LAMPIRAN

LISTING PROGRAM

EnemyManager

```
using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
[Serializable]
public class EnemyManager {
    public Transform spawnPoint;
    [HideInInspector]public GameObject e_instance;
   public Transform[] patrolPoint;

    private AIyo ai;
       // Use this for initialization
       public void Setup()
    {
        ai = e_instance.GetComponent<AIyo>();
        for(int i = 0; i<patrolPoint.Length;i++)
        {
            ai.point[i] = patrolPoint[i];
        }
    }
    public void EnableControl()
    {
        ai.enabled = true;
    }
    public void DisableControl()
    {
        ai.enabled = false;
    }
    public void Reset()
    {
        e_instance.transform.position = spawnPoint.position;
        e_instance.transform.rotation = spawnPoint.rotation;

        e_instance.SetActive(false);
        e_instance.SetActive(true);
    }
}
```

PlayerControl.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PlayerControl : MonoBehaviour
{
    Animator anim;
    public AudioClip stepClip;

    public AudioSource stepSource;
    float front;
```

```
    float turn;


    // Use this for initialization
    void Start()
    {
        anim = GetComponent<Animator>();
        stepSource.clip = stepClip;

    }

    // Update is called once per frame
    void Update()
    {
        front = Input.GetAxis("Vertical");
        turn = Input.GetAxis("Horizontal");
        anim.SetFloat("speed", front, 0.1f, Time.deltaTime);
        anim.SetFloat("turn", turn, 0.1f, Time.deltaTime);

        Vector3 movement = new Vector3(turn, 0.0f, front);
        if (turn != 0 || front != 0)
        {

            transform.rotation = Quaternion.Slerp(transform.rotation,
Quaternion.LookRotation(movement), 0.15F);




        }
        Sound();

    }

    void Sound()
    {
        if (Input.GetKeyDown(KeyCode.W) || Input.GetKeyDown(KeyCode.S) ||
Input.GetKeyDown(KeyCode.A) || Input.GetKeyDown(KeyCode.D))
        {

            stepSource.Play();

        }
        else if (turn == 0 && front == 00)
        {
            stepSource.Stop();
        }

    }


    private void OnCollisionEnter(Collision collision)
    {
        if (collision.gameObject.CompareTag("Enemy"))
        {
            Destroy(gameObject);
        }
    }
```

```
}
```

Ablility.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Abilty : MonoBehaviour {
    private Bomming droppedUp;

       // Use this for initialization
       void Start () {
        droppedUp = GetComponent<Bomming>();
       }

       // Update is called once per frame
       void Update () {

       }
    private void OnTriggerEnter(Collider other)
    {
        if (other.CompareTag("LimitUp"))
        {
            Destroy(other.gameObject);

            droppedUp.maxDrop++;

        }
        else if (other.CompareTag("PowerUp"))
        {
            Destroy(other.gameObject);
            GameManager.lengthFire++;
        }
    }
}
```

clickOnLoad.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class clickOnLoad : MonoBehaviour {

    public void LoadByIndex(int sceneIndex)
    {
        SceneManager.LoadScene(sceneIndex);
    }
}
```

Pause.cs

```
using System.Collections;
```

```
using System.Collections.Generic;
using UnityEngine;

public class Pause : MonoBehaviour {

public void PauseOnClick(bool a)
    {
        if(a == true)
        {
            Time.timeScale = 0;
        }
        else if (a == false)
        {
            Time.timeScale = 1;
        }
    }
}
```

BankScore.cs

```
using System.Collections;
using System.Collections.Generic;
using System;
using System.Runtime.Serialization.Formatters.Binary;
using System.IO;
using UnityEngine.UI;
using UnityEngine;

public class BankScore : MonoBehaviour {
    public static BankScore bank;
    public static List<ScoreName> listScore = new List<ScoreName>();
    // Use this for initialization
    void Awake()
    {
        if (bank == null)
        {
            DontDestroyOnLoad(gameObject);
            bank = this;
        }
        else if (bank != null)
        {
            Destroy(gameObject);
        }
    }
    private void OnEnable()
    {
        OnLoad();
    }
    private void OnDisable()
    {
        OnSave();
    }
    void Start () {

       }

       // Update is called once per frame
       void Update () {
```

```
       }
    private void Sortir()
    {
        listScore.Sort();
        listScore.Reverse();
    }
    void OnSave()
    {
        BinaryFormatter bf = new BinaryFormatter();
        FileStream file = File.Create(Application.persistentDataPath +
"/bank2.dat");

