

```
                                position_cedula_jumped.append((l_x, l_c))

        elif self.matriz_players[l][c].isupper() and
self.matriz_players[l][c] == player.upper() and \
                self.round % 2 == index:

            if not self.jumping and (player.lower() == 'x' and l != 7) or
(player.lower() == 'o' and l != 0):
                count_row = l
                count_column = c
                while True:
                    if count_row - 1 < 0 or count_column - 1 < 0:
                        break
                    else:
                        if self.matriz_players[count_row -
1][count_column - 1] not in array:
                            l_x = count_row - 1
                            l_c = count_column - 1

                            if l_x - 1 >= 0 and l_c - 1 >= 0:
                                if self.matriz_players[l_x - 1][l_c - 1]
== '-':
                                    position_cedula_jumped.append((l_x,
l_c))
                                    while True:
                                        if l_x - 1 < 0 or l_c - 1 < 0:
                                            break
                                        else:
                                            if self.matriz_players[l_x -
1][l_c - 1] == '-':
                                                obligatories.append([l_x
- 1, l_c - 1])
                                            else:
                                                break
                                        l_x -= 1
                                        l_c -= 1
                            break
                    count_row -= 1
                    count_column -= 1
                count_row = l
                count_column = c
                while True:
                    if count_row - 1 < 0 or count_column + 1 > 7:
                        break
                    else:
                        if self.matriz_players[count_row -
1][count_column + 1] not in array:
                            l_x = count_row - 1
                            l_c = count_column + 1

                            if l_x - 1 >= 0 and l_c + 1 <= 7:
                                if self.matriz_players[l_x - 1][l_c + 1]
== '-':
                                    position_cedula_jumped.append((l_x,
l_c))
                                    while True:
                                        if l_x - 1 < 0 or l_c + 1 > 7:
                                            break
```

```
                                        else:
                                            if self.matriz_players[l_x -
1][l_c + 1] == '-':
                                                obligatories.append([l_x
- 1, l_c + 1])
                                            else:
                                                break
                                        l_x -= 1
                                        l_c += 1
                                break
                        count_row -= 1
                        count_column += 1

                    count_row = l
                    count_column = c
                    while True:
                        if count_row + 1 > 7 or count_column + 1 > 7:
                            break
                        else:
                            if self.matriz_players[count_row +
1][count_column + 1] not in array:
                                l_x = count_row + 1
                                l_c = count_column + 1

                                if l_x + 1 <= 7 and l_c + 1 <= 7:
                                    if self.matriz_players[l_x + 1][l_c + 1]
== '-':
                                        position_cedula_jumped.append((l_x,
l_c))
                                        while True:
                                            if l_x + 1 > 7 or l_c + 1 > 7:
                                                break
                                            else:
                                                if self.matriz_players[l_x +
1][l_c + 1] == '-':
                                                    obligatories.append([l_x
+ 1, l_c + 1])
                                                else:
                                                    break
                                            l_x += 1
                                            l_c += 1
                                break
                        count_row += 1
                        count_column += 1

                    count_row = l
                    count_column = c
                    while True:
                        if count_row + 1 > 7 or count_column - 1 < 0:
                            break
                        else:
                            if self.matriz_players[count_row +
1][count_column - 1] not in array:
                                l_x = count_row + 1
                                l_c = count_column - 1

                                if l_x + 1 <= 7 and l_c - 1 >= 0:
```

```
                                    if self.matriz_players[l_x + 1][l_c - 1]
== '-':
                                        position_cedula_jumped.append((l_x,
l_c))
                                        while True:
                                            if l_x + 1 > 7 or l_c - 1 < 0:
                                                break
                                            else:
                                                if self.matriz_players[l_x +
1][l_c - 1] == '-':
                                                    obligatories.append([l_x
+ 1, l_c - 1])
                                                else:
                                                    break
                                            l_x += 1
                                            l_c -= 1
                            break
                    count_row += 1
                    count_column -= 1

        return obligatories, position_cedula_jumped

    def possibly_movements(self, cedula_location):
        movements, jumps = self.obligatories_movement(cedula_location)

        if movements == []:
            row_actual = cedula_location[0]
            column_actual = cedula_location[1]

            if self.matriz_players[row_actual][column_actual].islower():
                if self.matriz_players[row_actual][column_actual] == 'o':
