Experiment No.3
Develop a scene in Unity that includes: i. a cube, plane and sphere, apply transformations on the 3 game objects. ii. add a video and audio source
Date of Performance:
Date of Submission:

AIM:

Develop a scene in Unity that includes:

I. a cube, plane and sphere, apply transformations on the 3 game objects.
II. add a video and audio source

OBJECTIVES:

- Game Object Setup and Transformation: Add cube, sphere, and plane to the scene, apply transformations (position, rotation, scale).
- Video Source Integration: Create a "VideoPlayer" GameObject, add Video Player component, and play a specified video clip within the scene.
- Audio Source Integration: Attach Audio Source component to a game object, play an audio clip, and configure audio settings.
- Camera and Lighting Setup (Optional): Position camera for scene capture, add lighting for visual quality.
- Testing and Interaction (Optional): Verify functionality, add user interactions, scripting if needed.
- Documentation and Presentation: Document the process and create a presentation.
- Optimization and Refinement (Optional): Optimize for performance and refine aesthetics based on feedback.

THEORY:

- Unity Game Objects and Transformations:

Unity is a popular game development engine that uses GameObjects as the fundamental building blocks of a scene.

GameObjects represent entities in the scene, and they can be 3D models, lights, cameras, or any other element.

Transformations in Unity involve manipulating a GameObject's position, rotation, and scale to place and orient it as needed within the 3D world.

To apply transformations, you use the Transform component of a GameObject. You can manipulate this component directly in the Unity Inspector or programmatically in C# scripts.

Common transformations include:

Position: Adjusts the object's location in the 3D space (X, Y, Z coordinates).

Rotation: Changes the object's orientation (Euler angles or Quaternion).

Scale: Modifies the object's size.

unity

- Adding Video and Audio Sources:

Video Source Integration:

Unity provides the Video Player component for playing video content within a scene.

To integrate video:
Create an empty GameObject (e.g., "VideoPlayer") to serve as the video player.
Attach the Video Player component to this GameObject.
Import video clips into your Unity project or specify URLs for online videos.
Assign the video clip to the Video Player component.
Configure additional settings, such as video playback speed, loop behavior, and rendering mode.
Create a UI element or texture to display the video content, and set it as the target for the Video Player component.
Audio Source Integration:

Unity uses the Audio Source component to play audio content, including music, sound effects, and voiceovers.
To integrate audio:
Attach the Audio Source component to a GameObject (e.g., a cube or any object that should emit sound).
Import or specify audio clips to be played within the scene.
Assign the audio clip to the Audio Source component.
Configure audio settings such as volume, spatial blend, and 3D sound settings (if needed).

- Use C# scripts to control audio playback, such as triggering audio on specific events or in response to user interactions.
- Ensure that the video and audio formats you use are compatible with Unity.
- Optimize video and audio assets for performance and quality.
- Consider using UI elements, textures, or materials to display video content in the scene.

Use Unity's AudioSource and VideoPlayer documentation for more details on configuration and scripting.
By mastering these foundational concepts in Unity, you can develop a scene that includes 3D objects with transformations, video playback, and audio integration to create engaging and interactive experiences for your users.

CODE:

Code for Unity Game Objects and Transformations:

```
using UnityEngine;
public class ObjectTransformer : MonoBehaviour
{
// Public variables to control the transformations
public Vector3 moveDirection = Vector3.zero;
public Vector3 rotationSpeed = Vector3.zero;
public Vector3 scalingSpeed = Vector3.zero;
public bool isActive = true;
// Update is called once per frame
void Update()
{
```

