15-10 Task: 12 ... Simulate Gaming concepts using pygame

Aim: To simulate Gaming concept using pygame

problem 1. Write a program to create a snake game

conditions:

1. Set the window size
2. create a snake
3. make the snake to move in the Directions
4. when the snake hits
5. If the snake hits the window. Game

Algorithm

1. Import pygame package and initialize
2. Define the window size and title
3. create a snake class which initialize the snake position
4. create a Fruit class which initializes
5. create a function to check if the snake collides
6. create a function to check If the snake collides

Program

```
import pygame
import time
import random
snake_speed = 15
window_x = 720
window_y = 480
black = pygame.color(0,0,0).
```

output

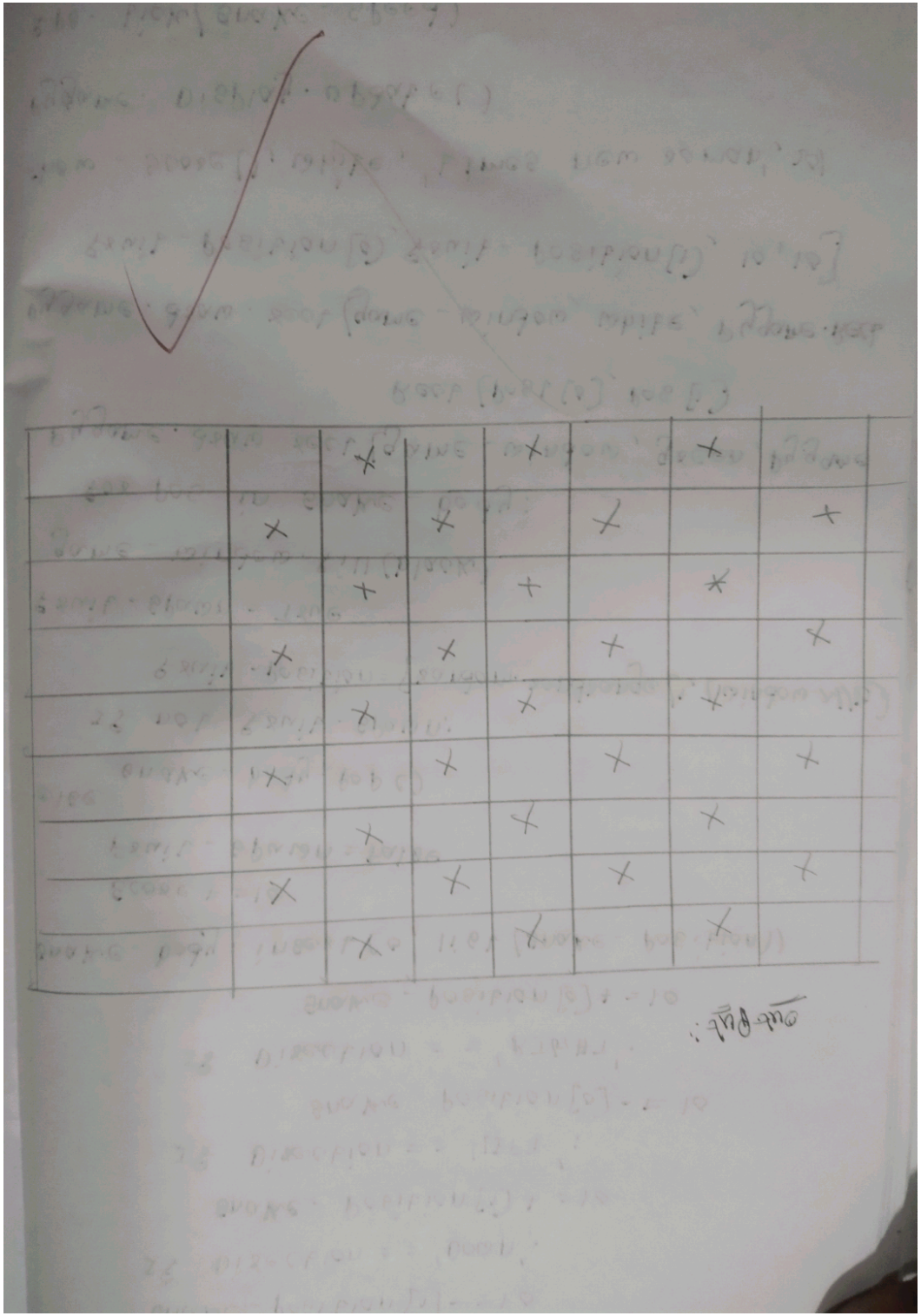
source = 0.

```
white = pygame.color(255, 255, 255)
red = pygame.color(255,0,0)
green = pygame.color(0,255,0)
blue = pygame.color(0,0,255)
pygame.init()
pygame.display.set_caption('GeeksforGeeks snakes')
game_window = pygame.Display.set_mode
fps = pygame.time.clock()
snake_position = [100, 50]
snake_body = [[100,50],
              [90,50],
              [80,50]
              [70,50]
fruit_position = [random.randrange(1, window_x//10)) * 10
                  random.randrange(1, (window_y//10)) * 10]

fruit_spawn = True
direction = 'RIGHT'
change_to = direction
score = 0
def show_score(choice, color, font, size);
score_font = pygame.font.SysFont(font, size)
score_surface = score_font.render('score:' + str(score)
score_rect = score_surface.get_rect()
game_window.blit(score_surface, score_rect)
def game_over();
```

```
my_font = pygame.font.SysFont('times new roman', 50)
game_over_surface = my_font.render(
    'your score is: ' + str(score), True, red)
game_over_rect = game_over_surface.get_rect()
game_over_rect = game_over_surface.get_rect()
game_over_rect.midtop = (window_x/2, window_y/4)
game_window.blit(game_over_surface, game_over_
                 rect)
time.sleep(2)
pygame.quit()
while True:
for event in pygame.event.get():
  If event.type == pygame.KEY DOWN:
If event.key == pygame.K_UP:
        change_to = 'UP'
    If event.key == pygame.K_DOWN:
        change_to = 'Down'
    If event key == pygame.K_LEFT:
        change_to = 'LEFT'
    If event.key == pygame.K_RIGHT:
        change_to = 'RIGHT'
If change_to == 'UP' and Direction != 'DOWN':
    Direction = 'UP'
If change_to == 'DOWN' and Direction != 'UP':
    Direction = 'DOWN'
If change_to == 'LEFT' and Direction != 'RIGHT':
    Direction = 'LEFT'
```

```
If change_to == 'RIGHT' and Direction != 'LEFT':
        Direction = 'RIGHT'
If Direction == 'up':
    snake_position[1] -= 10
    If Direction == 'down':
        snake_Position[1] += 10
    If Direction == 'LEFT':
        snake_position[0] -= 10
    If Direction == 'RIGHT':
        snake_position[0] += 10
snake_body.insert(0, list(snake_position))
    score += 10
    Fruit_spawn = False
else
    snake_body.pop()
    If not fruit_spawn:
    fruit_position = [random.randrange(1, (window_x//10)
fruit_spawn = True
game_window.fill(black)
  for pos in snake_body:
Pygame.draw.rect(game_window, green, Pygame
                    Rect(Post[0], pos[1]
Pygame.draw.rect(game_window, white, Pygame.Rect
    fruit_position[0], fruit_position[1], 10, 10]
show_score(), white, 'times new roman', 20
Pygame.Display.update()
    ke_speed)
```