                    if row_actual > 0:
                        if column_actual < 7:
                            if self.matriz_players[row_actual -
1][column_actual + 1] == '-':
                                movements.append([row_actual - 1,
column_actual + 1])
                        if column_actual > 0:
                            if self.matriz_players[row_actual -
1][column_actual - 1] == '-':
                                movements.append([row_actual - 1,
column_actual - 1])

                elif self.matriz_players[row_actual][column_actual] ==
'x':
                    if row_actual < 7:
                        if column_actual < 7:
                            if self.matriz_players[row_actual +
1][column_actual + 1] == '-':
                                movements.append([row_actual + 1,
column_actual + 1])
                        if column_actual > 0:
                            if self.matriz_players[row_actual +
1][column_actual - 1] == '-':
                                movements.append([row_actual + 1,
column_actual - 1])
            elif
self.matriz_players[row_actual][column_actual].isupper():
```

```
                count_row = row_actual
                count_column = column_actual
                while True:
                    if count_row - 1 < 0 or count_column - 1 < 0:
                        break
                    else:
                        if self.matriz_players[count_row -
1][count_column - 1] == '-':
                            movements.append([count_row - 1, count_column
- 1])
                        else:
                            break
                    count_row -= 1
                    count_column -= 1

                count_row = row_actual
                count_column = column_actual
                while True:
                    if count_row - 1 < 0 or count_column + 1 > 7:
                        break
                    else:
                        if self.matriz_players[count_row -
1][count_column + 1] == '-':
                            movements.append([count_row - 1, count_column
+ 1])
                        else:
                            break
                    count_row -= 1
                    count_column += 1

                count_row = row_actual
                count_column = column_actual
                while True:
                    if count_row + 1 > 7 or count_column + 1 > 7:
                        break
                    else:
                        if self.matriz_players[count_row +
1][count_column + 1] == '-':
                            movements.append([count_row + 1, count_column
+ 1])
                        else:
                            break
                    count_row += 1
                    count_column += 1

                count_row = row_actual
                count_column = column_actual
                while True:
                    if count_row + 1 > 7 or count_column - 1 < 0:
                        break
                    else:
                        if self.matriz_players[count_row +
1][count_column - 1] == '-':
                            movements.append([count_row + 1, count_column
- 1])
                        else:
                            break
                    count_row += 1
```

```
                    count_column -= 1

        return movements, jumps
    def play(self, player, cedula_location, row_destiny, column_destiny,
jump):
        row_actual = cedula_location[0]
        column_actual = cedula_location[1]
        char = self.matriz_players[row_actual][column_actual]

        self.matriz_players[row_destiny][column_destiny] = char
        self.matriz_players[row_actual][column_actual] = '-'

        if jump:
            self.jumping = True

        if (player == 'x' and row_destiny == 7) or (player == 'o' and
row_destiny == 0):
            if not self.jumping:
                self.matriz_players[row_destiny][column_destiny] =
char.upper()
            elif not self.possibly_movements((row_destiny,
column_destiny))[0]:
                self.matriz_players[row_destiny][column_destiny] =
char.upper()

        if jump:
            self.matriz_players[jump[0]][jump[1]] = '-'
            self.selected_spaces = [row_destiny, column_destiny]
            self.jumping = True

        else:
            self.selected_spaces = None
            self.next_round()
        winner = self.check_winner()

        if winner != None:
            self.status = 'Perdeu'
    def next_round(self):
        self.round += 1
    def check_winner(self):

        x = sum([counter.count('x') + counter.count('X') for counter in
self.matriz_players])
        o = sum([counter.count('o') + counter.count('O') for counter in
self.matriz_players])

        if x == 0:
            return 'o'

        if o == 0:
            return 'x'

        if x == 1 and o == 1:
            return 'Empate'

        if self.selected_spaces:
            if not self.possibly_movements(self.selected_spaces)[0]:
                if x == 1 and self.round % 2 == 0:
```

```
                    return 'o'
                if o == 1 and self.round % 2 == 1:
                    return 'x'

        if not self.existy_possibly():
            return 'Empate'

        return None

    def draw(self):
        matriz = []

        for i in range(8):
            if i % 2 == 0:
                matriz.append(['#', '-', '#', '-', '#', '-', '#', '-'])
            else:
                matriz.append(['-', '#', '-', '#', '-', '#', '-', '#'])

        y = 0
        for l in range(len(matriz)):
            x = 0
            for c in range(len(matriz[l])):
                if matriz[l][c] == '#':
                    pygame.draw.rect(display, board_color, (x, y, 75,
75))
                else:
                    pygame.draw.rect(display, white, (x, y, 75, 75))
                x += 75
            y += 75

        if self.selected_spaces:
            obligatories = self.all_obligatories()
            move = self.possibly_movements(self.selected_spaces)

            if obligatories != {}:
                if (self.selected_spaces[0], self.selected_spaces[1]) not
in obligatories:
                    x_red = height / 8 * self.selected_spaces[1]
                    y_red = height / 8 * self.selected_spaces[0]