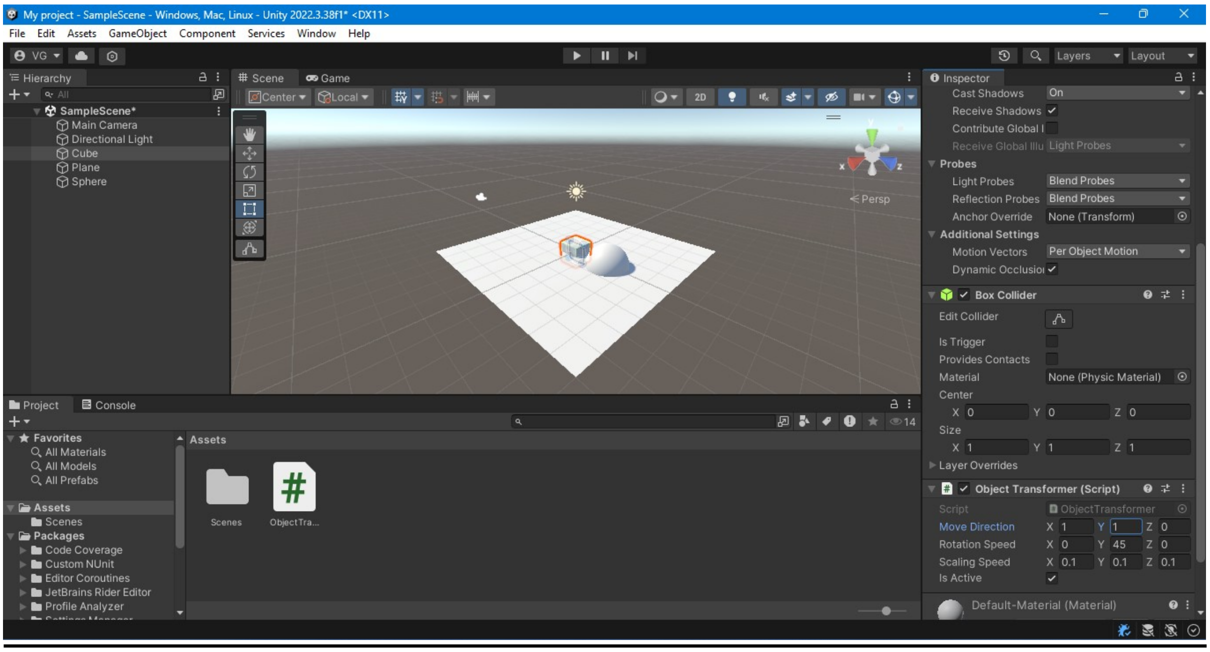
```
if (isActive)
{
// Move the object
transform.Translate(moveDirection * Time.deltaTime);
// Rotate the object
transform.Rotate(rotationSpeed * Time.deltaTime);
// Scale the object
transform.localScale += scalingSpeed * Time.deltaTime;
}
}
// Methods to enable/disable transformation
public void EnableTransform()
{
isActive = true;
}
public void DisableTransform()
{
isActive = false;
}
// Methods to set transformation parameters
public void SetMoveDirection(Vector3 newMoveDirection)
{
moveDirection = newMoveDirection;
}
public void SetRotationSpeed(Vector3 newRotationSpeed)
{
rotationSpeed = newRotationSpeed;
}
public void SetScalingSpeed(Vector3 newScalingSpeed)
{ scalingSpeed = newScalingSpeed;
}
}
```

Code for Adding Video and Audio Sources:

```
using UnityEngine;
using UnityEngine.Video;

public class ObjectTransformer : MonoBehaviour
{
   // Public variables to control the transformations
   public Vector3 moveDirection = Vector3.zero;
   public Vector3 rotationSpeed = Vector3.zero;
   public Vector3 scalingSpeed = Vector3.zero;
   public bool isActive = true;

   // Video and Audio components
   public VideoPlayer videoPlayer;
   public AudioSource audioSource;

   // Update is called once per frame
   void Update()
   {
       if (isActive)
       {
           // Move the object
           transform.Translate(moveDirection * Time.deltaTime);

           // Rotate the object
           transform.Rotate(rotationSpeed * Time.deltaTime);

           // Scale the object
           transform.localScale += scalingSpeed * Time.deltaTime;
       }
   }

   // Methods to enable/disable transformation
   public void EnableTransform()
   {
       isActive = true;
   }

   public void DisableTransform()
   {
       isActive = false;
   }

   // Methods to set transformation parameters
   public void SetMoveDirection(Vector3 newMoveDirection)
   {
       moveDirection = newMoveDirection;
   }

   public void SetRotationSpeed(Vector3 newRotationSpeed)
   {
       rotationSpeed = newRotationSpeed;
   }

   public void SetScalingSpeed(Vector3 newScalingSpeed)
   {
       scalingSpeed = newScalingSpeed;
   }
```

```
    // Methods to control video and audio
    public void PlayVideo()
    {
        if (videoPlayer != null)
        {
            videoPlayer.Play();
        }
    }

    public void PauseVideo()
    {
        if (videoPlayer != null)
        {
            videoPlayer.Pause();
        }
    }

    public void StopVideo()
    {
        if (videoPlayer != null)
        {
            videoPlayer.Stop();
        }
    }

    public void PlayAudio()
    {
        if (audioSource != null)
        {
            audioSource.Play();
        }
    }

    public void PauseAudio()
    {
        if (audioSource != null)
        {
            audioSource.Pause();
        }
    }

    public void StopAudio()
    {
        if (audioSource != null)
        {
            audioSource.Stop();
        }
    }
}
```

OUTPUT:



Cast Shadows: On
Receive Shadows: ✓
Contribute Global I: □
Receive Global Illu: Light Probes