Problem 12.2 write a python program to Develop a chess board using Pygame

Algorithm

1. Import Pygame and initialize
2. Set screen size and title
3. Define colors for the board and pieces
4. Draw the board and piece on the screen
5. Start the game loop

Program

```
import pygame
pygame.init()
screen_size = (640, 640)
screen = pygame.display.set_mode(screen_size)
pygame.display.set_caption('Chess Board')
black = (0,0,0)
white = (255, 255, 255)
brown = (153, 76, 0)
def draw_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)
```

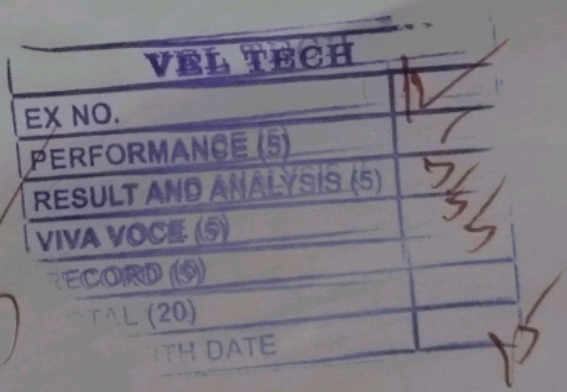
```
def draw_piece(board):
    piece_images = {
        'r': pygame.image.load('images/rook.png'),
        'n': pygame.image.load('images/knight.png'),
        'b': pygame.image.load('images/bishop.png'),
        'q': pygame.image.load('images/queen.png'),
        'k': pygame.image.load('images/king.png'),
        'p': pygame.image.load('images/pawn.png'),
        'R': pygame.image.load('images/white_rook.png'),
        'N': pygame.image.load('images/white_knight.png'),
        'B': pygame.image.load('images/white_bishop.png'),
        'Q': pygame.image.load('images/white_queen.png'),
        'K': pygame.image.load('images/white_king.png'),
        'P': pygame.image.load('images/white_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)

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()
draw_pieces(board)
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
```

completed



Result: Hence writing a Python program to Develop a chess Board using Python program is Done successfully