ARTIFICAL INTELLIGENCE

SNAKE GAME AI

KRISHNA P [RA2311042010017]
K MANTHRA {RA2311042010019]

PROBLEM STATEMENT

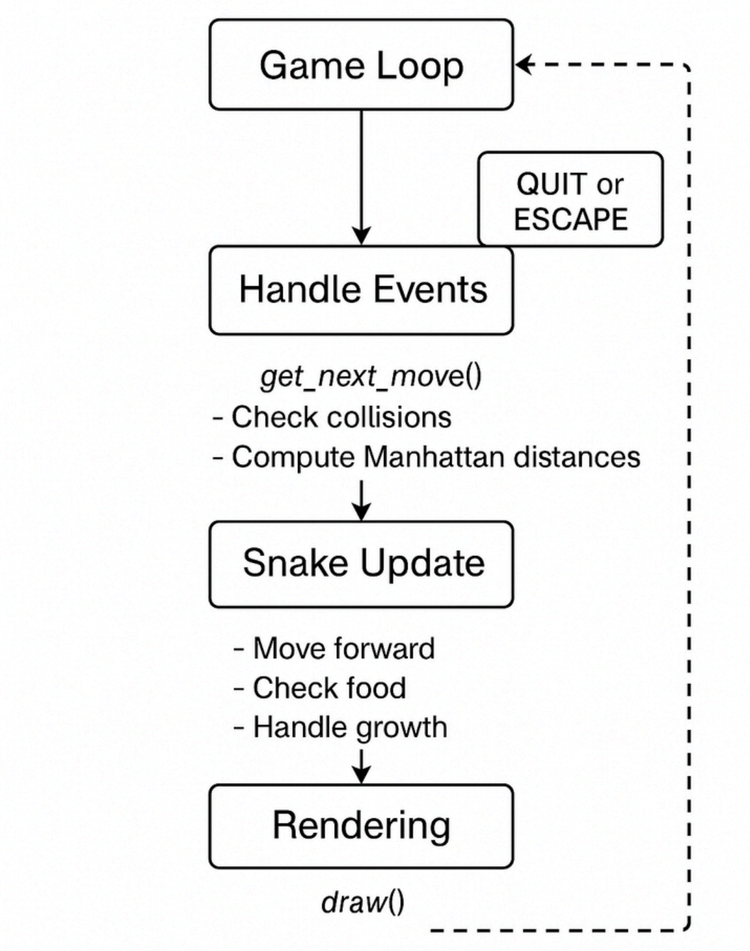
Design and implement a Snake Game with the following features:

- A grid-based game board.
- Random food placement.
- AI-controlled snake that finds the shortest path to food using a Greedy algorithm.
- Avoids walls and self-collision.

The system should:

- Display the snake and food in real-time.
- Move the snake automatically using AI decisions.
- Show the score based on food collected..

OVERALL ARCHITECTURE DIAGRAM



IMPLEMENTATION DETAILS

1. Technology Stack
Python 3
Pygame: For rendering graphics and managing game loop
Basic AI logic: Greedy algorithm using Manhattan distance for pathfinding

2. Core Components
Component Purpose
pygame.display Renders the game window
snake list Maintains the positions of all segments of the snake
food tuple Stores the food's location
direction Current direction of movement
get_next_move() AI logic to select the next optimal move
update_snake() Moves the snake forward, handles food consumption, and detects collisions

3. AI Behavior
Considers four directions: up, down, left, right
For each valid move, calculates the Manhattan distance to the food
Picks the move that minimizes the distance, avoiding collisions
Limitation: Doesn't consider dead ends, traps, or tail chasing (no path planning)

