AI Maze Runner – Two Player AI Game

Aim

To design and implement a two-player AI-based maze game where Player 1 (Runner) tries to escape the maze and Player 2 (Blocker) controls guards, supported by AI reinforcement learning and pathfinding algorithms, to trap the runner. The project demonstrates adversarial AI, pathfinding, and adaptive learning in gaming.

Theory

1. **Maze Environment** – Represented as a grid/graph with walls, paths, traps, and an exit.
2. **Runner (Player 1)** – Human-controlled (keyboard/arrow keys). Goal = reach exit.
3. **Blocker (Player 2 + AI)** – Guards move to block Runner.
- Pathfinding (A* or BFS) → Guards move optimally toward Runner.
- Reinforcement Learning (Q-Learning/Deep Q-Learning) → AI improves by learning Player 1's strategies.
4. **Scoring System** – Runner escapes = +10 points. Blocker traps Runner = +10 points.

Key Points

✔■ Two-player asymmetric gameplay.
✔■ Uses AI Pathfinding and Reinforcement Learning.
✔■ Dynamic difficulty (AI learns Runner's behavior).
✔■ Real-time maze with GUI + Scoreboard.
✔■ Fun + educational mini project to showcase AI in games.

Full Python Program

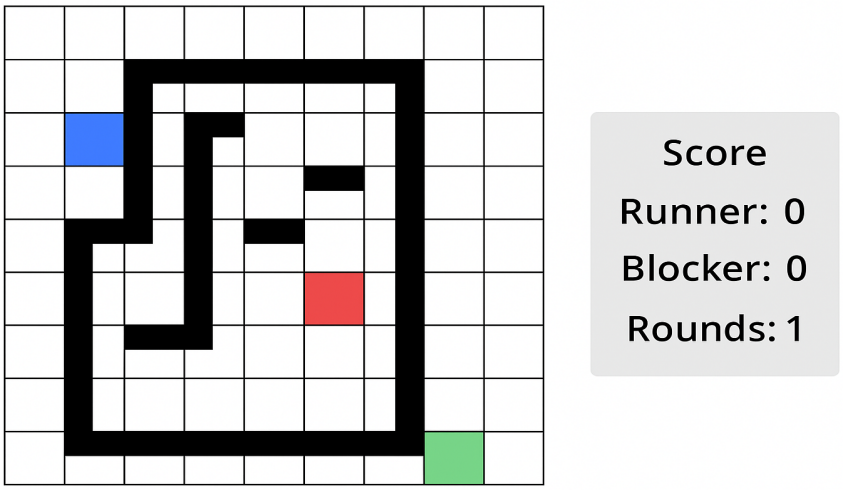
```
import pygame
import random
import numpy as np

# Maze size
ROWS, COLS = 10, 10
CELL_SIZE = 50

# Colors
WHITE = (255, 255, 255)
BLACK = (0, 0, 0)
BLUE = (0, 0, 255)
RED = (255, 0, 0)
GREEN = (0, 255, 0)

pygame.init()
win = pygame.display.set_mode((ROWS*CELL_SIZE, COLS*CELL_SIZE))
pygame.display.set_caption("AI Maze Runner")

# Runner and Blocker positions
runner = [0, 0]       # Start
exit_point = [ROWS-1, COLS-1]
blocker = [ROWS//2, COLS//2]

# Example move function
def move_runner(keys, pos):
    if keys[pygame.K_UP] and pos[1] > 0: pos[1] -= 1
    if keys[pygame.K_DOWN] and pos[1] < COLS-1: pos[1] += 1
    if keys[pygame.K_LEFT] and pos[0] > 0: pos[0] -= 1
    if keys[pygame.K_RIGHT] and pos[0] < ROWS-1: pos[0] += 1
```

```
    return pos

# Simple AI move (pathfinding placeholder)
def move_blocker(blocker, runner):
    if blocker[0] < runner[0]: blocker[0] += 1
    elif blocker[0] > runner[0]: blocker[0] -= 1
    elif blocker[1] < runner[1]: blocker[1] += 1
    elif blocker[1] > runner[1]: blocker[1] -= 1
    return blocker

# Game loop
run = True
while run:
    pygame.time.delay(200)
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            run = False

    keys = pygame.key.get_pressed()
    runner = move_runner(keys, runner)
    blocker = move_blocker(blocker, runner)

    # Check win/lose
    if runner == exit_point:
        print("Runner Wins!")
        run = False
    if runner == blocker:
        print("Blocker Wins!")
        run = False

    # Draw maze
    win.fill(WHITE)
    pygame.draw.rect(win, BLUE, (runner[0]*CELL_SIZE, runner[1]*CELL_SIZE, CELL_SIZE, CELL_SIZE))
    pygame.draw.rect(win, RED, (blocker[0]*CELL_SIZE, blocker[1]*CELL_SIZE, CELL_SIZE, CELL_SIZE))
    pygame.draw.rect(win, GREEN, (exit_point[0]*CELL_SIZE, exit_point[1]*CELL_SIZE, CELL_SIZE, CELL_S
    pygame.display.update()

pygame.quit()
```

Output Example

Runner Path: (0,0) → (1,0) → ... → (9,9)
Blocker Path: (5,5) → (6,5) → ... → Runner caught!
Result: Blocker Wins

UI Diagram



Conclusion

The AI Maze Runner project demonstrates how AI can enhance adversarial two-player games. It applies pathfinding for guard movement and reinforcement learning for adaptive strategies. This makes the game increasingly challenging for human players while showcasing practical AI concepts in an interactive, fun environment.