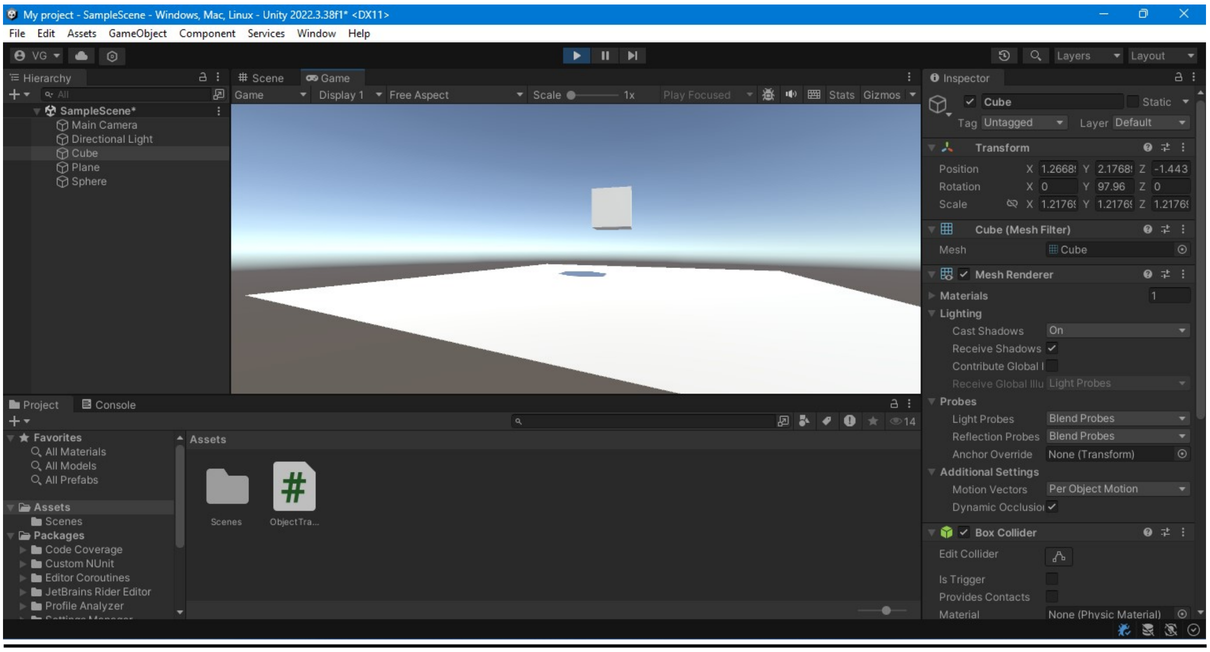
Probes
Light Probes: Blend Probes
Reflection Probes: Blend Probes
Anchor Override: None (Transform)

Additional Settings
Motion Vectors: Per Object Motion
Dynamic Occlusioı: ✓

✓ Box Collider
Edit Collider
Is Trigger: □
Provides Contacts: □
Material: None (Physic Material)
Center: X 0 Y 0 Z 0
Size: X 1 Y 1 Z 1
Layer Overrides

✓ Object Transformer (Script)
Script: ObjectTransformer
Move Direction: X 1 Y 1 Z 0
Rotation Speed: X 0 Y 45 Z 0
Scaling Speed: X 0.1 Y 0.1 Z 0.1
Is Active: ✓

Default-Material (Material)



✓ Cube □ Static
Tag: Untagged Layer: Default

Transform
Position: X 1.2668! Y 2.1768! Z -1.443
Rotation: X 0 Y 97.96 Z 0
Scale: X 1.2176! Y 1.2176! Z 1.2176!

Cube (Mesh Filter)
Mesh: Cube

✓ Mesh Renderer
Materials: 1
Lighting
Cast Shadows: On
Receive Shadows: ✓
Contribute Global I: □
Receive Global Illu: Light Probes
Probes
Light Probes: Blend Probes
Reflection Probes: Blend Probes
Anchor Override: None (Transform)
Additional Settings
Motion Vectors: Per Object Motion
Dynamic Occlusioı: ✓

✓ Box Collider
Edit Collider
Is Trigger: □
Provides Contacts: □
Material: None (Physic Material)

CONCLUSION: This experiment demonstrates basic scene setup in Unity, including the creation and transformation of 3D objects (cube, plane, and sphere). By applying different transformations such as scaling, rotating, and translating these objects, you gain an understanding of how to manipulate GameObjects in a 3D space. Additionally, integrating a video and audio source introduces multimedia elements into the scene, enhancing the interactive and immersive experience. This basic scene can serve as a foundation for more complex projects involving animations, user interaction, and further multimedia integration.