Task 12: Simulate Gamming concepts using Pygame.

Aim: To simulate Gaming Concepts using pygame

Program 1. write a python program to create a snack Game using pygame package.

Conditions:

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

Algorithms

1. Import pygame pakage and initialize it.
2. Define the window size and title.
3. Create a snake class which initializes the snake position, Colour, and movement
4. create a fruit class which initializes the fruit position and Colour.
5. Create a function to check if the snake collides with the window or end the game
6. create a function to update the game display and draw the snake and fruit
7. Creat a game loop to continously update the game display, snake position, and check for collisions
8. 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.Colour (0,0,0)
white = pygame . colour (255, 255, 255)
red = pygame. Colour (255, 0,0)
green = pygam . colour (0,255,0)
blue = pygame . Colour (0,0,255)
pygame.init()
pygame.display.set_caption ('Greek for Geeks snakes')
game_window = pygame . display.set_mode ((window_x, window_y))
fps = pygame . time Clocks()
snake_body = [[100,50]], [90,50), [80,50], [70, 50]]
fruit_position = [random.randrange [1, (window_x /10)] *10 ,random.
                  randorange [1,[window_y //10]) *10]

  fruit_spawn = 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,
                                     colour)
  Score_rect = score_surface.get_rect()
  game_over():
 my_font = pygame.font. sysfont ('times new roman', 50)
 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)
  pygamedisplay.flip()
```

```
time.sleep(2)
py.game.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_Left
        change_to = 'LEFT'
     if event.key = = pygame.K_Right:
        change_to = 'RIGHT'
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)) *
                   random.randrange (1, (window_y//10)) * 10]
if snake - position [0] < 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).
```

Problem 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 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 initialize state of the board as a list of lists containing the pieces
6. Draw the board and pieces on the screen
7. 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 =(225,225,225)
brown =(153, 76,0)
def draw_board():
 for row in range (8):
   for col in range (8):
     square_colour = white if (row+col) % 2 == 0 else brown
     square_rect = pygame.Rect (col*80, row*80, 80,80)
     pygame.draw.rect.(screen, square_colour, square_rect)
 def draw_pieces (board):
     piece_image = {
       'r' = pygame.image.load ("image/rook.png"),
{
for row in range (8):
  for col in range (8):
    piece = board [row][col]
    if piece != '.':
    piece - image = piece - image [pieces]
   piece - rect = pygame.Rect [col*80 , row*80 , 80,80]
   screen.blit (piece - image , piece - rect)
board = [
  ['r','n' ,'b' ,'q' ,'k' ,'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()
   draw - pieces (board)
   while True:
    for event in pygame . event.get():
      if event. type = = pygame. QUIT:
       pygame . quit()
        quit()
  pygame.display . update()
```

Completed

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

Result: Thus the program for Pygame is executed and verified succesfully