Task 12:- Simulate Gaming Concepts using PY Game

Aim:- To simulate Gaming concepts using py game.

Snack Game:-

problem 12.1:- write a python program to create a snake Game using py game 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 py game 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 colour.
5. create a function to check if snake collides with the fruit and increases 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 display position, & check for collisions.
10. End the game if the user quites (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(0, 255,0)
blue = pygame.Color(0,0,255)
# Initiate game window
pygame.display.set_caption('Greeks for Greeks snakes')
game_window = pygame.display.set_mode((window
_x, window_x))
# 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_x//10)*10)]
fruit_spawn = True.
# setting default snake direction towards
# right
direction = 'right'
```

```
chage_ho:direction.
#initial score
score=0
#displaying score function
def show_score(choice,color,font,size):
  # creating font object score_font
  score_font=pygame.font.sysfont(font,size):
#create the display surface object
# score_surface
score_surface=score_font.render('score:'+str(score),
True,color)
#create a rectogubar 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 font object my_font
my_font=pygame.font.sysfont('times new roman',50)
#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 rectogubar object for the text
#surface object
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 2 seconds we will quit the program.
time.sleep(2)
# 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.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'

  #Moving the snake
  if direction == 'up':
      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
  #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:
# frame per second (Refresh rate
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 colors for the 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 the board by looping for each pieces and placing image.
5. define the initial state of the board as a list of the pieces.
6. draw the board
7. start the game loop

Program:-

```
import pygame
#initialize pygame
pygame.init()
#set screen size and title
screen_size = (600, 600)
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.rect(screen, square_color, square.
            rect)
#Define function to draw the pieces
def draw_pieces(board):
    piece_images = {

#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 and verified successfully.

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