```
importpygame
import random
# Initialize Pygame package
pygame.init()
# Define window size and title
WINDOW_WIDTH = 500
WINDOW_HEIGHT = 500
WINDOW_TITLE = "Snake Game"
window = pygame.display.set_mode((WINDOW_WIDTH, WINDOW_HEIGHT))
pygame.display.set_caption(WINDOW_TITLE)
# Definecolors
BLACK = (0, 0, 0)
WHITE = (255, 255, 255)
RED = (255, 0, 0)
GREEN = (0, 255, 0)
# Define Snake class
class Snake:
def __init__(self):
self.x = WINDOW_WIDTH/2
self.y = WINDOW_HEIGHT/2
self.color = GREEN
self.direction = "right"
self.speed = 10
self.size = 10
self.body = [(self.x, self.y)]
def move(self):
ifself.direction == "right":
self.x += self.speed
```

```
elifself.direction == "left":
self.x -= self.speed
elifself.direction == "down":
self.y += self.speed
elifself.direction == "up":
self.y -= self.speed
self.body.insert(0, (self.x, self.y))
self.body.pop()
def draw(self):
for x, y in self.body:
pygame.draw.rect(window, self.color, [x, y, self.size, self.size])
# Define Fruit class
class Fruit:
def __init__(self):
self.x = random.randint(0, WINDOW_WIDTH-self.size)
self.y = random.randint(0, WINDOW_HEIGHT-self.size)
self.color = RED
self.size = 10
def draw(self):
pygame.draw.rect(window, self.color, [self.x, self.y, self.size, self.size])
# Define function to check collision with fruit
defcheck_fruit_collision(snake, fruit):
ifsnake.x == fruit.x and snake.y == fruit.y:
fruit.x = random.randint(0, WINDOW_WIDTH-fruit.size)
fruit.y = random.randint(0, WINDOW_HEIGHT-fruit.size)
snake.body.append((snake.x, snake.y))
return True
return False
```

```
# Define function to check collision with window
defcheck_window_collision(snake):
ifsnake.x< 0 or snake.x> WINDOW_WIDTH-snake.size or snake.y< 0 or snake.y>
WINDOW_HEIGHT-snake.size:
return True
return False
# Initialize Snake and Fruit objects
snake = Snake()
fruit = Fruit
```

```
importpygame
# Initialize pygame
pygame.init()
# Set screen size and title
screen_size = (640, 640)
screen = pygame.display.set_mode(screen_size)
pygame.display.set_caption('Chess Board')
# Definecolors
black = (0, 0, 0)
white = (255, 255, 255)
brown = (153, 76, 0)
# Define function to draw the board
defdraw_board():
for row in range(8):
for col in range(8):
square_color = white if (row + col) % 2 == 0 else brown
square_rect = pygame.Rect(col * 80, row * 80, 80, 80)
pygame.draw.rect(screen, square_color, square_rect)
# Define function to draw the pieces
defdraw_pieces(board):
piece_images = {
'r': pygame.image.load('images/rook.png'),
'n': pygame.image.load('images/knight.png'),
'b': pygame.image.load('images/bishop.png'),
'q': pygame.image.load('images/queen.png'),
'k': pygame.image.load('images/king.png'),
'p': pygame.image.load('images/pawn.png')
}
```

```
for row in range(8):
for col in range(8):
piece = board[row][col]
if piece != '.':
piece_image = piece_images[piece]
piece_rect = pygame.Rect(col * 80, row * 80, 80, 80)
screen.blit(piece_image, piece_rect)
# Define initial state of the board
board = [
['r', 'n', 'b', 'q', 'k', 'b', 'n', 'r'],
['p', 'p', 'p', 'p', 'p', 'p', 'p', 'p'],
['.', '.', '.', '.', '.', '.', '.', '.'],
['.', '.', '.', '.', '.', '.', '.', '.'],
['.', '.', '.', '.', '.', '.', '.', '.'],
['.', '.', '.', '.', '.', '.', '.', '.'],
['P', 'P', 'P', 'P', 'P', 'P', 'P', 'P'],
['R', 'N', 'B', 'Q', 'K', 'B', 'N', 'R']
]
# Draw board and pieces
draw_board()
draw_pieces(board)
# Start game loop
while True:
for event in pygame.event.get():
ifevent.type == pygame.QUIT:
pygame.quit()
quit()
pygame.display.update()
```