Task 12: Simulate Gaming concepts using Pygame

Aim: To simulate Gaming concepts using Pygame

Snack Game:

problem ~~Game 12~~ 1:- write a python program to create a snake Game using Pygame package.

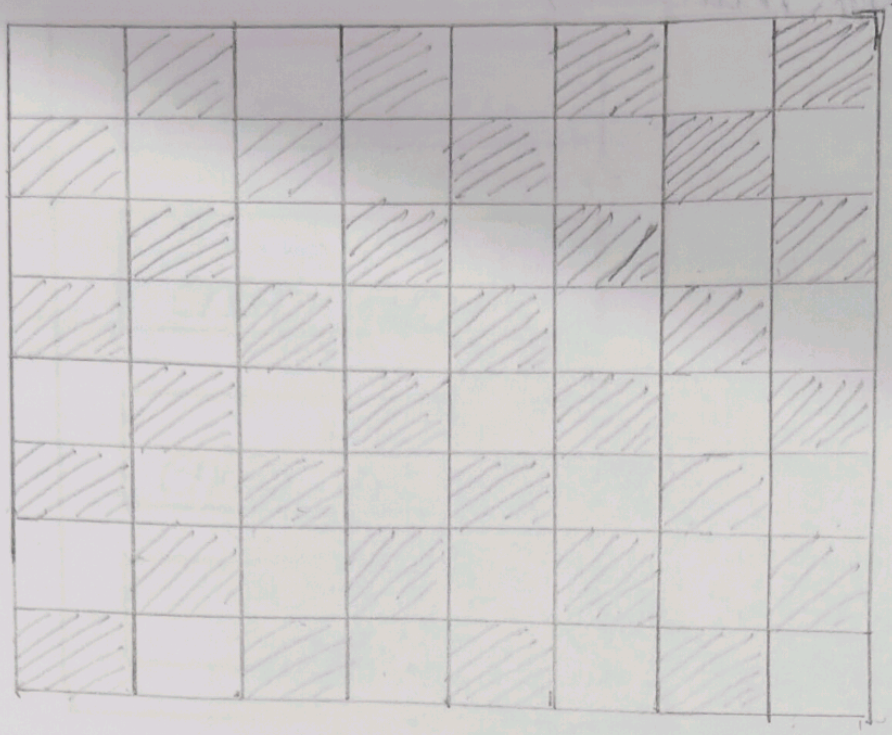
conditions:

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

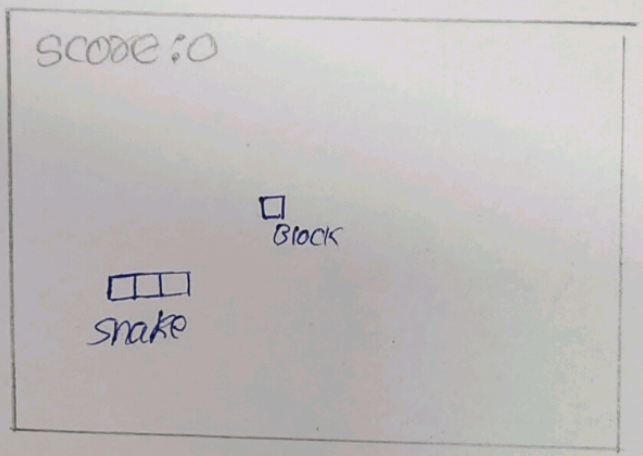
Algorithm:-

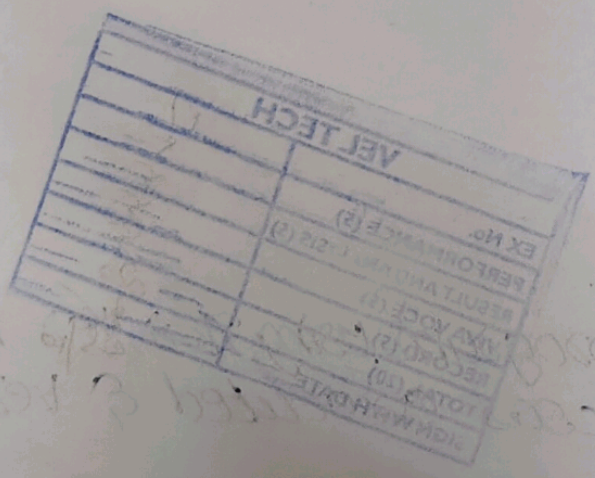
1. import pygame package and initialize it.
2. Define the window size & title
3. Create a snake class which initialize the snake position, colours and movement
4. create a fruits class which initialize the fruit position and colour
5. if snake hits the window. game over
6. create a function to check if the snake collides with windows & and 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 & fruit
9. Create a game loop a continuously update the game display, & check for collisions.
10. end the game if the user quits (or) the snake collides with the window.

sample output :



output:





Program:

```
#importing libraries
import pygame
import time
import random
snake_speed=15
# window size
window_x=720
window_y=480
# defining colours
black = pygame.colour(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)
# initialise game window
pygame.display.set_caption('Green for Geeks
        snakes')
game_window=pygame.display.set_mode(window
        _x, window_y)
# FPS(frames per second) controller
fps = pygame.time.clock()
snake_body =[100,50]
        [90,50]
        [80,50],
        [70,50]
# fruit position
fruit_position=[random.randrange(1,(window_x//10)*
        random.randrange(1,(window_y//10)*
fruit_spawn=true
# setting default snakee direction tocords
#right
direction='RIGHT'
    change_to=direction
```

```
# initial score
  score=0
# displaying score function
  def show_score(choice,colour,font,size)
# create font=Pygame.font.SysFont(font,size)
# create the display surface object
# score - surface
  score_surface=score_font.render('score'),t
  str(score),True colour)
# 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():
# create font a text surface on which text
# will be draw
  game_over_surface=my_font.render(
  your score is:'str(score),True,red)
# create a rectangular object for the text
# surface object
  game_over_rect.midtop=(window x/2,window_yi
# blit will draw the text on screen gam_window
  .blit(game_over_surface,game_over_rect)
  py game.display.flip()
# after 2 seconds we will quit the program tim.
  sleep()
# deactivating Py game library pygame.quit()
# quit the program
  quit()
# main function
  while True:
# handling key events
  for event in pygame.event.get():
    if event.type==py game.KEY DOWN:
      if event.key==pygame.K_UP:
        change_to='UP'
      if event.key==pygame.K_DOWN:
        change_to 'DOWN'
```

```
# simultaneously
if change_to == 'UP' and direction = 'DOWN'
    direction = 'UP'
if change_to == 'LEFT' and direction != 'RIGHT'
    direction = 'LEFT'
if change_to == 'RIGHT' and direction != '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
# 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_not = [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,
# display 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 2: write a python to develop a chess board using pygame samer

Algorithm:

1) import pygame & initialize it
2) set screen size and title
3) define colors for the board & pieces

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

```
program : Import pygame
        #initialize pygame
          pygame.init()
        # set screen size and title
          screen=pygame.size=(640,640)
          screen=pygame.set_caption('Chess Board')
        # Define colours
          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):
          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)
# Define function to draw the pieces
  def draw_pieces(board)
      piece_images={
        'r':Pygame.image.load('images/rook.png'),
        'n':Pygame.image.load('image/knight.png'),
        'b':Pygame.image.load('image/bishop.png'),
        'q':Pygame.image.load('image/queen.png'),
        'k':Pygame.image.load('image/king.png'),
        'p':pygame.image.load('image/pawn.png'),
        }
        for row in range(8):
          for col in range(8):
            piece=board[row][col]
          if piece!='':
                piece.image=piece_image[piece]
```

```
        ['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():
        if event.type == pygame.QUIT:
            pygame.quit()
            quit()
    pygame.display.update()
```

Result: Thus the program for pygame is executed & verified successfully

VEL TECH	
PERFORMANCE (5)	5
VIVA VOCE (5)	5
RECORD (5)	5
TOTAL (20)	20
SIGN WITH DATE	25/10