Task 12:- Simulate Gaming Concepts using pygame

Aim:- To simulate gaming concepts using python.

SnakeGame:-

Problem:- write a python 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 when left, right, down and up is pressed
4. If the snake hits the windows, Game over.

Algorithm:-

1. Import pygame package and initialize it.
2. Define the window size and title
3. Create a snake class which initalites the snake position, color and movement.
4. Create a function to check if the snake collides with the fruit and increases the score.
5. Create a game loop to continously update the game display, snake Position and check for collisions
6. End the game if the user quits or the snake collides with the window

Program:-

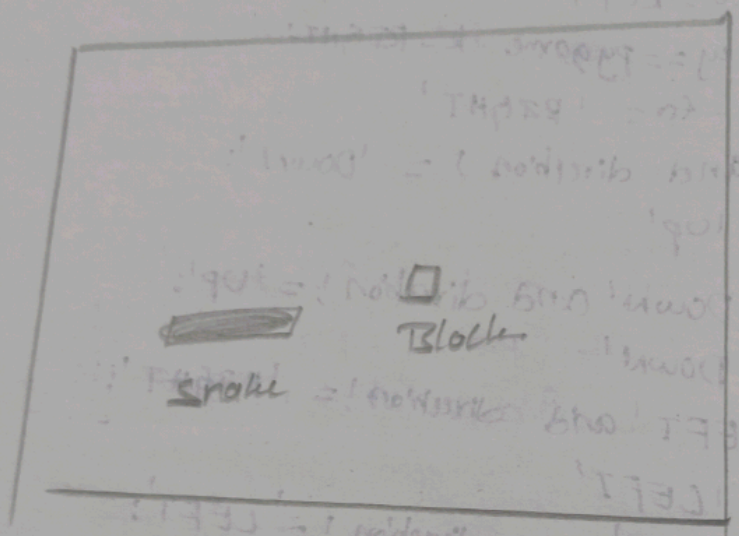
```
import pygame
import time
import random
snake_speed = 15
window_x = 720
window_y = 480
black = pygame.Color(0,0,0)
white = pygame.Color(255,255,255)
red = pygame.Color(255,0,0)
green = pygame.Color(0,255,0)
blue = pygame.Color(0,0,255)
```

```
pygame.init()
pygame.display.set_captain('Geeksforgeeks snakes')
game_window = pygame.display.set_mode((window_x, window_y))
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_position = 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), True, color)
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.midtop = (window_x/2, window_y/4)
game_window.blit(game_over_surface, game_over_rect)
pygame.display.flip()
time.sleep(2)
pygame.quit()
```

```
quit()
while true:
for event in pygame.event.get():
    if event.type == pygame.KEYDOWN:
        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))
if snake_position[0] == fruit_position[0] and snake_position[1]
    == fruit_position[1]:
        score += 10
        fruit_spawn = false
else:
    snake_body.pop()
if not fruit_spawn:
    fruit_position = [random.randrange(1, (window_x//10))*10,
                      random.randrange(1, (window_y//10))*10]
fruit_spawn = true
game_window.fill(black)
for pos in snake_body:
    pygame.draw.rect(game_window, green)
                pygame.Rect(pos[0], pos[1], 10, 10)]
pygame.draw.rect(game_window, white, pygame.Rect(
    fruit_position[0], fruit_position[1], 10, 10))
if snake_position[0] < 0 or snake_position[0] > window_x-10:
    game_over()
if snake_position[1] < 0 or snake_position[1] > window_y-10:
    game_over()
for block in snake_body[1:]:
    if snake_position[0] == block[0] and snake_position[1] ==
                                        block[1]:
        game_over()
show_score(1, white, 'times new woman', 20)
pygame.display.update()
fps.tick(snake_speed)
```

output



2. write a python program to Develop a chess board using pygame.

Algorithm:-

1. Import pygame and initialize it
2. set screen size and title
3. Define colors for the board and pieces.
4. Define a function to draw the pieces on the board by loading images for each piece and placing them on the corresponding square.
5. Define the initial state of the board as a list of lists containing the pieces.
6. 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_pieces(board):
    pieces_images = {
        'r': pygame.image.load('image/rook.png'),
        'n': pygame.image.load('images/bishop.png'),
```

output :-

```
'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')

}
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','N','B','R']
]
draw_board()
draw_pieces(board)
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            quit()
    pygame.display.update()
```

VEL TECH	
EX No.	12
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	5
RECORD (5)	10
TOTAL (20)	
SIGN WITH DATE	8/10

Result: Thus, the program for pygame is executed and verified successfully.