RESULTS -CODE

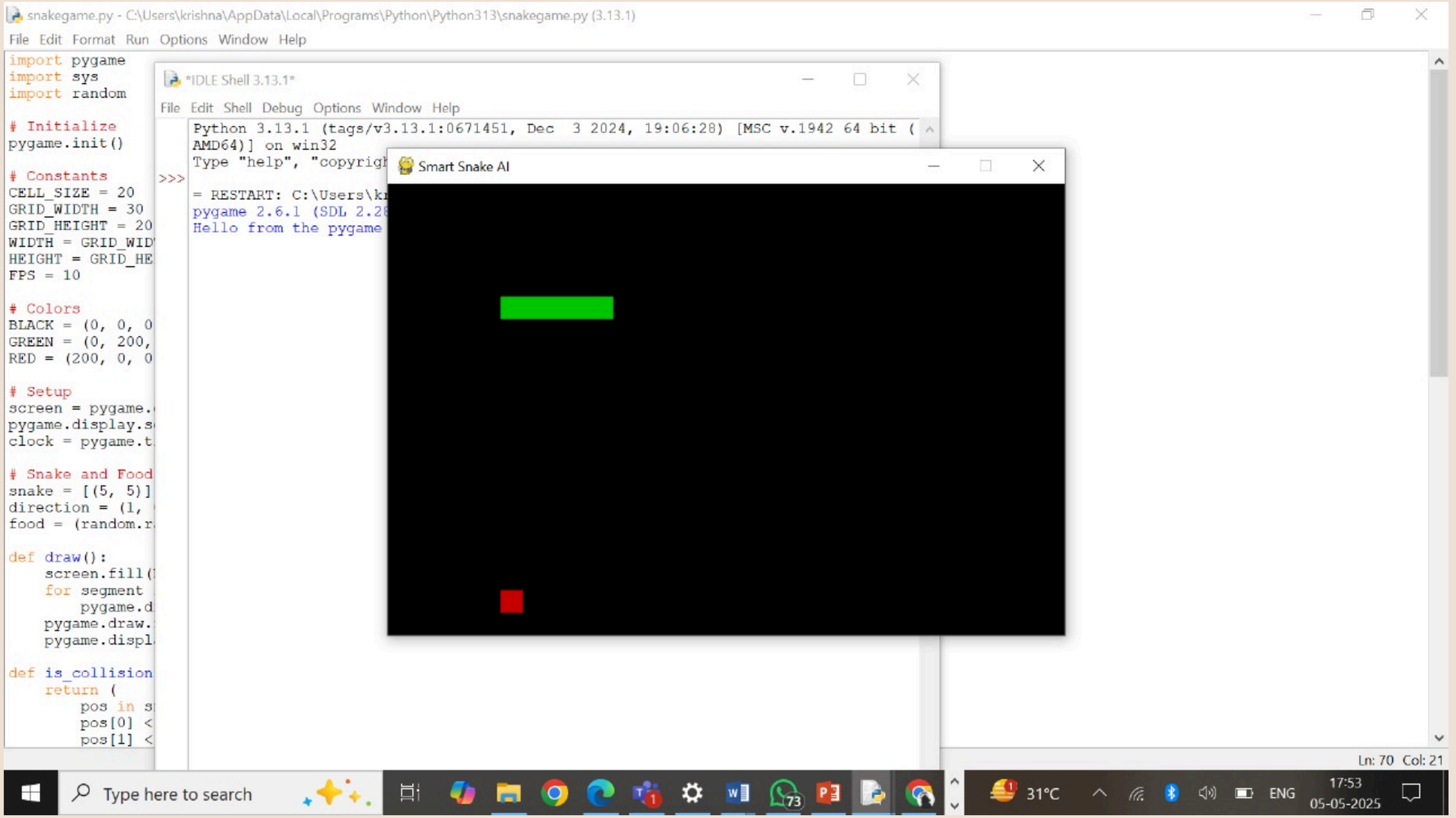
```
import pygame
import sys
import random

# Initialize
pygame.init()

# Constants
CELL_SIZE = 20
GRID_WIDTH = 30
GRID_HEIGHT = 20
WIDTH = GRID_WIDTH * CELL_SIZE
HEIGHT = GRID_HEIGHT * CELL_SIZE
FPS = 10

# Colors
BLACK = (0, 0, 0)
GREEN = (0, 200, 0)
RED = (200, 0, 0)

# Setup
screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("Smart Snake AI")
clock = pygame.time.Clock()

# Snake and Food
snake = [(5, 5)]
direction = (1, 0)
food = (random.randint(0, GRID_WIDTH-1), random.randint(0, GRID_HEIGHT-1))
```

