Task12 - Simulate Gaming concepts using Pygame

Aim:- To simulate Gaming concepts using Pygame.

Snake Game:

Problem1: write a Python program to create a snake game using Pygame package.

Condition:

1. Set the window size
2. create a snake
3. make the snake to move in the directions the when left, right, down and upkey is pressed.
4. When the snake to move in the lits the fruit. increase the score by 10.
5. If the snake hits the window. Game over.

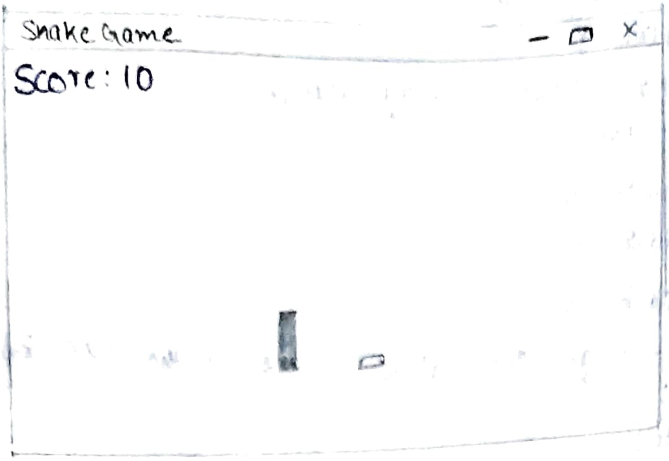
Algorithm:

1. Import Pygame package and initialize it.
2. Define the window size and title
3. Create a snake class which initialize the snake position, color and movement.
4. create a function to check if the snake collides with the fuit and increases the score.
5. Create a game loop to continnonsly 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.

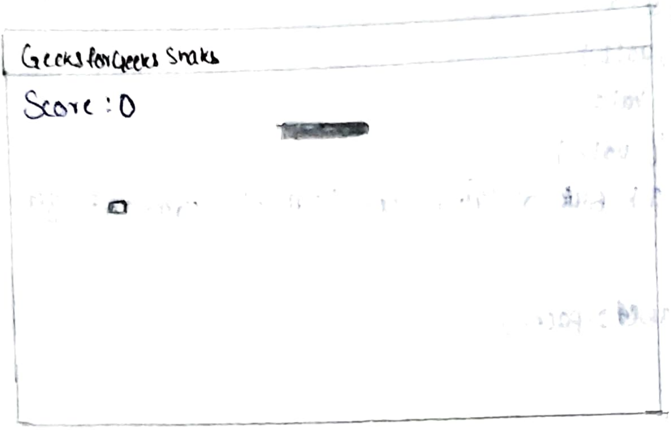
program:-

```
#importing libraries
 import pygame
 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('Geeks for Geeks snakes')
game_window=pygame.display.set_mode((window_x,window_y))
#FPS (frames per second) controller
  fps=pygame.time.clock()
```

Sample output:-



Output:

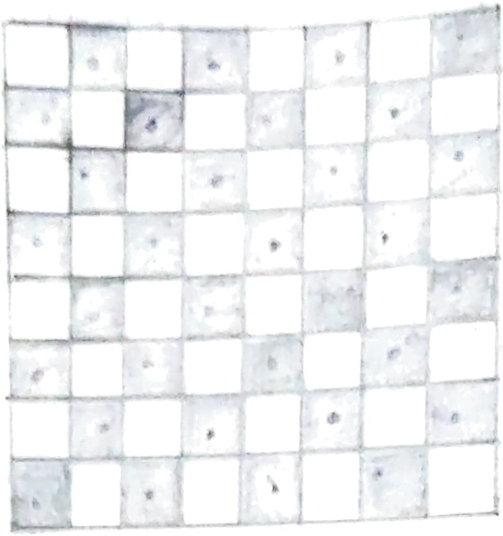


```
snake-position =[100, 50]
snake_body = [[100, 50],
             [90,50],
             [80,50],
              [70,50],
               ]
fruit-position = [random.randrange(1,(winmow_x//10)*10,random.rand
                range(1,(window_y//10)*10]
fruit-Position = True
direction ='RIGHT'
  change_to = direction
  Score = 0
def Snow_Score(choice,color, font,size):
#creating font obJects score_font
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)
 # game over function.
def game_over():
# creating font obJect my_font
my_font = Pygame.font.SysFont ('times new romah', 50)
 game_over_surface =my_font.render(
      'your score is: '+str(score), True, red)
game_over_rect=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  seconds we will quit the program
 time.sleep(2)
# deativating Pygame library
Py game quit()
#quit the program
quit()
# main function
while True:
# handling key events
for event in Pygame.event.get():
    if even.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_RIGHT:
            change_to= 'RIGHT'
```

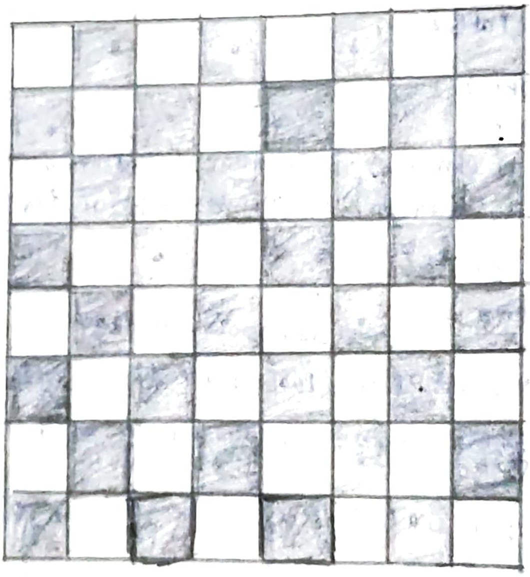
```
if change_to = = 'up' and direction! = 'DOWN':
    direction = 'UP'
if change_to = = 'DOWN' and direction ! = 'UP':
    direction = 'DOWN'
if change_to = = 'LEFT' and direction! = 'RIGHT':
    direction = 'LEFT'
if change_to = = 'RIGHT' and direction ! = 'LEFT':
    direction = 'RIGHT'
if direction = = '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 growing mechanism
# 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(110)) * 10, random.
                      randrange (1, (window -y (110)) * 10]
fruit_spawn = True
game_window. fill (black)
for pos in snake_body:
    py game. draw. rect (game_window, green)
                  py game. Rect (pos[0], pos[1], 10,10,)
py game. draw. rect (game_window, white, py game. 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)
py game. display. update( )
# Frame per second | Refresh Rate
fps. tick (snake_speed)
```

Problem 2:

Sample Output:



Output:-



Problem2 : write a python program to Develop a chess board using Pygame.

Algorithm:

```
1. Improt 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
# initialize Pygame
 py game.intl )
# set screen size and title
screen_size = (649 640)
 screen = Pygame. display. set_mode(screen_size)
 py game. display. set_ caption ('chess Board')
# Define colors
 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_cdor = white if (row + col )%2 == 0else brown
            square -rect = Pygame. Rect (col*80, row * 80, 80,80)
            Pygame.drawn. rect (screen -color, square _rect )

def drawn_ Pieces (board):
    piece _images ={
     (r) : py game. image. load (image /rook. png'),
     'k' : Py game. image. load ('image /knight.png'),
     'b' : Py ~~game: image~~. load ('images / bishop.png'),
     'q' : Py game. image .load ('images / queen. png');
     'k': py game. image. load ('images / king. Png'),
     'p': Pygame .image. load (images /pawn.png')
 }
```

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