**Task12 Use the Tkinter module for UI design.**

**12.1 Develop a Maze Game using Pygame**

To design and implement a simple Maze Game using Python’s Pygame library, where the player navigates a square through a maze to reach the goal without colliding with walls.

Algorithm:

1. Start the program.
2. Import required modules:
   * Import pygame for game development functions.
   * Import sys to exit the program cleanly.
3. Initialize Pygame using pygame.init().
4. Set up the game window:
   * Define screen width and height.
   * Create the display surface using pygame.display.set\_mode().
   * Set the window title using pygame.display.set\_caption().
5. Define colors using RGB values for white, black, blue, green, and red.
6. Create player and goal rectangles:
   * Player starts at (50, 50).
   * Goal is placed at (550, 350).
7. Create maze walls:
   * Define a list of rectangular wall obstacles using pygame.Rect().
8. Define a function check\_collision(rect, walls):
   * Check if the player’s rectangle collides with any wall.
   * Return True if a collision occurs, else False.
9. Main game loop:
   * Fill the background with white.
   * Handle QUIT events to close the game.
   * Detect keyboard input (arrow keys) for player movement.
   * Move the player and check for wall collisions.
   * If collision detected → undo the movement.
   * If player reaches the goal → display “You Win!” message and exit.
10. Draw game objects:
    * Draw walls (black), goal (green), and player (blue).
11. Update display using pygame.display.flip() and maintain frame rate with clock.tick(30).
12. Exit the game using pygame.quit() and sys.exit().
13. End the program.

**Program**

mport pygame

import sys

# Initialize pygame

pygame.init()

# Screen setup

WIDTH, HEIGHT = 600, 400

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

pygame.display.set\_caption("Maze Game")

# Colors

WHITE = (255, 255, 255)

BLACK = (0, 0, 0)

BLUE = (0, 0, 200)

GREEN = (0, 200, 0)

RED = (200, 0, 0)

# Clock

clock = pygame.time.Clock()

# Player setup

player\_size = 20

player = pygame.Rect(50, 50, player\_size, player\_size)

# Goal setup

goal = pygame.Rect(550, 350, player\_size, player\_size)

# Maze walls (list of rectangles)

walls = [

pygame.Rect(100, 0, 20, 300),

pygame.Rect(200, 100, 20, 300),

pygame.Rect(300, 0, 20, 250),

pygame.Rect(400, 150, 20, 250),

pygame.Rect(500, 0, 20, 250),

]

# Function to check wall collisions

def check\_collision(rect, walls):

for wall in walls:

if rect.colliderect(wall):

return True

return False

# Game loop

running = True

while running:

screen.fill(WHITE)

for event in pygame.event.get():

if event.type == pygame.QUIT:

running = False

keys = pygame.key.get\_pressed()

move\_x, move\_y = 0, 0

if keys[pygame.K\_LEFT]:

move\_x = -3

if keys[pygame.K\_RIGHT]:

move\_x = 3

if keys[pygame.K\_UP]:

move\_y = -3

if keys[pygame.K\_DOWN]:

move\_y = 3

# Move player

player.move\_ip(move\_x, move\_y)

# Collision with walls → undo move

if check\_collision(player, walls):

player.move\_ip(-move\_x, -move\_y)

# Win condition

if player.colliderect(goal):

font = pygame.font.SysFont("Arial", 28)

text = font.render("🎉 You Win!", True, RED)

screen.blit(text, (230, 180))

pygame.display.flip()

pygame.time.wait(2000)

running = False

# Draw walls

for wall in walls:

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

# Draw goal

pygame.draw.rect(screen, GREEN, goal)

# Draw player

pygame.draw.rect(screen, BLUE, player)

pygame.display.flip()

clock.tick(30)

pygame.quit()

sys.exit()

**Sample Output / Game Description:**

* The window displays a **maze** made of black walls.
* The **blue square** represents the player.
* The **green square** represents the goal.
* The player moves using **arrow keys**:
  + ⬆️ Up
  + ⬇️ Down
  + ⬅️ Left
  + ➡️ Right
* If the player touches a wall, the move is undone.
* When the player reaches the goal, a message **“🎉 You Win!”** appears for 2 seconds, and the game exits.