Appendix

Table of Contents:

Appendix 1: Initial Interview with F

Me: Hi, how are you?

F: Good, you?

Me: Good as well, how are practices going for you?

F: Not bad, but I feel like my hand-eye coordination is holding me back from being able to catch the ball better, I'm having trouble remembering the plays and what to do, and I feel like I'm not reacting on the snap very quickly, and all this adding up could be really bad.

Me: Ok, so you're having trouble with hand-eye coordination, memory, and reaction time.

F: Yeah, that seems right.

Me: Ok let me see if I can help.

Appendix 2: Second Interview with F

Me: Ok, I have an idea of what could help you out, but I need to some questions first.

F: Ok, hit me.

Me: So what's the operating system of the computer you use?

F: It's a Macbook.

Me: Ok good, do you think having 5 different exercises that can test your hand-eye coordination, memory, and reaction time, would be sufficient?

F: Yeah that sounds like more than enough, just make them engaging so that I don't get bored doing them.

Me: Ok last one, do you think I should add a way to save your data, or do you think that's unnecessary?

F: I think it would be really helpful to be able to save, I don't think I'll remember my scores.

Appendix 3: Post-Design Interview

Me: So I've completed all the flowcharts for the app, including all the exercises, navigation systems, and scoring systems, and I'd like to show them to you and get some feedback.

F: All the exercises look good and well thought out, they would be able to test their respective skill pretty well.

Me: Ok great! What about the navigation system, does it seem too much or overwhelming?

F: The chart seems a little overwhelming, but I'm sure it'll look good within the app.

Me: Ok, that's good. How about the scoring?

F: The scoring system makes sense. Overall, I have no issues with it!

Me: Ok, thank you!

Appendix 4: Client Review of Final Product

Me: Hi again, I've finished the app and want to see what you think!

F: Ok great! At first glance, it looks pretty nice, and I like the responsiveness of all the buttons and the navigation.

Me: Ok, take a look at the exercises.

F: I like that there's a good chunk of exercises, but it seems mostly memory-based.

Me: Ok, pick one exercise to do.

F: Ok, I'll try the reaction time test. I like how it gives me 5 tries instead of just 3 or something, so it gives me a more accurate result of my reaction time. I also noticed that if I pressed too early it makes me retry, and when get my final result, it was in milliseconds, and looks better than if it was in seconds with decimal places.

Me: Now try going back to the dashboard menu and seeing your result there.

F: Oh, so the result I got in that test I just did I can also see in the dashboard.
Me: Yeah, now try saving your results and exiting the app, to test the saving mechanisms.

F: Ok, so I press the save button, then exit, then when I reenter the app and go into the dashboard and press load, my score from before reappears! And when I press the delete button my score goes away.

Me: Exactly, now you can just go through each test and tell me what you think.

F: Ok, next up, the sequence memory test. Ok, I think I like the sequence memory test the most, mainly because of the difficulty of getting a higher score, and it was much more challenging than I expected because I did much worse than I thought I would.

F: When I started the aim trainer test, the first thing I noticed was the "Remainder: #" text where it constantly updated the number of targets I had left, which I thought was nice. And when I was done it showed my score in milliseconds just like in the reaction time test.

F: The language memory test I thought was also pretty enjoyable, and there was a lot words, the amount of words never seemed to run out. Whether it showed a new or seen word seemed completely random, and when I ran out of my 3 lives it showed my final score.

F: I like the number memory test the number changed every time but the number of digits increased every time. All the numbers I pressed were inputted correctly, and the enter and delete keys worked as intended. I wasn't able to input anything other than numbers.

Me: Ok thank you for all your feedback!

Appendix 5: Bibliography

Rock Milk. "Why we choose Godot Engine." *Medium*,

medium.com/rock-milk/why-godot-engine-e0d4736d6eb0. Accessed 20 Dec. 2024.

