**Prac 1:**

**1]create blank canvas**

import pygame

pygame.init()

screen =pygame.display.set\_mode((800,600))

pygame.display.set\_caption("Space Invaders")

done=False

while not done:

for event in pygame.event.get():

if event.type==pygame.QUIT:

done=True

pygame.display.flip()

**2] add image**

import pygame

import sys

pygame.init()

display\_surface = pygame.display.set\_mode((400, 400))

pygame.display.set\_caption('image')

image = pygame.image.load("Man-with-headphones-standing-beneath-an-apple-tree-in-a-park-listening-music--Make-it-Animated.jpg")

while True:

display\_surface.blit(image, (0, 0))

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

sys.exit()

pygame.display.update()

**3] add name**

import pygame

pygame.init()

screen = pygame.display.set\_mode((640, 480))

font = pygame.font.SysFont("Times New Roman", 72)

text = font.render("Mukesh", True, (158, 16, 16))

while True:

for event in pygame.event.get():

if event.type == pygame.QUIT or (event.type == pygame.KEYDOWN and event.key == pygame.K\_ESCAPE):

pygame.quit()

exit()

screen.fill((255, 255, 255))

screen.blit(text, (320 - text.get\_width() // 2, 240 - text.get\_height() // 2))

pygame.display.flip()

**4] collision detection**

import pygame

pygame.init()

screen = pygame.display.set\_mode((640, 480))

clock = pygame.time.Clock()

red = (255, 0, 0)

blue = (0, 0, 255)

rect1 = pygame.Rect(100, 100, 50, 50)

rect2 = pygame.Rect(300, 200, 50, 50)

speed = 5

while True:

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

exit()

keys = pygame.key.get\_pressed()

if keys[pygame.K\_LEFT]:

rect1.x -= speed

if keys[pygame.K\_RIGHT]:

rect1.x += speed

if keys[pygame.K\_UP]:

rect1.y -= speed

if keys[pygame.K\_DOWN]:

rect1.y += speed

if rect1.colliderect(rect2):

print("Collision detected!")

screen.fill((255, 255, 255))

pygame.draw.rect(screen, red, rect1)

pygame.draw.rect(screen, blue, rect2)

pygame.display.flip()

clock.tick(60)

**Snake Game**

import pygame

import time

import random

snake\_speed = 8

window\_x = 720

window\_y = 480

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)

pygame.init()

pygame.display.set\_caption('Magic Snake Game')

game\_window = pygame.display.set\_mode((window\_x, window\_y))

fps = pygame.time.Clock()

snake\_position = [100, 50]

snake\_body = [[100, 50],

[90, 50],

[80, 50],

[70, 50]

]

fruit\_position = [random.randrange(1, (window\_x//10)) \* 10,

random.randrange(1, (window\_y//10)) \* 10]

fruit\_spawn = True

direction = 'RIGHT'

change\_to = direction

score = 0

def show\_score(choice, color, font, size):

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)

def game\_over():

my\_font = pygame.font.SysFont('times new roman', 50)

game\_over\_surface = my\_font.render(

'Your Score is : ' + str(score), True, red)

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)

pygame.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\_DOWN:

change\_to = 'DOWN'

if event.key == pygame.K\_LEFT:

change\_to = 'LEFT'

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.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//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,

pygame.Rect(pos[0], pos[1], 10, 10))

pygame.draw.rect(game\_window, white, pygame.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 roman', 20)

pygame.display.update()

fps.tick(snake\_speed)

**Infinite Scrolling**

import math

import pygame as py

py.init()

clock = py.time.Clock()

screen = py.display.set\_mode((1200, 300))

bg = py.image.load("Man-with-headphones-standing-beneath-an-apple-tree-in-a-park-listening-music--Make-it-Animated.jpg").convert()

scroll = 0

tiles = math.ceil(1200 / bg.get\_width()) + 1

while True:

clock.tick(33)

for i in range(tiles):

screen.blit(bg, (bg.get\_width() \* i + scroll, 0))

scroll -= 6

if abs(scroll) > bg.get\_width():

scroll = 0

for event in py.event.get():

if event.type == py.QUIT:

py.quit()

exit()

py.display.update()

**Create camera shake effect**

Step 1] create new 2D project

Step2] Go into project(bottom side)

Step 3] click on create (select Sprites)

Step4] select circle and Diamond

Step 5] Select camera click add components -> new Script -> give name -> select cSharp -> click on edit Script

Step 6] write code in visual studio and save it

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Shake : MonoBehaviour

{

// Start is called before the first frame update

public Transform camTransform;

public float camShakeDuration;

public float camShakeAmount;

public float decrementFactor;

public Vector3 \_camOriginalPosition;

public void OnEnable()

{

\_camOriginalPosition = camTransform.position;

}

// Use this for initialization

void Start () {

}

// Update is called once per frame

void Update () {

if (camShakeDuration > 0)

{

camTransform.localPosition = \_camOriginalPosition + Random.insideUnitSphere \* camShakeAmount;

camShakeDuration -= Time.deltaTime \* decrementFactor;

}

else

{

camShakeDuration = 0f;

camTransform.localPosition = \_camOriginalPosition;

}

}

}

Step 7] go to unity -> drag and drop main camera in cam transform

Step 8] cam shake duration (10)

Step 9] cam shake Amount(0.2)

**SNOWFALL**

1] select effect -> particle system

2] go to inspector tab

3]shape -> scale (x=10,z=10)

4] start speed (0)

5]velocity over lifetime (random between 2 constant linear -> y=(-1,-2)

6]Emmision -> rate over time (100)

7]start size (random between 2 constant (0.05, 0.2)

8]size over lifetime( change the graph)

9]Noise -> strength (0.2)

**Practical 8**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class enemy : MonoBehaviour

{

public Transform player;

public float followSpeed;

// Use this for initialization

// Update is called once per frame

void Update()

{

transform.position = Vector3.MoveTowards(this.transform.position, player.position, followSpeed = Time.deltaTime);

}

}

////////////////player script

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class player : MonoBehaviour

{

public float speed = 5F;

// Use this for initialization

void Start()

{

}

// Update is called once per frame

void Update()

{

float moveHorizontal = Input.GetAxis("Horizontal");

float moveVertical = Input.GetAxis("Vertical");

Vector3 movement = new Vector3(moveHorizontal, 0.0f, moveVertical);

transform.Translate(movement \* speed \* Time.deltaTime);

}

}