Program 1

```
using System.Collections;

using System.Collections.Generic;

using Unity.VisualScripting;

using UnityEngine;

public class FirstLab : MonoBehaviour

{

public float speed = 100f;

public ParticleSystem Particles;

private void Update()

{

// Move the capsule forward in the Z direction

if(Input.GetKeyDown(KeyCode.W))

{

transform.Translate(Vector3.forward * speed * Time.deltaTime);

}

if(Input.GetKeyDown(KeyCode.A))

{

transform.Translate(Vector3.left * speed * Time.deltaTime);

}

if(Input.GetKeyDown(KeyCode.D))

{

transform.Translate(Vector3.right * speed * Time.deltaTime);

}

}

private void OnCollisionEnter(Collision collision)

{

if (collision.gameObject.tag=="End")
{
Particles.Play(); // Activate particles
}
}
}
```

Program-2

```
using UnityEngine;
public class LAB2 : MonoBehaviour
{
public Animator anim;
public void two()
{
anim.SetBool("a",true);
}
public void one()
{
anim.SetBool("a",false);
anim.SetBool("b",false);
anim.SetBool("c",false);
}
public void three()
{
anim.SetBool("b",true);
}
public void four()
{
```

```
anim.SetBool("c",true);
}
}
```

Program-3

Create c# script AI and attach to Cylinder i.e our AI

```
using UnityEngine;
using UnityEngine.AI;
public class AI : MonoBehaviour
{
public GameObject player;
public NavMeshAgent agent;
void Update () {
agent.SetDestination(player.transform.position);
}
}
```

Create C# Script ,Attach script to character and attach reference for character controller

```
using UnityEngine;
public class LAB4 : MonoBehaviour
{
public float moveSpeed = 50f;
public float rotationSpeed = 700f;
public CharacterController controller;
private Vector3 moveDirection;
void Update()
{
```

```
float moveX = Input.GetAxis("Horizontal");
float moveZ = Input.GetAxis("Vertical");
// Calculate movement direction based on input
moveDirection = new Vector3(moveX, 0f, moveZ);
if (moveDirection.magnitude > 0)
{
Quaternion toRotation = Quaternion.LookRotation(moveDirection, Vector3.up);
transform.rotation = Quaternion.RotateTowards(transform.rotation, toRotation,
rotationSpeed * Time.deltaTime);
}
// Apply the movement to the character
controller.Move(moveDirection * moveSpeed * Time.deltaTime);
}
}
```

Program-4

```
using UnityEngine;
public class LAB3 : MonoBehaviour
{
public GameObject cube;
public GameObject sphere;
void Start()
{
sphere.SetActive(false);
}
// Update is called once per frame
void Update()
{
cube.transform.Rotate(0,30,0);
}
public void show()
{
sphere.SetActive(true);
}
public void hide()
{
sphere.SetActive(false);
}
}
```

Program-5

create script name it PlacementIndicator and attach to placement

```
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.XR.ARFoundation;
using UnityEngine.XR.ARSubsystems;
public class PlacementIndicator : MonoBehaviour
{
private ARRaycastManager rayManager;
private GameObject visual; // Start is called before the first frame update
void Start()
{
rayManager = FindObjectOfType<ARRaycastManager>();
visual = transform.GetChild(0).gameObject;
//hide placement indicator
visual.SetActive(false);
}
void Update()
{
List<ARRaycastHit> hits = new List<ARRaycastHit>();
//shoot raycast from center of screen
rayManager.Raycast(new Vector2(Screen.width / 2, Screen.height / 2), hits,
TrackableType.Planes);
//if we hit AR plane update position and rotation
if (hits.Count > 0)
{
transform.position = hits[0].pose.position;
transform.rotation = hits[0].pose.rotation;
if (!visual.activeInHierarchy)
visual.SetActive(true);
}
}
}
```

