Contents

1 Unity C# Functions code practice

```
using UnityEngine;
using System.Collections;

public class ActiveObjects : MonoBehaviour {

    // bgn Physics

    // rigidbody.AddForce(Vector3 direction & magnitude, [optional]Mode of force being used); /
    void OnMouseDown() {
        // AddForce() can be used in FixedUpdate() to apply a constant force
        // 4 ForceModes: Acceleration, Force, Impulse, VelocityChange
        // Acceleration & VelocityChange are NOT affected by mass
        // Force & Impulse are affected by mass
        rigidbody.AddForce(-transform.forward * 1000, ForceMode.Force);
        rigidbody.useGravity = true;
    }

    // rigidbody.AddTorque(Vector3 torque, ForceMode mode);
    // rigidbody.AddTorque(Vector3 as axis to apply torque around, [optional]Type of Force to a
    // 4 ForceModes: Acceleration, Force, Impulse, VelocityChange
    // Acceleration & VelocityChange are NOT affected by mass
    // Force & Impulse are affected by mass
    // AddTorque() is significantly affected by Angular drag, if Angular drag is increased, it
    public float amount = 50f;
    void FixedUpdate() {
        float h = Input.GetAxis("Horizontal") * amount * Time.deltaTime;
        float v = Input.GetAxis("Vertical") * amount * Time.deltaTime;
        rigidbody.AddTorque(transform.up * h, ForceMode.);
        rigidbody.AddTorque(transform.right * v);
    }

    // bgn Scripts
    public GameObject myObject;
    //int [] myInt = new int[5];
    int [] myInt = new int[] {1, 2, 3, 4, 5}; // default is private for C3, public for javascri
    public GameObject players;
    void Start() {
        Debug.Log("Active Self: " + myObject.activeSelf);
        Debug.Log("myObject.activeInHierarchy: " + myObject.activeInHierarchy);
        gameObject.SetActive(false);

        myInt[0] = 12;

        players = GameObject.FindGameObjectsWithTag("Player");
        for (int i = 0; i < players.Length; i++) {
            Debug.Log("Player number " + i + " is named " + players[i].name);
        }
    }

    public float speed = 8f;
    public float countdown = 3f;
    void Update() {
```

```
    countdown -= Time.deltaTime;
    if (countdown <= 0.0f)
        light.enabled = true;
    if (Input.GetKey(KeyCode.RightArrow))
        transform.position += new Vector3(speed * Time.deltaTime, 0f, 0f);
}

public GameObject other;
void Update() {
    if (Input.GetKey(KeyCode.Space)) {
        // Destory(gameObject, delayTime);
        // remove entire gameObject
        Destroy(other, 3); // 3 second delayTime
        // remove components
        Destroy(gameObject.GetComponent<MeshRenderer>());
     }
}
// Activate Deactivate
void Update() {
    if (Input.GetKey(KeyCode.Space))
        myLight.enabled = !myLight.enabled;
}

public GameObject otherGameObject;
private AnotherScript anotherScript;
private YetAnotherScript yetAnotherScript;
private BoxCollider boxCol;
void Awake() {
    anotherScript = GetComponent<AnotherScript>();
    yetAnotherScript = otherGameObject.GetComponent<YetAnotherScript>();
    boxCol = otherGameObject.GetComponent<BoxCollider>();
}
void Start() {
    boxCol.size = new Vector3(3, 3, 3);
}

private Vector3 newPosition;
private float newIntensity;
public float smooth = 2;
void Awake() {
    newIntensity = light.intensity;
}
void Update() {
    PositionChanging();
    IntensityChanging();
}
void PositionChanging() {
    Vector3 positionA = new Vector3(-5, 3, 0);
    Vector3 positionB = new Vector3(5, 3, 0);
    if (Input.GetKeyDown(KeyCode.Q))
        newPosition = positionA;
    if (Input.GetKeyDown(KeyCode.E))
        newPosition = positionB;
    // Vector3.Lerp(from Vector3, to Vector3, time float)
    transform.position = Vector3.Lerp(transform.position, newPosition, time.deltaTime * smo
}
void IntensityChanging() {
    float intensityA = 0.5f;
```

```
        float intensityB = 5f;
        if (Input.GetKeyDown(KeyCode.A))
            newIntensity = intensityA;
        if (Input.GetKeyDown(KeyCode.D))
            newIntensity = intensityB;
        // Mathf.Lerp(from float, to float, time float)
        light.intensity = Mathf.Lerp(light.intensity, newIntensity, Time.deltaTime * smooth);

        // Similarly, Color.Lerp(from Color, to Color, time float)
        light.color = Color.Lerp(light.color, newColor, Time.deltaTime * smooth);
    }

    void Update(){
        if (Input.GetKeyDown(KeyCode.R))
            gameObject.renderer.material.color = Color.red;
    }

    public float speed = 10f;
    void Update(){
        // transform.Rotate(Axis around which to rotate, amount to rotate by)
        transform.Rotate(Vector3.up, speed * Time.deltaTime); // (0, 1, 0)
        if (Input.GetKey(KeyCode.UpArrow))
            transform.Translate(Vector3.forward * speed * Time.deltaTime); // (0, 1, 0)
        if (Input.GetKey(KeyCode.DownArrow))
            transform.Translate(-Vector3.forward * speed * Time.deltaTime); // (0, 1, 0)
    }

    // VectorA: (x, y, z)
    // VectorB: (x, y, z)
    // Vector3.Dot(VectorA, VectorB)  // Unity function
    // (Ax * Bx) + (Ay * By)+ (Az * Bz) = Dot Product

    // Cross product, L 于原来的两个向量
    // Vector3.Cross(VectorA, VectorB) // Unity function
}
```