        DataBank data = new DataBank();
        data.listScore = listScore;

        bf.Serialize(file, data);
        file.Close();
    }
    void OnLoad()
    {
        if (File.Exists(Application.persistentDataPath + "/bank2.dat"))
        {
            BinaryFormatter bf = new BinaryFormatter();
            FileStream file = File.Open(Application.persistentDataPath +
"/bank2.dat", FileMode.Open);
           DataBank data = (DataBank)bf.Deserialize(file);
            file.Close();

            listScore = data.listScore;

        }
    }
}
[Serializable]
class DataBank
{
    public List<ScoreName> listScore = new List<ScoreName>();
}
```

GetScoreName.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine.UI;
using UnityEngine;

public class GetScoreName : MonoBehaviour {

    public InputField store;

    public void AmbilNama()
    {
        if (BankScore.listScore.Count != 5)
        {
            BankScore.listScore.Add(new
ScoreName(Disimpan.scoreWhenPlaying, store.text));
            BankScore.listScore.Sort();
            BankScore.listScore.Reverse();
        }
```

```
        else if(BankScore.listScore[4].angka < Disimpan.scoreWhenPlaying)
        {
            BankScore.listScore.RemoveAt(4);
            BankScore.listScore.Add(new
ScoreName(Disimpan.scoreWhenPlaying, store.text));
            BankScore.listScore.Sort();
            BankScore.listScore.Reverse();
        }
        Disimpan.scoreWhenPlaying = 0;
    }

}
```

InputHighScore.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine.UI;
using UnityEngine;

public class InputHighScore : MonoBehaviour {
    private Text self;
    private int currentValue;
       // Use this for initialization
       void Start () {
        self = GetComponent<Text>();

       }

       // Update is called once per frame
       void Update () {
        if (currentValue < HighScore.value)
        {
            currentValue = HighScore.value;
        }
        self.text = currentValue.ToString();

       }
}
```

TampilkanHighScore.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class TamplikanHighScore : MonoBehaviour {
    public Text[] nama;
    public Text[] score;
    // Use this for initialization
    void Start() {

       }

       // Update is called once per frame
       void Update () {
```

```
        if (BankScore.listScore.Count >= 1)
        {
            for (int i = 0; i < BankScore.listScore.Count; i++)
            {
                nama[i].text = BankScore.listScore[i].nama;
                score[i].text = BankScore.listScore[i].angka.ToString();
            }
        }
        else
        {
            for (int i = 0; i < nama.Length; i++)
            {
                nama[i].text = string.Empty;
                score[i].text = string.Empty;
            }
        }
    }
}
```

AIyo.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine.AI;
using UnityEngine;

public class AIyo : MonoBehaviour {
    // [HideInInspector]public Transform[] point;
    [HideInInspector]
    public List<Transform> point = new List<Transform>();
    private int bil;
    private NavMeshAgent self;
    private bool  ketemu;

    private bool patrol;

    private Transform target;
    private float startTimeelapsed;



       // Use this for initialization
       void Start () {
        self = GetComponent<NavMeshAgent>();
        ketemu = false;
        patrol = true;
        bil = 0;


       }


    // Update is called once per frame
    void Update()
    {
        Look();

    }
```

```
    void Patrol()
    {



        self.destination = point[bil].position;
            self.Resume();
            if (self.remainingDistance <= self.stoppingDistance &&
!self.pathPending)
            {

            bil = Random.Range(0, point.Count);
            }




    }
    void Look()
    {

        RaycastHit hit;
        Debug.DrawRay(transform.position, transform.forward.normalized *
5f, Color.green);
        if (Physics.SphereCast(transform.position, 0.5f, transform.forward,
out hit,5f)
             && hit.collider.CompareTag("Player"))
        {
            target = hit.transform;
            Chaser();

        }
        else
        {
            Patrol();


        }

    }


    void Chaser()
    {

            self.SetDestination(target.position);
        self.Resume();

