Task 12 : Simulate Gaming concepts using pygame

Aim: To simulate Gaming concepts using Pygame

Snake Game:

Problem 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 the directions when left, right, down & up key is pressed
4) When the snake hits the fruit increase the score by 10
5) If the snake ~~the~~ window Game over

Algorithm

1) Import pygame package & initialize it
2) Define the window size and title
3) create a fruit class which initializes the snake position color and movement
4) create a fruit class which initializes the fruit position and color
5) Create a function to check if the snake collides with the fruit and increase the score
6) Create a function to check if the snake collides with the window and end the game.
7) Create a function to update the snake position based on the user input

Program:

```
#importing libraries
import pygames
import time
import random
snake_speed = 15
```

```
# window size
window_x = 720
window_y = 480

# defining colors
  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)

# Initialising Pygame
 pygame.init()

# Initialise game window
pygame.display.set_Caption('Geeksfor Geeks Snakes')
pygame.display.set_mode((window_x,window_y))
 game_window = pygame.display.set_mode

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

# defining snake default position
  snake_position = [100,50]
 # defining first 4 blocks of snakebody
      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):
# creating font object score_font
  score_font = pygame.font.

# create the display surface object
# score_surface

score_surface = score_font.render('score:' + str(score), True, (0,0,0))
# create a rectangular object for the text
# surface object
score_rect = score_surface.get_rect()

# displaying text
game_window.blit(score_surface, score_rect)
# game over function
  def game_over():
# creating a text surface on which text
# will be drawn
game_over_surface = my_font.render(
  'your score is:' + str(score), True, red)
# create a rectangular object for the text
# surface object
game_over_react = game_over_surface.get_rect()
# setting position of the text
game_over_rect.midtop = (window_x/2, window_y/4)
# blit will draw the text on screen
  game_window.blit(game_over_surface, game_over_rect)
  pygame.display.flip()
# after 2 seconds we will quit the program
  time.sleep(2)
# deactivating pygame library
  pygame.quit()
```

```
while True:

# handling key events
 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 two keys pressed simultaneously
# we don't want snake to move into two
# directions simultaneously.
 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'

 Fruit_Spawn = True
  game_window.fill(black)
  for pos in snake_body:
   pygame.draw.rect(game_window, green,
     pygame.Rect(post[0], post[1], 10, 10))

 pygame.draw.react(game_ window, white, Pygame.Rec
# 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 persecond/ Refresh Rate
 fps. tick (snake_speed)
```

output

Score : 0

Aim:
Write a python program to Develop a chess board using Pygame

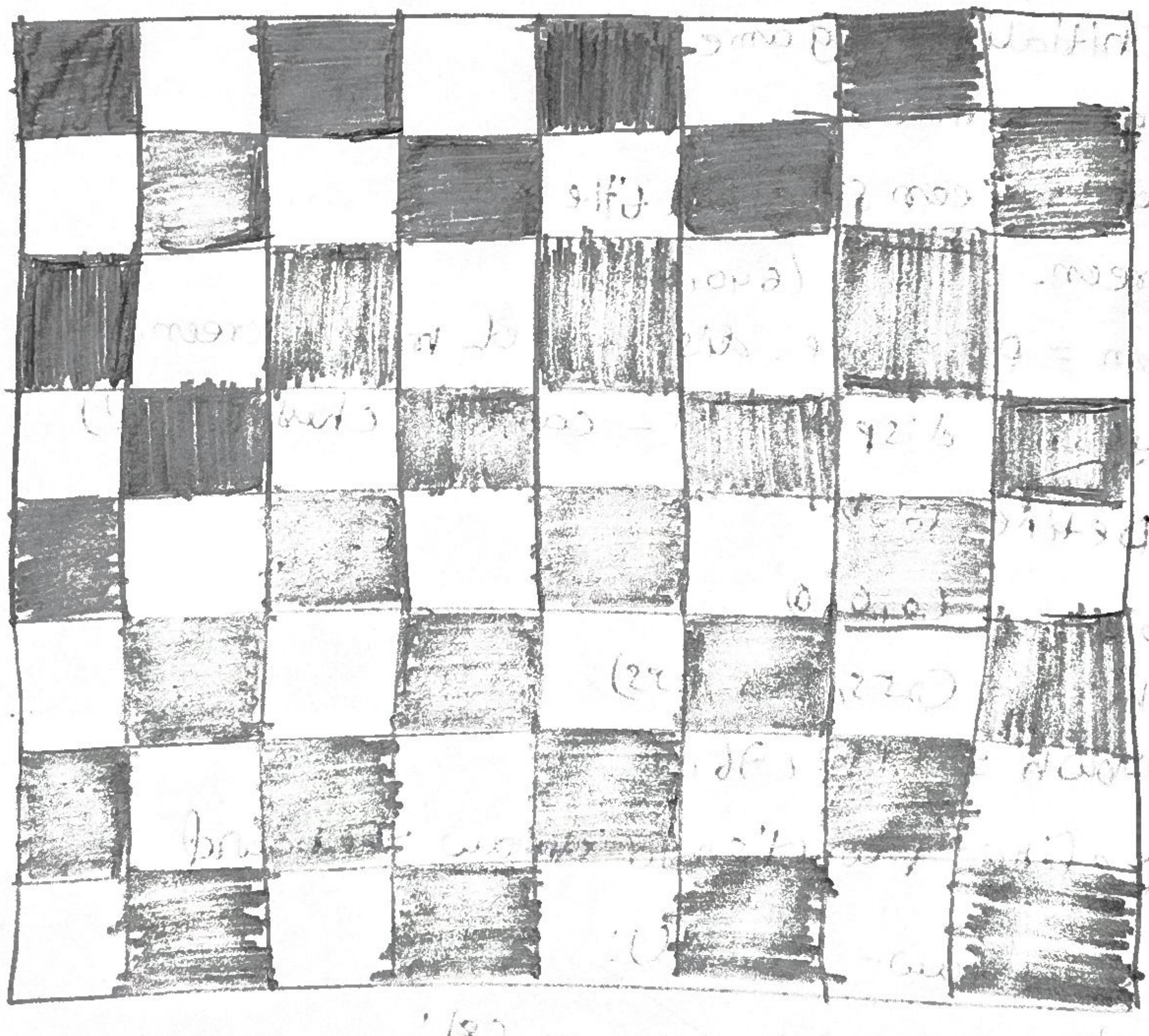
Algorithm:

1) Import pygame & initialize it
2) set screen size & title
3) Define colors for the boards and pieces
4) Define a function to drawn by looping over rows
5) Define the initial state of the board as list pieces
6) Draw the board & pieces on the 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_ rect = white if (row+col)%2)==0
     square - rect = py game. Rect(col*80, row*80,80,80)
   pygame draw. rect(screen, square color, square - rect
# Define function to draw the pieces
      piece_ images =
      'r' = pygame.image.lod ('images/rook.png)
      'n' = pygame.imageload('images/knight.png')
      'b' = pygame. imageload ('images/bishop.png')
```

Output



```
}
for row in range(8):
    for col in range(8)
        piece = board [row][col]
            if piece :=
            price - 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'],
        ['R', 'N', 'B', 'Q', 'K', 'B', 'N', 'R'].]
# Draw board and pieces
     draw - board ()
     draw - pieces (board)
   # Start game loop
   while True:
      if event. type== pygame QUIT:
      pygame. display. update()
```

completed

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

15

Result:

Thus the programe is executed and verified succesfully