                    pygame.draw.rect(display, light_red, (x_red, y_red,
75, 75))
                else:
                    if move[0] == []:
                        x_red = height / 8 * self.selected_spaces[1]
                        y_red = height / 8 * self.selected_spaces[0]

                        pygame.draw.rect(display, light_red, (x_red,
y_red, 75, 75))
                    else:
                        for i in range(len(move[0])):
                            x_possivel = height / 8 * move[0][i][1]
                            y_possivel = height / 8 * move[0][i][0]

                            pygame.draw.rect(display, light_green,
(x_possivel, y_possivel, 75, 75))
            else:
                if self.jumping:
```

```
                    x_red = height / 8 * self.selected_spaces[1]
                    y_red = height / 8 * self.selected_spaces[0]

                    pygame.draw.rect(display, light_red, (x_red, y_red,
75, 75))
                else:
                    if move[0] == []:
                        x_red = height / 8 * self.selected_spaces[1]
                        y_red = height / 8 * self.selected_spaces[0]

                        pygame.draw.rect(display, light_red, (x_red,
y_red, 75, 75))
                    else:
                        for i in range(len(move[0])):
                            x_possivel = height / 8 * move[0][i][1]
                            y_possivel = height / 8 * move[0][i][0]

                            pygame.draw.rect(display, light_green,
(x_possivel, y_possivel, 75, 75))

        for l in range(len(self.matriz_players)):
            for c in range(len(self.matriz_players[l])):
                element = self.matriz_players[l][c]
                if element != '-':
                    x = height / 8 * c + height / 16
                    y = height / 8 * l + height / 16

                    if element.lower() == 'x':
                        pygame.draw.circle(display, red, (x, y), 20, 0)
                        if element == 'X':
                            pygame.draw.circle(display, black, (x, y),
10, 0)
                            pygame.draw.circle(display, blue, (x, y), 5,
0)
                    else:
                        pygame.draw.circle(display, white, (x, y), 20, 0)
                        if element == 'O':
                            pygame.draw.circle(display, black, (x, y),
10, 0)
                            pygame.draw.circle(display, blue, (x, y), 5,
0)

        font = pygame.font.Font(None, 20)

        x = sum([counter.count('x') + counter.count('X') for counter in
self.matriz_players])
        o = sum([counter.count('o') + counter.count('O') for counter in
self.matriz_players])

        if self.status != 'Game Over':

            surface_text, rect_text = text_objects("Vermelho: " + str(12
- o), font, light_red)
            rect_text.center = (650, 30)
            display.blit(surface_text, rect_text)

            surface_text, rect_text = text_objects("Branco: " + str(12 -
x), font, white)
```

```
            rect_text.center = (650, height - 30)
            display.blit(surface_text, rect_text)

            if self.round % 2 == 1:
                surface_text, rect_text = text_objects("Vez do Branco",
font, white)
                rect_text.center = (700, height / 2)
                display.blit(surface_text, rect_text)
            else:
                surface_text, rect_text = text_objects("Vez do Vermelho",
font, light_red)
                rect_text.center = (700, height / 2)
                display.blit(surface_text, rect_text)
        else:
            surface_text, rect_text = text_objects("Perdeu", font, blue)
            rect_text.center = (700, height / 3)
            display.blit(surface_text, rect_text)

def text_objects(text, font, color):
    textSurface = font.render(text, True, color)
    return textSurface, textSurface.get_rect()

def cria_botao(msg, sqr, color1, color2, color_text, action=None):
    mouse = pygame.mouse.get_pos()
    click = pygame.mouse.get_pressed()

    if sqr[0] + sqr[2] > mouse[0] > sqr[0] and sqr[1] + sqr[3] > mouse[1]
> sqr[1]:
        pygame.draw.rect(display, color2, sqr)
        if click[0] == 1 and action != None:
            action()
    else:
        pygame.draw.rect(display, color1, sqr)

    litle_font = pygame.font.SysFont('comicsansms', 20)
    surface_text, rect_text = text_objects(msg, litle_font, color_text)
    rect_text.center = (sqr[0] + 60, sqr[1] + 20)
    display.blit(surface_text, rect_text)

def screen_winner(winner):
    exit = False

    while not exit:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                exit = True
                pygame.quit()
                quit()
            if event.type == pygame.KEYDOWN or event.type ==
pygame.MOUSEBUTTONDOWN:
                exit = True

        display.fill(black)

        font = pygame.font.SysFont('comicsansms', 50)

        surface_text, rect_text = None, None
```

```
        if winner == "empate":
            surface_text, rect_text = text_objects("EMPATE!", font,
white)
        elif winner == "x":
            surface_text, rect_text = text_objects("VITORIA DO
VERMELHO", font, red)
        elif winner == "o":
            surface_text, rect_text = text_objects("VITORIA DO BRANCO",
font, white)

        rect_text.center = ((width / 2), height / 3)
        display.blit(surface_text, rect_text)

        pygame.display.update()
        clock.tick(60)

def column_clicked(pos):
    x = pos[0]
    for i in range(1, 8):
        if x < i * height / 8:
            return i - 1
    return 7


def row_clicked(pos):
    y = pos[1]
    for i in range(1, 8):
        if y < i * height / 8:
            return i - 1
    return 7
def loop_game():
    exit = False

    game = Game()

    while not exit:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                exit = True
                pygame.quit()
                quit()
            if event.type == pygame.MOUSEBUTTONDOWN:
                game.check_click(pygame.mouse.get_pos())

        display.fill(black)
        game.draw()

        winner = game.check_winner()

        if winner is not None:
            exit = True
            screen_winner(winner)

        pygame.display.update()
        clock.tick(60)

loop_game()
pygame.quit()
```

```
quit()
```