create script spawn_object attach to SpawnManager

```
using UnityEngine;
public class Spawn_object : MonoBehaviour
{
public GameObject objectToSpawn;
private PlacementIndicator placeIndicate;
private GameObject spawnedObject; // Reference to the spawned object
private float initialDistance; // Distance between fingers for scaling
private Vector3 initialScale; // Initial scale of the object

private bool isScaling = false; // Flag to check if scaling is active

void Start()

{

placeIndicate = FindObjectOfType<PlacementIndicator>();

}

void Update()

{

if (Input.touchCount > 0)

{

Touch touch = Input.touches[0];

// Check for object spawn on touch begin

if (touch.phase == TouchPhase.Began && spawnedObject == null)

{

ShowObject();

}

// Handle scaling with pinch gesture

if (Input.touchCount == 2)

{

ScaleObject();

}

// Handle rotation with single finger drag

else if (Input.touchCount == 1 && spawnedObject != null)

{

RotateObject(touch);

}

}

}

void ShowObject()
```

```
{
spawnedObject = Instantiate(objectToSpawn, placeIndicate.transform.position,
placeIndicate.transform.rotation);
}
void ScaleObject()
{
Touch touch1 = Input.GetTouch(0);
Touch touch2 = Input.GetTouch(1);
if (touch1.phase == TouchPhase.Began || touch2.phase == TouchPhase.Began)
{
initialDistance = Vector2.Distance(touch1.position, touch2.position);
initialScale = spawnedObject.transform.localScale;
isScaling = true;
}
if (touch1.phase == TouchPhase.Moved && touch2.phase == TouchPhase.Moved &&
isScaling)
{
float currentDistance = Vector2.Distance(touch1.position, touch2.position);
float scaleFactor = currentDistance / initialDistance;
spawnedObject.transform.localScale = initialScale * scaleFactor;
}
}
void RotateObject(Touch touch)
{
if (touch.phase == TouchPhase.Moved)
{
float rotationSpeed = 0.2f;
spawnedObject.transform.Rotate(Vector3.up, -touch.deltaPosition.x * rotationSpeed);
}
}
}
```

Program-6

```
<html>
<head>
<script src="https://aframe.io/releases/1.4.0/aframe.min.js"></script>
</head>
<body>
<a-scene>
<!-- Camera and lighting -->
<a-entity position="0 1.6 4">
<a-camera></a-camera>
</a-entity>
<a-light type="directional" position="1 1 1" intensity="0.8"></a-light>
<a-light type="ambient" intensity="0.5"></a-light>
<!-- 3D Cube -->
<a-box position="0 1 -3" rotation="0 45 0" color="red" animation="property: rotation; to: 0
90 0; loop: true; dur: 1000"></a-box>
<a-box position="2 1 -3" rotation="0 0 45" color="green" animation="property: rotation; to: 0

90 0; loop: true; dur: 2000"></a-box>
<!-- Ground -->

<a-plane position="0 0 -4" rotation="-90 0 0" width="10" height="10" color="#7BC8A4"></a-
plane>

</a-scene>
```

```
</body>
</html>
```

Program-7

```
<!doctype html>
<html>
<head>
<title>A-Frame Geolocation</title>

<script src="https://aframe.io/releases/1.2.0/aframe.min.js"></script>
<script>
document.addEventListener('DOMContentLoaded', function() {
function showShapes(position)
 {
var currentLatitude = position.coords.latitude;
var currentLongitude = position.coords.longitude;
console.log("Latitude: " + currentLatitude);
console.log("Longitude: " + currentLongitude);

var locations = [
{ id: "box", lat:12.9957888,lon:77.6994816, threshold: 0.005 },
{ id: "cylinder", lat: 12.90509057, lon: 77.55971556, threshold: 0.005 },
{ id: "sphere", lat: 12.9564672,lon: 77.594624,threshold: 0.005}
];
locations.forEach(location => {
var shape = document.querySelector(`#${location.id}`);
if (shape && Math.abs(currentLatitude - location.lat) < location.threshold &&

Math.abs(currentLongitude - location.lon) < location.threshold) {

shape.setAttribute('visible', true);

}

});

}

function locationError(error) {

console.error("Error getting location: ", error);

document.getElementById('currentLocation').innerHTML =

`Error getting location: ${error.message}`;

}

function getLocation() {

if (navigator.geolocation) {

navigator.geolocation.getCurrentPosition(showShapes, locationError, { enableHighAccuracy: true,
timeout: 10000, maximumAge: 0 });

} else {

document.getElementById('currentLocation').innerHTML =

"Geolocation is not supported by this browser.";

}

}

getLocation();

});</script>

</head>

<body><h3 id="currentLocation">Fetching location...</h3>

<a-scene><a-box id="box" position="0 0.5 -3" rotation="0 45 0" color="red" visible="false"></a-box>

<a-cylinder id="cylinder" position="2 0.5 -3" radius="0.5" height="1.5" color="blue" visible="false"></a-
cylinder>

<a-sphere id="sphere" position="-2 0.75 -3" radius="0.75" color="green" visible="false"></a-sphere>

<a-camera gps-camera rotation-reader></a-camera>
```

```
</a-scene>
</body>
</html>
```

