Código Secundario.

Player:

Este código fue hecho para que el jugador pueda moverse en este juego

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

public class Player : MonoBehaviour
{
    Transform cam;
    CharacterController control;

    public float speedCam;
    private float camRotation = 0f;
    public float speed;
    public float sprintSpeed;
    public float gravityForce;
    private float gravityMove = 0f;
    public float jumpForce;

    public TextMeshProUGUI countText;
    private int count = 0;
    private int dead = 0;
    public GameObject WinText;

    public AudioSource pasos;
    private bool HActivo;
    private bool VActivo;

    private void Start()
    {
        cam = transform.GetChild(0).GetComponent<Transform>();
        control = GetComponent<CharacterController>();

        Cursor.lockState = CursorLockMode.Locked;
    }

    private void Update()
    {
        float mouseX = Input.GetAxis("Mouse X");
        float mouseY = Input.GetAxis("Mouse Y");

        transform.Rotate(new Vector3(0, mouseX, 0) * speedCam * Time.deltaTime);

        camRotation -= mouseY * speedCam * Time.deltaTime;
        camRotation = Mathf.Clamp(camRotation, -90, 90);
```

```
        cam.localRotation = Quaternion.Euler(new Vector3(camRotation, 0, 0));

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

        Vector3 movement = Vector3.zero;

        if (Input.GetKey(KeyCode.LeftShift) && (moveX != 0 || moveZ != 0))
        {
            movement = (transform.right * moveX + transform.forward * moveZ)
            * sprintSpeed * Time.deltaTime;
        }
        else
        {
            movement = (transform.right * moveX + transform.forward * moveZ)
            * speed * Time.deltaTime;
        }

            control.Move(movement);

        control.Move(new Vector3(0, gravityMove, 0) * Time.deltaTime);

        if(!control.isGrounded)
        {
            gravityMove += gravityForce;
        }
        else
        {
            gravityMove = 0f;
        }

        if(Input.GetKeyDown(KeyCode.Space) && control.isGrounded)
        {
            gravityMove = jumpForce;
        }

        if(Input.GetButtonDown("Horizontal")) //Reproduce audio de pisadas
        {
            if(VActivo == false)
            {
                HActivo = true;
                pasos.Play();
            }

        }
        if(Input.GetButtonDown("Vertical"))
        {
            if(HActivo == false)
            {
                VActivo = true;
                pasos.Play();
            }

        }

        if(Input.GetButtonUp("Horizontal"))
```

```
        {
            HActivo = false;
            if(VActivo == false)
            {
                pasos.Pause();
            }

        }
        if (Input.GetButtonUp("Vertical"))
        {
            VActivo = false;
            if(HActivo == false)
            {
                pasos.Pause();
            }

        }
    }

    void OnTriggerEnter(Collider other)
    {


        if (other.gameObject.CompareTag("punto"))
        {
            count++;

            other.gameObject.SetActive(false);
            if (count == 113)
            {
                WinText.SetActive(true);
            }
        }

    }

}
```

Nota: Tiene un contador oculto para la recolección de puntos.

MenuInicial:

Este código fue creado para el inicio del juego se pueda manipular de manera sencilla.

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

public class MenuInicial : MonoBehaviour
{
```

```
public void jugar()
    {
        SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex + 1);

    }

public void salir()
    {
        Debug.Log("salir");
        Application.Quit();
    }
}
```

Pausa:

Este código fue creado para poder pausar el juego con la letra escape y reanudar el juego con la misma tecla.

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

public class Pausa : MonoBehaviour
{
    [SerializeField] private GameObject botonPausa;
    [SerializeField] private GameObject menuPausa;

    private bool juegoPausado = false;

    private void Update()
    {
        if(Input.GetKeyDown(KeyCode.Escape))
        {
            if(juegoPausado)
            {
                Reanudar();
            }
            else
            {
                Pausar();
            }
        }
    }
    public void Pausar()
    {
        juegoPausado = true;
        Time.timeScale = 0f;
        botonPausa.SetActive(false);
        menuPausa.SetActive(true);
    }
```

```
    public void Reanudar()
    {
        juegoPausado = false;
        Time.timeScale = 1f;
        botonPausa.SetActive(true);
        menuPausa.SetActive(false);

    }
}
```

Puntos:

Este código fue creado para que los puntos a recolectar tenga un movimiento sobre su propio eje

```
using UnityEngine;

public class Puntos : MonoBehaviour
{
    public float velocidadRotacionX = 10f; // Velocidad de rotación en el eje X
    public float velocidadRotacionY = 20f; // Velocidad de rotación en el eje Y

    void Update()
    {
        // Rotar el objeto en su eje Y y eje X a las velocidades especificadas
        transform.Rotate(Vector3.up, velocidadRotacionY * Time.deltaTime);
        transform.Rotate(Vector3.right, velocidadRotacionX * Time.deltaTime);
    }
}
```

Intro:

Este código fue hecho para la intro del juego pase un fantasma por el fondo.

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

public class Intro : MonoBehaviour
{
    public float velocidadMovimiento = 1.0f;
    public float distanciaRecorrido = 10.0f;
    private float tiempoActual = 0.0f;
    private int direccion = 1; // Dirección de movimiento del NPC (-1 o 1)
```

```
    void Update()
    {
        // Mover el NPC en el eje Z en la dirección actual
        transform.Translate(Vector3.forward * velocidadMovimiento * direccion * Time.deltaTime);

        // Actualizar el tiempo actual transcurrido
        tiempoActual += Time.deltaTime;

        // Comprobar si el NPC ha recorrido la distancia total en la dirección actual
        if (Mathf.Abs(transform.position.z) >= distanciaRecorrido / 2)
        {
            // Cambiar la dirección del NPC para que regrese por el mismo eje
            direccion *= -1;
        }

        // Comprobar si ha pasado el tiempo de regreso
        if (tiempoActual >= 10.0f)
        {
            // Reiniciar el tiempo actual
            tiempoActual = 0.0f;

            // Cambiar la dirección del NPC para que se de media vuelta
            direccion *= -1;
        }
    }
}
```