Task 12:- Simulate Gaming Concepts using PYGame

Aim:- To Simulate Gaming Concepts using Pygame

SnackGame:-

Problem 12.1 :- Write a Python Program to Create a Snake Game using Pygame Package.

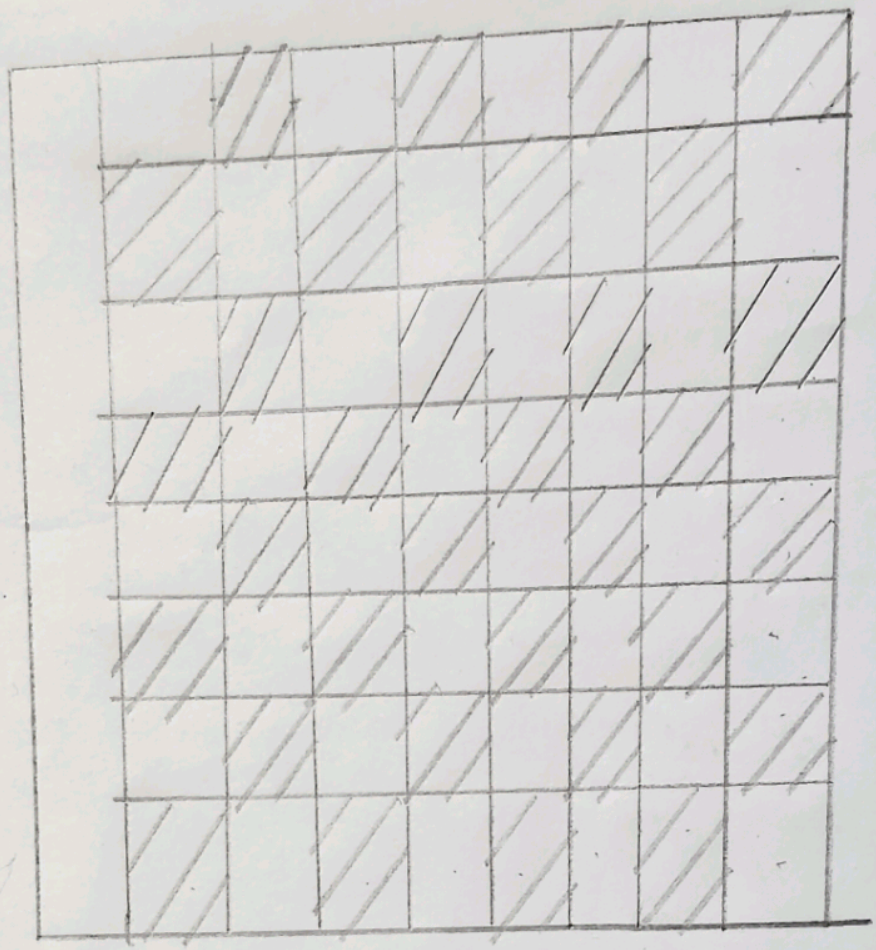
Conditions:-

1. Set the window size.
2. Create a snake.
3. Make the snake to move in directions when right, left, down and up key is pressed.
4. when the snake hits the Fruit. increase the score by 10
5. if snake hits the window. Game over

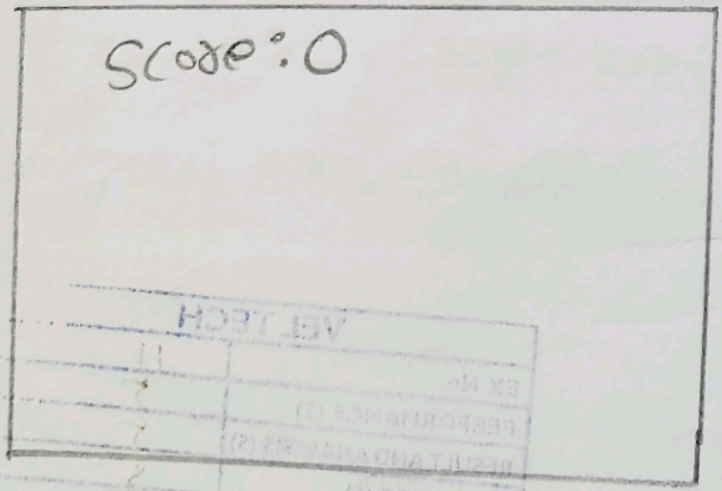
Algorithm:-

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 Fruit class which initializes the Fruit position and color.
5. Create a Function to check if snake collides with the Fruit and increase the score.
6. Create a Function to check if the snake collides with window and end the game.
7. Create a Function to update the snake position based on user input.
8. Create a Function to update the game display and draw the Snake and Fruit.
9. Create a game loop to continuously update the game

Sample output :-



Output :-



display, snake position, & check for collisions.
10. End the game if the user quits (or) the snake collides with the window.

Program:-

```
# importing libraries
import pygame
import time
import random

snake_speed = 15

# window size
window_x = 720
window_y = 480

# defining colours
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)

# intialise game window
pygame.display.set_Caption('Geeks For Geeks Snakes')
game_window = pygame.display.set_mode(window
                                      _x, window_y))

# FPS (Frames per second) controller
fps = pygame.time.Clock()

# defining first 4 blocks of snake body
snake_body = [100, 50]
              [90,50],
              [80,50],
              [70,50]
```

```
#Fruit position
fruit_position = [random.randrange(1,(window_x//10)*10,
                  random.randrange(1,(window_y//10)*10]

fruit_spawn = True

#setting default snake direction towards
#right
direction = 'RIGHT'
change_to = direction

#initial score
score = 0

#displaying score function
def show_score(choice, color, font, size):

#deactivating pygame library
pygame.quit()

#quit the program
quit()

#main function
while True:

    #handling key events
    for event in pygame.event.get():
        if event.type == pygame.event.get():
            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'
```

```
    direction = 'RIGHT'

# Moving the Snake
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 growing mechanism
#if fruits and snakes collide then scores
#will be incremented by 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))
```

```
# Game over conditions
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()
# Touching the snake body
for block in snake_body[1:]:
    if snake_position[0]==block[0] and snake_
        position[1] == block[1]:
    game_over()
# displaying score continuously
show_score(1, white, 'times new roman', 20)

# Refresh game screen
pygame.display.update()

# Frame per second /Refresh Rate
fps.tick(snake_speed)
```

Problem 12.2 :- 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 colours for board and pieces.

