LAMPIRAN

Fuzzy Logic

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPro;
using UnityEngine.SceneManagement;
public class UseIfLogic : MonoBehaviour
{
    public double TestSpeed;
    public double TestDepth;
    public double TestAngle;
    public string ResultTextPorsity;
    public TextMeshProUGUI ResultTMP;
    public TMP_InputField InputFieldSpeed;
    public TMP_InputField InputFieldDepth;
    public TMP_InputField InputFieldAngle;

    public Material MatSedikitGembur;
    public Material MatGembur;
    public Material MatSangatGembur;
```

```
public void RestartScene()
{
    SceneManager.LoadScene(0);
}
private void Start()
{
    TestSpeed = 6.808;
    TestDepth = 3;
    TestAngle = 60;
}
private void Update()
{
    SpeedUpdate();
    dephtUpdate();
    angleUpdate();
    OnCalculate();
}
private void SpeedUpdate()
{
    if (Input.GetKey(KeyCode.Q))
    {
        TestSpeed -= 0.01;
```

```
        if (TestSpeed <= 1) TestSpeed = 1;
    }
    if (Input.GetKey(KeyCode.E))
    {
        TestSpeed += 0.01;
        if (TestSpeed >= 19.917) TestSpeed = 19.917;
    }
    InputFieldSpeed.text = TestSpeed.ToString();
}
private void dephtUpdate()
{
    if (Input.GetKey(KeyCode.R))
    {
        TestDepth -= 0.01;
        if (TestDepth <= 0) TestDepth = 0;
    }
    if (Input.GetKey(KeyCode.F))
    {
        TestDepth += 0.01;
        if (TestDepth >= 7) TestDepth = 7;
    }
    InputFieldDepth.text = TestDepth.ToString();
```

```
    }
    private void angleUpdate()
    {
        if (Input.GetKey(KeyCode.C))
        {
            TestAngle -= 0.01;
            if (TestAngle <= 55) TestAngle = 55;
        }
        if (Input.GetKey(KeyCode.V))
        {
            TestAngle += 0.01;
            if (TestAngle >= 70) TestAngle = 70;
        }
        InputFieldAngle.text = TestAngle.ToString();
    }
    public string ResultSpeed(double value)
    {
        string result = "";
        if (value >= 1 && value <= 6.808)
        {
            result = "Lambat";
        }
```

```
    else if (value > 6.808 && value <= 10.169)
    {
        result = "Sedang";
    }
    else if (value > 10.169 && value <= 19.917)
    {
        result = "Cepat";
    }
    return result;
}
public string ResultDepth(double value)
{
    string result = "";
    if (value >= 0 && value <= 3.5)
    {
        result = "Dangkal";
    }
    else if (value > 3.5 && value <= 7)
    {
        result = "Dalam";
    }
    return result;
```

```
}
public string ResultAngle(double value)
{
    string result = "";
    if (value >= 55 && value <= 60)
    {
        result = "Kecil";
    }
    else if (value > 60 && value <= 65)
    {
        result = "Sedang";
    }
    else if (value > 65 && value <= 70)
    {
        result = "Besar";
    }
    return result;
}
public float raycastDistance = 5f;
public void OnCalculate()
{
    ResultTextPorsity = "";
```

```
        CalculationPorosity();
        Ray ray = new Ray(transform.position, Vector3.down);
        RaycastHit hit;
        if (Physics.Raycast(ray, out hit, raycastDistance))
        {
            if (hit.collider.CompareTag("box"))
            {
                MeshRenderer boxRenderer =
hit.collider.GetComponent<MeshRenderer>();

                if (boxRenderer != null)
                {
                    switch (ResultTextPorsity)
                    {
                        case "Sedikit Gembur":
                            boxRenderer.material = MatSedikitGembur;
                            break;
                        case "Gembur":
                            boxRenderer.material = MatGembur;
                            break;
                        case "Sangat Gembur":
                            boxRenderer.material = MatSangatGembur;
```

```
                    break;
                default:
                    break;
            }
        }
    }
}
[ContextMenu ("Test Result")]
public void CalculationPorosity()
{
    string Speed = ResultSpeed(TestSpeed);
    string Depth = ResultDepth(TestDepth);
    string angle = ResultAngle(TestAngle);
    if (Speed == "Lambat" && Depth == "Dangkal" &&
angle == "Kecil")
    {
        ResultTextPorsity = "Sedikit Gembur";
    }
    else if (Speed == "Lambat" && Depth == "Dangkal" &&
angle == "Sedang")
    {
```

```
        ResultTextPorsity = "Sedikit Gembur";
    }
    else if (Speed == "Lambat" && Depth == "Dangkal" &&
angle == "Besar")
    {
        ResultTextPorsity = "Sedikit Gembur";
    }
    else if (Speed == "Lambat" && Depth == "Dalam" &&
angle == "Kecil")
    {
        ResultTextPorsity = "Sedikit Gembur";
    }
    else if (Speed == "Lambat" && Depth == "Dalam" &&
angle == "Sedang")
    {
        ResultTextPorsity = "Sedikit Gembur";
    }
    else if (Speed == "Lambat" && Depth == "Dalam" &&
angle == "Besar")
    {
        ResultTextPorsity = "Gembur";
    }
```

