```
extends Node
var score = 0
func add_point():
	score += 1
	print(score)



extends CharacterBody2D


const SPEED = 130.0
const JUMP_VELOCITY = -300.0

# Get the gravity from the project settings to be synced with RigidBody nodes.
var gravity = ProjectSettings.get_setting("physics/2d/default_gravity")

@onready var animated_sprite_2d = $AnimatedSprite2D

func _physics_process(delta):
	# Add the gravity.
	if not is_on_floor():
		velocity.y += gravity * delta

	# Handle jump.
	if Input.is_action_just_pressed("jump ") and is_on_floor():
		velocity.y = JUMP_VELOCITY

	# Get the input direction and handle the movement/deceleration.
	# As good practice, you should replace UI actions with custom gameplay actions.
	var direction = Input.get_axis("move_left", "move_right")
	if direction > 0:
		animated_sprite_2d.flip_h = false
	elif direction < 0:
		animated_sprite_2d.flip_h = true
	if is_on_floor():
		if direction == 0:
			animated_sprite_2d.play("Idle")
		else:
			animated_sprite_2d.play("run" )
	else:
		animated_sprite_2d.play("jumping")
	if direction:
		velocity.x = direction * SPEED
```

```
	else:
		velocity.x = move_toward(velocity.x, 0, SPEED)

	move_and_slide()
```

```
extends Area2D

@onready var game_manager = %GameManager

@onready var animation_player = $AnimationPlayer

func _on_body_entered(_body):
	game_manager.add_point()
	animation_player.play("pickup")
```

```
extends Area2D

@onready var timer = $Timer

func _on_body_entered(body):
	print("You died")
	Engine.time_scale = 0.5
	body.get_node("CollisionShape2D").queue_free()
	timer.start()

func _on_timer_timeout():
	Engine.time_scale = 1.0
	get_tree().reload_current_scene()
```

```
extends Node2D
```

```
const speed = 60

var direction = 1
@onready var ray_cast_right = $RayCastRight
@onready var ray_cast_left = $RayCastLeft
@onready var animated_sprite_2d = $AnimatedSprite2D


# Called every frame. 'delta' is the elapsed time since the previous frame.
func _process(delta):
	if ray_cast_right.is_colliding():
		direction = -1
		animated_sprite_2d.flip_h = true
	if ray_cast_left.is_colliding():
		direction = 1
		animated_sprite_2d.flip_h = false
	position.x += direction * speed * delta
```