    }
    private void OnDrawGizmos()
    {
        Gizmos.DrawWireSphere(transform.position, 0.5f);
    }
}
```

Game manager.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;

public class GameManager : MonoBehaviour {
    public GameObject scorePanel;
    public GameObject giveupButton;
    public GameObject player;
    public GameObject enemy;
    public Transform spawnPlayer;
    public Text defaultText;
    public EnemyManager[] enemies;
    public float waitDelay;
    public AudioSource mainSOU;
    public AudioClip bgm;
    public AudioClip winClip;
    public AudioClip loseClip;
    public static int lengthFire;

    private List<GameObject> playerCheck = new List<GameObject>();
    private Scene scene;
    private int currentSceneIndex;
    private PlayerControl controlPlayer;

       // Use this for initialization
       void Start () {


        lengthFire = 1;

        mainSOU.clip = bgm;
        mainSOU.Play();
        scene = SceneManager.GetActiveScene();
        currentSceneIndex = scene.buildIndex;
        defaultText.text = string.Empty;

        SpawnPlayer();
        SetupPlayer();


        SpawnAllEnemies();
        StartCoroutine(StagePlay());
       }


    IEnumerator StagePlay()
    {
        yield return StartCoroutine(StartingPlay());
        yield return StartCoroutine(Playing());
        if (AllEnemyDie())
        {
            yield return StartCoroutine(GoodEnding());
        }
        else if(PlayerDie())
        {
            yield return StartCoroutine(BadEnding());
        }
```

```
    }
    IEnumerator StartingPlay()
    {
        defaultText.text = "Stage " + currentSceneIndex.ToString();

        ResetPlayer();

        ResetEnemy();
        DisableEnemyControl();

        yield return new WaitForSeconds(waitDelay);
    }
    IEnumerator Playing()
    {
        defaultText.text = string.Empty;
          EnablePlayerControl();
          EnableEnemyControl();

        while ( !PlayerDie() && !AllEnemyDie())
        {
            yield return null;
        }
    }
    IEnumerator GoodEnding()
    {
        mainSOU.clip = winClip;
        mainSOU.Play();
        if (currentSceneIndex == 20)
        {
            defaultText.text = "CONGRATULATION FOR WIN ALL GAME";
            yield return DecidingIfTop5();
        }
        else
            defaultText.text = "GOOD JOB";
        yield return new WaitForSeconds(waitDelay);


        if (currentSceneIndex == 5)
        {
            SceneManager.LoadScene(0);
        }
        else
        SceneManager.LoadScene(currentSceneIndex +1);
    }
    IEnumerator BadEnding()
    {
        mainSOU.clip = loseClip;
        mainSOU.Play();
        defaultText.text = "YOU LOSE";

        yield return DecidingIfTop5();


        yield return new WaitForSeconds(waitDelay);
        SceneManager.LoadScene(0);

    }

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~PLAYER~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~
```

```
    void SpawnPlayer()
    {
        GameObject a = (GameObject)Instantiate(player,
spawnPlayer.position, Quaternion.identity);
        playerCheck.Add(a);
    }

    void SetupPlayer()
    {
        controlPlayer = player.GetComponent<PlayerControl>();
    }
    void EnablePlayerControl()
    {
        controlPlayer.enabled = true;
    }

    void ResetPlayer()
    {
        player.transform.position = spawnPlayer.position;
        player.transform.rotation = spawnPlayer.rotation;
        player.SetActive(false);
        player.SetActive(true);
    }

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ENEMY~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~

void SpawnAllEnemies()
    {
        for(int i = 0; i<enemies.Length;i++)
        {
            enemies[i].e_instance = Instantiate(enemy,
enemies[i].spawnPoint.position, Quaternion.identity) as GameObject;
            enemies[i].Setup();
        }

    }


    void EnableEnemyControl()
    {
        for (int i = 0; i < enemies.Length; i++)
        {

            enemies[i].EnableControl();
        }
    }
    void DisableEnemyControl()
    {
        for (int i = 0; i < enemies.Length; i++)
        {
            enemies[i].DisableControl();
        }
    }
    void ResetEnemy()
    {
        for(int i = 0; i<enemies.Length;i++)
        {
            enemies[i].Reset();
        }
```

```
    }

//#############################################################################
##############
    bool AllEnemyDie()
    {
        int enemyDied = 0;
        for (int i = 0; i<enemies.Length; i++)
        {
            if (!enemies[i].e_instance)
                enemyDied++;
        }


                return enemyDied == enemies.Length;
    }
    bool PlayerDie()
    {
        if (!playerCheck[0])
        {
            return true;

        }
        else
            return false;
    }

//########################################Scoring#############################
######
    IEnumerator DecidingIfTop5()
    {

        giveupButton.SetActive(false);
        if(Disimpan.scoreWhenPlaying >0)
        if (BankScore.listScore.Count != 5 || BankScore.listScore[4].angka
< Disimpan.scoreWhenPlaying)
        {
            yield return new WaitForSeconds(waitDelay);
            scorePanel.SetActive(true);
            defaultText.text = string.Empty;
            mainSOU.Stop();

            while (scorePanel.activeSelf == true)
            {
                yield return null;
            }
        }
        else yield return null;
    }
    void Pause()
    {
        Time.timeScale = 0;

    }
    void Resume()
    {
        Time.timeScale = 1;
    }

}
```