**Monkey–Banana Problem**

**P Kartikeya Raj**

**2023001492**

**Aim**

To simulate the Monkey–Banana problem using a goal-based agent in a 2D environment.

**Agent**

Monkey (goal-based agent)

**Goal**

Grab the banana and reach the door.

**Environment**

* Single room
* Fully observable
* Deterministic
* Static
* Discrete (2D)

**Objects**

Monkey, Box, Banana, Door, Walls

**Actions**

* Move (left, right, up, down)
* Push box
* Climb box
* Grab banana

**Initial State**

Monkey and box on the floor. Banana is out of reach.

**Goal State**

Monkey has the banana and reaches the door.

**Code**

import pygame

import sys

pygame.init()

# Window

WIDTH, HEIGHT = 800, 500

screen = pygame.display.set\_mode((WIDTH, HEIGHT))

pygame.display.set\_caption("Monkey Banana Room")

clock = pygame.time.Clock()

# Load images

def load\_img(path, size):

    try:

        img = pygame.image.load(path)

        return pygame.transform.scale(img, size)

    except Exception as e:

        print("Image load error:", path)

        pygame.quit()

        sys.exit()

monkey\_img = load\_img("images/monkey.png", (60, 60))

monkey\_banana\_img = load\_img("images/monkey\_banana.png", (60, 60))

box\_img = load\_img("images/box.png", (80, 80))

banana\_img = load\_img("images/banana.png", (40, 40))

door\_img = load\_img("images/door.png", (110, 110))

# Game objects (Rectangles)

monkey = pygame.Rect(50, 350, 60, 60)

box = pygame.Rect(350, 350, 80, 80)

banana = pygame.Rect(380, 120, 40, 40)

door = pygame.Rect(700, 50, 80, 80)

monkey\_on\_box = False

has\_banana = False

# Walls (room boundary)

walls = [

    pygame.Rect(0, 0, WIDTH, 10),         # top

    pygame.Rect(0, 0, 10, HEIGHT),        # left

    pygame.Rect(0, HEIGHT-10, WIDTH, 10), # bottom

    pygame.Rect(WIDTH-10, 0, 10, HEIGHT)  # right

]

FONT = pygame.font.SysFont(None, 26)

MOVE = 5

def draw():

    screen.fill((200, 220, 240))

    # Draw walls

    for wall in walls:

        pygame.draw.rect(screen, (100,100,100), wall)

    # Draw objects

    if not has\_banana:

        screen.blit(banana\_img, banana)

    screen.blit(box\_img, box)

    if has\_banana:

        screen.blit(monkey\_banana\_img, monkey)

    else:

        screen.blit(monkey\_img, monkey)

    screen.blit(door\_img, door)

    # Info text

    info = FONT.render(

        "Arrows Move | P Push | C Climb | G Grab | R Reset",

        True, (0,0,0)

    )

    screen.blit(info, (20, 20))

    pygame.display.update()

def reset\_game():

    global monkey, box, banana, monkey\_on\_box, has\_banana

    monkey.topleft = (50, 350)

    box.topleft = (350, 350)

    banana.topleft = (380, 120)

    monkey\_on\_box = False

    has\_banana = False

# Collision helper

def check\_collision(rect):

    for wall in walls:

        if rect.colliderect(wall):

            return True

    return False

while True:

    clock.tick(60)

    for event in pygame.event.get():

        if event.type == pygame.QUIT:

            pygame.quit()

            sys.exit()

    keys = pygame.key.get\_pressed()

    dx = dy = 0

    if keys[pygame.K\_LEFT]:

        dx = -MOVE

    if keys[pygame.K\_RIGHT]:

        dx = MOVE

    if keys[pygame.K\_UP]:

        dy = -MOVE

    if keys[pygame.K\_DOWN]:

        dy = MOVE

    # Push box and stick monkey

    if keys[pygame.K\_p] and monkey.colliderect(box) and not monkey\_on\_box:

        # Try moving box

        box.x += dx

        box.y += dy

        if check\_collision(box):

            box.x -= dx

            box.y -= dy

            dx = dy = 0  # prevent monkey from moving into wall

        # Monkey sticks to box

        monkey.x += dx

        monkey.y += dy

    else:

        # Move monkey normally

        monkey.x += dx

        monkey.y += dy

        if check\_collision(monkey):

            monkey.x -= dx

            monkey.y -= dy

    # Climb box

    if keys[pygame.K\_c] and monkey.colliderect(box):

        monkey\_on\_box = True

        monkey.bottom = box.top + 10

    # Grab banana

    if keys[pygame.K\_g] and monkey\_on\_box and monkey.colliderect(banana):

        has\_banana = True

    # Reset game

    if keys[pygame.K\_r]:

        reset\_game()

    # Check for door (end game)

  # Check for door (end game)

    if has\_banana and monkey.colliderect(door):

    # Display message

       screen.fill((0, 0, 0))  # black background

       message = FONT.render("Monkey got the banana! Game Ends!", True, (255, 255, 0))

       screen.blit(message, (WIDTH//2 - message.get\_width()//2, HEIGHT//2 - message.get\_height()//2))

       pygame.display.update()

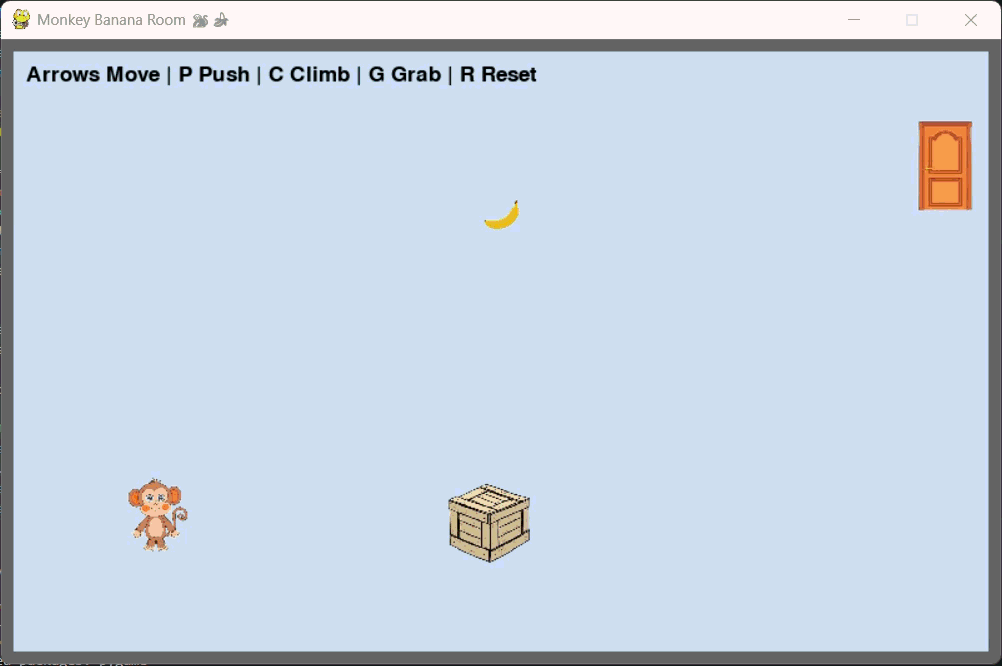
       pygame.time.wait(2000)  # wait 2 seconds

       pygame.quit()

       sys.exit()

    draw()

**Output**



.gif

Gameplay link: [GITHUB](https://github.com/k-artik-k/AI-lab-sem6/tree/main/Lab-3-MonkeyBanana/outputs).