Task-12:- Simulate Gaming Concepts using Pygame

Aim:- To simulate gaming Concept using python

Snake Game:-

Problem:- Write a python program to create a snake Game

Conditions:-

1. Set the windowsize
2. Create a snake
3. Make the snake to make in the directions when left, right down and up, it presed.
4. If the Snake bits 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 initialize the snake position Colour 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 without position and Check for collisions.
6. End the some if the user quits or the Snake Collides with the window

Program:-

```
Import pygame
import time
import random
snake_speed = 15
window_x = 750
window_y = 480
black = pygame.colour(0,0,0)
White = pygame.colour(255,255,255)
red = pygame.colour(255,0,0)
green = pygame.colour(0,255,0)
blue = pygame.colour(0,0,255)
```

```
pygame.init()
Pygame.display.set_caption('Snakes for Cracks 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/20))*10]
fruit_position = True
direction = 'RIGHT'
Change_to = direction
Score = 0.
def show_score(choice, colour, 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)
Py.game.display.flip()
time.sleep(2)
Pygame.quit()
```

```
While fpve:
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
```

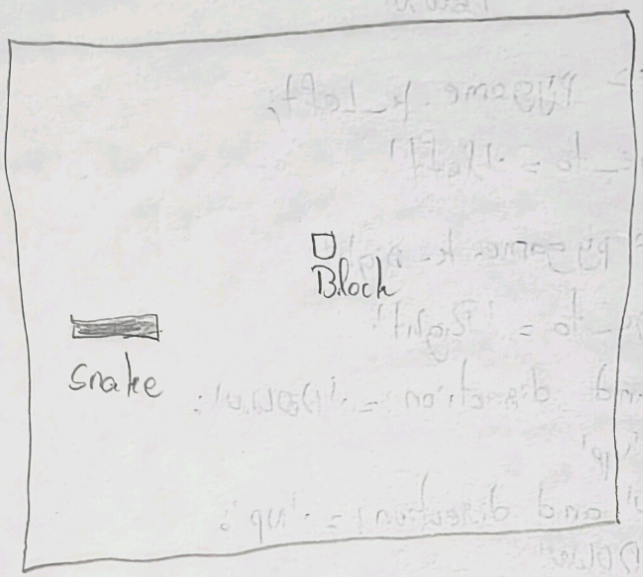
```
else:
    snake_body.pop()
if not fruit_spawn:
    fruit_position = [random.randrange(1,(window_x//10))*10,
    random.randrange(1,(window_y//10))*10]
fruit_position = [random.randrange(1,(window_x//10))*10,
for pos in snake_boby:
    pygame.draw.rect(game_window, green)
        pygame.Rect[pos[0], pos[1], 10, 10])
py.game.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 roman', 20)
Pygame.display.update().
fps.tick(snake_speed)
```

2) Write a python program to Develop a chess board using Py game.

Algorithm:-

1. Import pygame and Initialize it
2. Set screen size and title
3. Define Classes for the board and pieces.
4. Define a function to draw the pieces on the board by loading image for each piece and placing them on the

Output :—



corresponding square

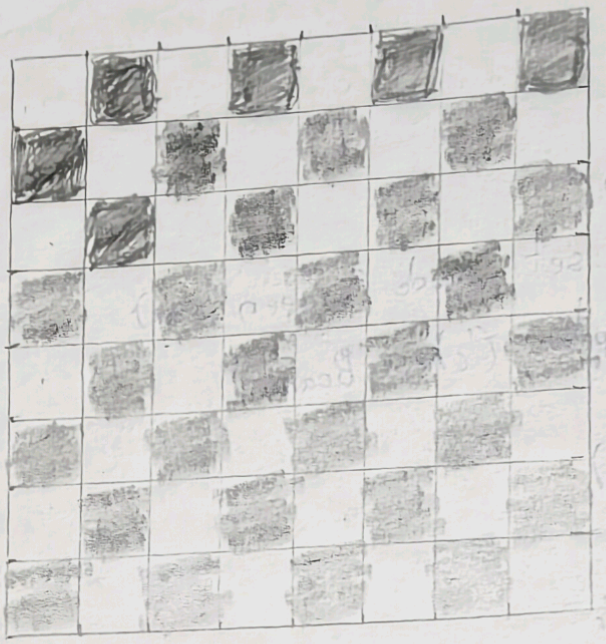
5. Define the initial state of the board as a list of list containing the pieces.

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
square_rect = pygame.Rect (col * 80, row * 80, 80, 80)
Pygame.draw.rect (screen, color, square_rect)
def draw_pieces (board):
    pieces_images = {
        'r': pygame.image.load ('image/rook.png'),
        'n': pygame image load ('images/bishop.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/pown.png'),
    }
    for row in range (8):
        for col in range (8):
```

Output



```
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 = [
    ["bR", "bN", "bB", "bQ", "bK", "bB", "bN", "bR"],
    ["bp", "bp", "bp", "bp", "bp", "bp", "bp", "bp"],
    ["--", "--", "--", "--", "--", "--", "--", "--"],
    ["--", "--", "--", "--", "--", "--", "--", "--"],
    ["--", "--", "--", "--", "--", "--", "--", "--"],
    ["--", "--", "--", "--", "--", "--", "--", "--"],
    ["wp", "wp", "wp", "wp", "wp", "wp", "wp", "wp"],
    ["wR", "wN", "wB", "wQ", "wK", "wN", "wB", "wR"]
]
draw_board()
draw_pieces(board)
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            quit()
pygame.display.update()
```

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

VEL TECH	
EX No.	12
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	5
RECORD (5)	5
TOTAL (20)	15
SIGN WITH DATE	