CODE

```
def draw():
    screen.fill(BLACK)
    for segment in snake:
        pygame.draw.rect(screen, GREEN, (segment[0]*CELL_SIZE, segment[1]*CELL_SIZE, CELL_SIZE, CELL_SIZE))
    pygame.draw.rect(screen, RED, (food[0]*CELL_SIZE, food[1]*CELL_SIZE, CELL_SIZE, CELL_SIZE))
    pygame.display.flip()

def is_collision(pos):
    return (
        pos in snake or
        pos[0] < 0 or pos[0] >= GRID_WIDTH or
        pos[1] < 0 or pos[1] >= GRID_HEIGHT
    )

def manhattan_distance(a, b):
    return abs(a[0] - b[0]) + abs(a[1] - b[1])

def get_next_move():
    head = snake[0]
    options = [(1, 0), (-1, 0), (0, 1), (0, -1)]
    valid_moves = []

    for dx, dy in options:
        new_pos = (head[0] + dx, head[1] + dy)
        if not is_collision(new_pos):
            dist = manhattan_distance(new_pos, food)
            valid_moves.append((dist, (dx, dy)))

    if valid_moves:
        valid_moves.sort()
        return valid_moves[0][1]
    else:
        return (0, 0)  # No move possible
```

CODE

```
def update_snake():
    global food
    new_head = (snake[0][0] + direction[0], snake[0][1] + direction[1])

    if is_collision(new_head):
        pygame.quit()
        sys.exit()

    snake.insert(0, new_head)

    if new_head == food:
        food = (random.randint(0, GRID_WIDTH-1), random.randint(0, GRID_HEIGHT-1))
        while food in snake:
            food = (random.randint(0, GRID_WIDTH-1), random.randint(0, GRID_HEIGHT-1))
    else:
        snake.pop()

# Game Loop
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            sys.exit()

    direction = get_next_move()
    update_snake()
    draw()
    clock.tick(FPS)
```


snakegame.py - C:\Users\krishna\AppData\Local\Programs\Python\Python313\snakegame.py (3.13.1)
File Edit Format Run Options Window Help
import pygame
import sys
import random
# Initialize
pygame.init()
# Constants
CELL_SIZE = 20
GRID_WIDTH = 30
GRID_HEIGHT = 20
WIDTH = GRID_WID
HEIGHT = GRID_HE
FPS = 10
# Colors
BLACK = (0, 0, 0
GREEN = (0, 200,
RED = (200, 0, 0
# Setup
screen = pygame.
pygame.display.s
clock = pygame.t
# Snake and Food
snake = [(5, 5)]
direction = (1,
food = (random.r
def draw():
screen.fill(
for segment
pygame.d
pygame.draw.
pygame.displ
def is_collision
return (
pos in s
pos[0] <
pos[1] <
IDLE Shell 3.13.1
File Edit Shell Debug Options Window Help
Python 3.13.1 (tags/v3.13.1:0671451, Dec 3 2024, 19:06:28) [MSC v.1942 64 bit (
AMD64)] on win32
Type "help", "copyrigh
>>>
= RESTART: C:\Users\k
pygame 2.6.1 (SDL 2.2
Hello from the pygame
Smart Snake AI
Ln: 70 Col: 21
Type here to search
31°C
ENG
17:53
05-05-2025

DEMO

https://drive.google.com/file/d/1zrzIMjTL5EkpF_JnhsBTPaR11Ez4lKDe/view?usp=sharing

Thank you