Define a function to draw the board by looping over rows and columns and drawing squares of different colors.

4. Define a function to draw the pieces on board by loading images for each piece and placing them on corresponding square.

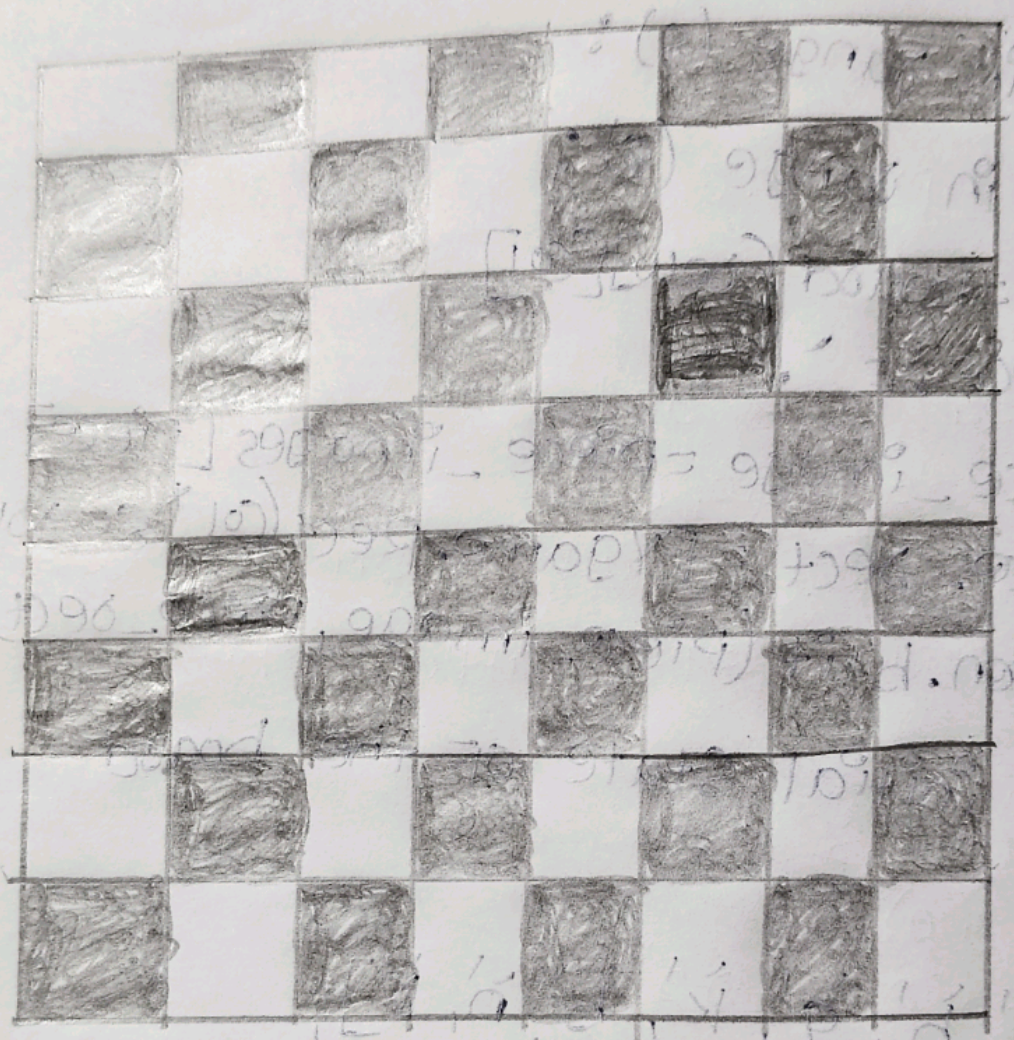
5. Define the initial state of the board as a list of lists containing the pieces.
6. Draw the board and pieces on screen.
7. Start the game loop.

Program:-

```
import pygame

# initialize pygame
pygame.init()

# set screen size and title
screen_size = (640, 640)
screen = pygame.display.set_mode(screen_size)
pygame.display.set_caption('Chess Board')

# Define colors
black = (0, 0, 0)
white = (255, 255, 255)
brown = (153, 76, 0)

# Define function to draw the board
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.image(square_color, square_
                              rect)

# Define function to draw the pieces
def draw_board():
    piece_images = {
        'r': pygame.image.load('images/rook.png'),
        'n': pygame.image.load('images/knight.png'),
        'b': pygame.image.load('images/bishop.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)

# Define initial state of the board
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 and pieces
draw_board()
draw_pieces(board)

# Start game loop
while True:
    for event in pygame.event.get():
```

output :-



```
    if event.type == pygame.QUIT:
        pygame.quit()
        quit()

    pygame.display.update()
```

VELTECH	
EX No.	12
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	5
RECORD (5)	5
TOTAL (20)	20
SIGN WITH DATE	15/10

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