PROGRAM 7

write a simple unity 3D program to design 3D environment and animation

🎮 Step-by-Step Unity 3D Program

📁 *1. Scene Setup*

1. **Create a new 3D project** in Unity.
2. **Add Terrain:**
    - Right-click in Hierarchy → 3D Object → Terrain
3. **Add a Cube as a Character:**
    - Right-click in Hierarchy → 3D Object → Cube
    - Rename it to `Player`
    - Reset its transform and move it slightly above the terrain (e.g., `Y = 1`)
4. **Add a Main Camera** and position it to look at the Player.

🎨 *2. Create a Simple Animation*

1. **Add an Animator:**
    - Select the `Player` cube → Add Component → Animator
2. **Create an Animator Controller:**
    - Assets → Create → Animator Controller → Name it `PlayerController`
    - Drag `PlayerController` to the `Animator` slot on the Player
3. **Create Idle Animation:**

- Assets → Right-click → Create → Animation → Call it `Idle`
- Open the Animation window: Window → Animation → Animation
- With Player selected and `Idle` open, click "Add Property" → Transform → Rotation → Add
- Add a simple rotation (e.g., Y-axis rotate 0 → 30 → 0 over 1 second)
- Save it

4. **Set Idle as Default State** in the Animator

⚙ *3. Movement Script with Animation*

Create a C# script `PlayerMovement.cs`:

```
using UnityEngine;

public class PlayerMovement : MonoBehaviour
{
    public float speed = 5f;
    private Animator animator;

    void Start()
    {
        animator = GetComponent<Animator>();
    }

    void Update()
    {
        float h = Input.GetAxis("Horizontal");
        float v = Input.GetAxis("Vertical");

        Vector3 move = new Vector3(h, 0, v);
        transform.Translate(move * speed * Time.deltaTime, Space.World);

        // Trigger animation when moving
        if (move.magnitude > 0)
            animator.Play("Idle");  // We play Idle as a placeholder for any animation
    }
}
```

Attach the script to the `Player` cube.

ADDITIONAL

To make the **Player rotate continuously** in Unity, you can add simple rotation logic in the `Update()` method using `Transform.Rotate()`.

```
void Update()
{
   // Continuous rotation around Y-axis
   transform.Rotate(0, 50f * Time.deltaTime, 0);  // 50 is the rotation speed in degrees/sec

   float h = Input.GetAxis("Horizontal");
   float v = Input.GetAxis("Vertical");

   Vector3 move = new Vector3(h, 0, v);
   transform.Translate(move * speed * Time.deltaTime, Space.World);

   // Optional: trigger animation if needed
   if (move.magnitude > 0)
      animator.Play("Idle");
}
```

PROGRAM 8

Developing a camera, Physics and core game mechanics for 3D games.

🧱 STEP 1: Setup the Scene

1. **Create a New 3D Project in Unity.**

2. **Add a Terrain:**

   - **In Hierarchy: Right-click → `3D Object → Terrain`.**

3. **Add a Player (Cube):**

   - **Right-click in Hierarchy → `3D Object → Cube`.**
   - **Rename it to `Player`.**
   - **Set its position to `X: 0, Y: 1, Z: 0`.**

4. **Add a Main Camera (if not already):**
    - Select the `Main Camera` in the Hierarchy.
    - Move it so it can see the player (e.g., `X: 0, Y: 5, Z: -10`).
    - Rotate it to look down a bit (`Rotation X: 20`).

🎮 STEP 2: Player Movement Script

1. **Create a Script:**
    - Assets → Right-click → `Create → C# Script` → Name it `PlayerMovement`.
2. **Add This Code:**

```
using UnityEngine;

public class PlayerMovement : MonoBehaviour
{
    public float moveSpeed = 5f;
    public float jumpForce = 5f;

    private Rigidbody rb;
    private bool isGrounded;

    void Start()
    {
        rb = GetComponent<Rigidbody>();
    }

    void Update()
```

```
    {
        float moveX = Input.GetAxis("Horizontal");
        float moveZ = Input.GetAxis("Vertical");

        Vector3 move = new Vector3(moveX, 0f, moveZ) * moveSpeed;
        Vector3 newVelocity = new Vector3(move.x, rb.velocity.y, move.z);
        rb.velocity = newVelocity;

        if (Input.GetKeyDown(KeyCode.Space) && isGrounded)
        {
            rb.AddForce(Vector3.up * jumpForce, ForceMode.Impulse);
            isGrounded = false;
        }
    }

    void OnCollisionEnter(Collision collision)
    {
        if (collision.gameObject.CompareTag("Ground"))
        {
            isGrounded = true;
        }
    }
}
```

0. Add Components:

- Select `Player` in Hierarchy.
- Click `Add Component → Rigidbody`.
- Check `Freeze Rotation` on X and Z axes to prevent tipping.
- Attach `PlayerMovement` script.

0. **Tag the Terrain as "Ground":**
   - Select `Terrain` → In the Inspector, click "Tag" → Add new tag `Ground`.
   - Assign it to the Terrain.

🎥 STEP 3: Camera Follow Script

1. **Create Script:**
   - Assets → Right-click → `Create → C# Script` → Name it `CameraFollow`.
2. **Add This Code:**

```
using UnityEngine;

public class CameraFollow : MonoBehaviour
{
    public Transform target;
    public Vector3 offset;

    void LateUpdate()
    {
        transform.position = target.position + offset;
```

```
        transform.LookAt(target);
    }
}
```

Set Camera to Follow Player:

- **Attach `CameraFollow` script to the `Main Camera`.**
- **Drag the `Player` into the `Target` field.**
- **Set Offset to something like: `X: 0, Y: 5, Z: -10`.**