Program-8

Step 1: download .Patt file and same .jpg file

https://github.com/jeromeetienne/AR.js/blob/master/three.js/examples/marker-training/examples/pattern-files/pattern-letterA.patt

```
<!DOCTYPE html>
<html>
 <head>
 <!-- Include A-Frame -->
 <script src="https://aframe.io/releases/1.2.0/aframe.min.js"></script>
 <!-- Include AR.js for A-Frame -->
 <script src="https://cdn.jsdelivr.net/gh/jeromeetienne/ar.js/aframe/build/aframear.min.js"></script>
 </head>
 <body style="margin: 0px; overflow: hidden;">
 <a-scene embedded arjs>
 <!-- Marker -->
 <a-marker type="pattern" url="pattern-letterA.patt">
 <a-box position="0 0.5 0" material="opacity: 0.5;"></a-box>
 <a-cylinder color="green" height="1.0" radius="0.5" position="1 0.5 0"></a-cylinder>
 </a-marker>
 <!-- Camera -->
 <a-entity camera></a-entity>
 </a-scene>
 </body>
</html>
```

Create server.js script for creating server

```
const express = require('express');
const path = require('path');
const app = express();
// Serve static files from the "public" directory
app.use(express.static(path.join(__dirname, 'public')));
// Start the server
const PORT = process.env.PORT || 3000;
app.listen(PORT, () => {
 console.log(`Server is running on http://localhost:${PORT}`);
});
```

Program-9

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <title>Simple A-Frame Scene</title>
 <script src="https://aframe.io/releases/1.2.0/aframe.min.js"></script>
 <style>
 /* Ensure the body takes full height */
 body, html {
 margin: 0;
 padding: 0;
 height: 100%;
 }
 /* Style the button to make it visible and positioned correctly */
 button {
 position: absolute;
 top: 20px;
 left: 50%;
 transform: translateX(-50%);
 padding: 10px 20px;
 font-size: 16px;
 background-color: #4CC3D9;
 border: none;
 color: white;
 border-radius: 5px;
 cursor: pointer;
 z-index: 10; /* Ensure it's above the scene */
 }
 /* Add hover effect for the button */
 button:hover {
 background-color: #3a9ca1;
 }
 </style>
</head>
<body>
 <a-scene>
 <a-assets>
 <a-asset-item id="value" src="Nike.glb"></a-asset-item>
 </a-assets>
 <a-entity position="0 1.6 0">
 <a-camera></a-camera>
 </a-entity>
 <a-entity id="model" gltf-model="#value" position="0 0 -5" rotation="0 45 0" scale="15 15
15"></a-entity>
 </a-scene>
 <!-- Button to trigger rotation -->
 <button onclick="rotateModel()">Rotate Model 45°</button>
 <script>
 // Function to rotate the model by 45 degrees each time
 function rotateModel() {
 const model = document.querySelector('#model');

 // Get the current rotation
 let currentRotation = model.getAttribute('rotation');

 // Increment the Y rotation by 45 degrees
 currentRotation.y += 45;

 // Apply the updated rotation
 model.setAttribute('rotation', currentRotation);
 }
 </script>
</body>
</html>
```

Create server.js script for creating server

```
const express = require('express');
const path = require('path');
const app = express();
// Serve static files from the "public" directory
app.use(express.static(path.join(__dirname, 'public')));
// Start the server
const PORT = process.env.PORT || 3000;
app.listen(PORT, () => {
 console.log(`Server is running on http://localhost:${PORT}`);
});
```