```
    else if (Speed == "Sedang" && Depth == "Dangkal" &&
angle == "Kecil")
    {
       ResultTextPorsity = "Sedikit Gembur";
    }
    else if (Speed == "Sedang" && Depth == "Dangkal" &&
angle == "Sedang")
    {
       ResultTextPorsity = "Gembur";
    }
    else if (Speed == "Sedang" && Depth == "Dangkal" &&
angle == "Besar")
    {
       ResultTextPorsity = "Gembur";
    }
    else if (Speed == "Sedang" && Depth == "Dalam" &&
angle == "Kecil")
    {
       ResultTextPorsity = "Sedikit Gembur";
    }
    else if (Speed == "Sedang" && Depth == "Dalam" &&
angle == "Sedang")
```

```
    {
        ResultTextPorsity = "Gembur";
    }
    else if (Speed == "Sedang" && Depth == "Dalam" &&
angle == "Besar")
    {
        ResultTextPorsity = "Sangat Gembur";
    }
    else if (Speed == "Cepat" && Depth == "Dangkal" &&
angle == "Kecil")
    {
        ResultTextPorsity = "Sedikit Gembur";
    }
    else if (Speed == "Cepat" && Depth == "Dangkal" &&
angle == "Sedang")
    {
        ResultTextPorsity = "Sedikit Gembur";
    }
    else if (Speed == "Cepat" && Depth == "Dangkal" &&
angle == "Besar")
    {
        ResultTextPorsity = "Gembur";
```

```
        }
        else if (Speed == "Cepat" && Depth == "Dalam" &&
angle == "Kecil")
        {
            ResultTextPorsity = "Gembur";
        }
        else if (Speed == "Cepat" && Depth == "Dalam" &&
angle == "Sedang")
        {
            ResultTextPorsity = "Gembur";
        }
        else if (Speed == "Cepat" && Depth == "Dalam" &&
angle == "Besar")
        {
            ResultTextPorsity = "Sangat Gembur";
        }
        ResultTMP.text = ResultTextPorsity;
    }
}
```

Movement/Pergerakan

```
using UnityEngine;

namespace Complete
{
    public class TankMovement : MonoBehaviour
    {
        public int m_PlayerNumber = 1;
        public float m_Speed = 12f;
        public float m_TurnSpeed = 180f;
        public AudioSource m_MovementAudio;
        public AudioClip m_EngineIdling;
        public AudioClip m_EngineDriving;

        private string m_MovementAxisName;
        private string m_TurnAxisName;
        private Rigidbody m_Rigidbody;
        private float m_MovementInputValue;
        private float m_TurnInputValue;

        private ParticleSystem[] m_particleSystems;
        public GameObject objekRodaLeft;
        public GameObject objekRodaRight;
```

```
    private void Awake ()
    {
        m_Rigidbody = GetComponent<Rigidbody> ();
    }


    private void OnEnable ()
    {
        m_Rigidbody.isKinematic = false;


        m_MovementInputValue = 0f;
        m_TurnInputValue = 0f;


        m_particleSystems =
GetComponentsInChildren<ParticleSystem>();
        for (int i = 0; i < m_particleSystems.Length; ++i)
        {
            m_particleSystems[i].Play();
        }
    }
    private void OnDisable ()
    {
        m_Rigidbody.isKinematic = true;
```

```
        for(int i = 0; i < m_particleSystems.Length; ++i)
        {
            m_particleSystems[i].Stop();
        }
    }
    private void Start ()
    {
        m_MovementAxisName = "Vertical" +
m_PlayerNumber;
        m_TurnAxisName = "Horizontal" + m_PlayerNumber;
    }
    public float rotationSpeed ;
    private void Update ()
    {
        m_MovementInputValue = Input.GetAxis
(m_MovementAxisName);
        if (m_MovementInputValue > 0)
        {
            objekRodaRight.transform.Rotate(Vector3.right  ,
rotationSpeed);
            objekRodaLeft.transform.Rotate(Vector3.right,
rotationSpeed);
```

```
        }
        m_TurnInputValue = Input.GetAxis
(m_TurnAxisName);
        EngineAudio ();
    }
    private void EngineAudio ()
    {
        if (Mathf.Abs (m_MovementInputValue) < 0.1f &&
Mathf.Abs (m_TurnInputValue) < 0.1f)
        {
            if (m_MovementAudio.clip == m_EngineDriving)
            {
                m_MovementAudio.clip = m_EngineIdling;
                m_MovementAudio.Play ();
            }
        }
        else
        {
            if (m_MovementAudio.clip == m_EngineIdling)
            {
                m_MovementAudio.clip = m_EngineDriving;
                m_MovementAudio.Play();
```

```
            }
        }
    }
    private void FixedUpdate ()
    {
        Move ();
        Turn ();
    }
    private void Move ()
    {
        Vector3 movement = transform.forward *
m_MovementInputValue * m_Speed * Time.deltaTime;
        m_Rigidbody.MovePosition(m_Rigidbody.position +
movement);
    }
    private void Turn ()
    {
        float turn = m_TurnInputValue * m_TurnSpeed *
Time.deltaTime;

        Quaternion turnRotation = Quaternion.Euler (0f, turn,
0f);
```

```
            m_Rigidbody.MoveRotation (m_Rigidbody.rotation *
turnRotation);
        }
    }

}
```