MAIN

```
using Godot;
using System;

public partial class Main : Node2D
{
    private int score = 0; // Score tracker
    private Label scoreLabel;
    private Label endgameLabel;

    public override void _Ready()
    {
        // Get references to the labels
        scoreLabel = GetNode<Label>("ScoreLabel");
        endgameLabel = GetNode<Label>("EndgameLabel");

        // Connect the star signal to the score update method
        GetNode<Star>("Star").Connect("StarCollected", this, nameof(OnStarCollected));
    }

    private uint nameof(Action onStarCollected)
    {
        throw new NotImplementedException();
    }

    private void OnStarCollected()
    {
        score += 1; // Increase the score
        scoreLabel.Text = "Score: " + score;

        if (score >= 5) // End game when score reaches 5
        {
            endgameLabel.Text = "You Win!";
            endgameLabel.Visible = true;
        }
    }
}
```

Player

```
using Godot;

public partial class Player : CharacterBody2D
{
    [Export] private float speed = 200f; // Speed of the player
    [Export] private float jumpStrength = 400f; // Jump strength

    public override void _PhysicsProcess( double delta) // Corrected to 'float' instead of 'double'
    {
        // Horizontal movement
        Vector2 direction = Vector2.Zero;

        // Move right
        if (Input.IsActionPressed("ui_right"))
        {
            direction.X = 1;
        }
        // Move left
        if (Input.IsActionPressed("ui_left"))
        {
            direction.X = -1;
        }

        // Apply movement to the character
        Velocity = new Vector2(direction.X * speed, Velocity.Y);

        // Check for jump (only if the player is on the floor)
        if (IsOnFloor() && Input.IsActionJustPressed("ui_up"))
        {
            Velocity = new Vector2(Velocity.X, -jumpStrength);
        }

        // Move the player based on the velocity
        MoveAndSlide();
    }

    // Signal connection to handle star collection
    private void _on_body_entered(Node body)
    {
        if (body is Star) // Ensure collision is with a Star
        {
            EmitSignal("StarCollected"); // Emit a signal when the star is collected
            body.QueueFree(); // Remove the star from the scene
        }
    }
}
```

Obstacle 1

```
using Godot;

public partial class Obstacle : Node2D
{
    [Export] private int Speed = 150; // Obstacle speed

    public override void _Process(float delta) // Override the _Process method
    {
        // Move obstacle to the left
        Position += new Vector2(-Speed * delta, 0);

        // Reset position when off-screen
        if (Position.X < -100)
        {
            Position = new Vector2(800, Position.y); // Reset to right side of screen
        }
    }
}
```

Obstacle 2

```
using Godot;

public partial class Obstacle : Node2D
{
    [Export] private int speed = 150; // Obstacle speed

    public override void _PhysicsProcess(float delta)
    {
        // Move obstacle to the left
        Position += new Vector2(-speed * delta, 0);

        // Reset position when off-screen
        if (Position.X < -100)
        {
            Position = new Vector2(800, Position.y); // Reset to right side of screen
        }
    }
}
```

Stars

```
using Godot;
using System;

public partial class Star : Area2D
{
    [Signal] public delegate void StarCollectedEventHandler(); // Change delegate name to follow
the convention

    private void _on_body_entered(Node body)
    {
        if (body is Player) // Check if it's the player
        {
            EmitSignal(nameof(StarCollectedEventHandler)); // Emit signal for collecting the star
            QueueFree(); // Remove the star from the scene
        }
    }

    internal void Connect(string v1, Main main, uint v2)
    {
        throw new NotImplementedException();
    }
}
```