Appendix 6: Code

Main:

```
extends Node

# Called when the node enters the scene tree for the first time.
func _ready():
	Global.save_path = "user://ia_data.save"


# Called every frame. 'delta' is the elapsed time since the previous frame.
func _process(delta):
	pass


func _on_games_button_pressed(): # Switches to exercises scene
	get_tree().change_scene_to_file("res://scenes/exercises.tscn")


func _on_dashboard_button_pressed(): # Switches to dashboard scene
	get_tree().change_scene_to_file("res://scenes/dashboard.tscn")


func _on_quit_button_pressed():
	save()
	get_tree().quit() # Exits app


func save():
	Global.save_path = "user://ia_data.save"
	var file = FileAccess.open(Global.save_path, FileAccess.WRITE)
	file.store_var(Global.reaction_time)
	file.store_var(Global.aim_trainer)
	file.store_var(Global.sequence_memory)
	file.store_var(Global.verbal_memory)
```

```
        file.store_var(Global.number_memory)


func _on_save_button_pressed():
        save()
```

Dashboard:

```
extends Node


# Called when the node enters the scene tree for the first time.
func _ready():
        pass


# Called every frame. 'delta' is the elapsed time since the previous frame.
func _process(delta): # Will update the data shown in dashboard
        if Global.reaction_time >= 10000000000000:
                $Labels/ReactionTimeTestLabel.text = "N/A"
        else:
                $Labels/ReactionTimeTestLabel.text = str(Global.reaction_time) + " ms"
        if Global.sequence_memory <= 0:
                $Labels/SequenceMemoryTestLabel.text = "N/A"
        else:
                $Labels/SequenceMemoryTestLabel.text = str(Global.sequence_memory)
        if Global.aim_trainer >= 10000000000000:
                $Labels/AimTrainerLabel.text = "N/A"
        else:
                $Labels/AimTrainerLabel.text = str(Global.aim_trainer) + " ms"
        if Global.number_memory <= 0:
                $Labels/VisualMemoryLabel.text = "N/A"
        else:
                $Labels/VisualMemoryLabel.text = str(Global.number_memory)
        if Global.verbal_memory <= 0:
                $Labels/VerbalMemoryLabel.text = "N/A"
        else:
                $Labels/VerbalMemoryLabel.text = str(Global.verbal_memory)


# Sends user to chosen scene
func _on_back_button_pressed():
```

```
        get_tree().change_scene_to_file("res://scenes/main.tscn")


func _on_play_button_pressed():
        get_tree().change_scene_to_file("res://scenes/reactiontime.tscn")


func _on_play_button_2_pressed():
        get_tree().change_scene_to_file("res://scenes/sequencememory.tscn")


func _on_play_button_3_pressed():
        get_tree().change_scene_to_file("res://scenes/aimtrainer.tscn")


func _on_play_button_4_pressed():
        get_tree().change_scene_to_file("res://scenes/number_memory.tscn")


func _on_play_button_5_pressed():
        get_tree().change_scene_to_file("res://scenes/verbalmemory.tscn")


func load_data():
                if FileAccess.file_exists(Global.save_path):
                        var file = FileAccess.open(Global.save_path, FileAccess.READ)
                        Global.reaction_time = file.get_var(Global.reaction_time)
                        Global.aim_trainer = file.get_var(Global.aim_trainer)
                        Global.sequence_memory = file.get_var(Global.sequence_memory)
                        Global.verbal_memory = file.get_var(Global.verbal_memory)
                        Global.number_memory = file.get_var(Global.number_memory)
                else: # No file exists
                        Global.reaction_time = 10000000000000
                        Global.aim_trainer = 10000000000000
                        Global.sequence_memory = 0
                        Global.verbal_memory = 0
                        Global.number_memory = 0


func delete_data():
```

```
	Global.save_path = ""
	Global.reaction_time = 10000000000000
	Global.aim_trainer = 10000000000000
	Global.sequence_memory = 0
	Global.verbal_memory = 0
	Global.number_memory = 0


func _on_delete_button_pressed():
	delete_data()


func _on_load_button_pressed():
	load_data()


func _on_save_button_pressed():
	Global.save()
```

Exercises:

```
extends Node


# Called when the node enters the scene tree for the first time.
func _ready():
	pass # Replace with function body.


# Called every frame. 'delta' is the elapsed time since the previous frame.
func _process(delta):
	pass

# Sends user to chosen scene
func _on_back_button_pressed():
	get_tree().change_scene_to_file("res://scenes/main.tscn")


func _on_number_button_pressed():
	get_tree().change_scene_to_file("res://scenes/number_memory.tscn")
```

```
func _on_verbal_button_pressed():
	get_tree().change_scene_to_file("res://scenes/verbalmemory.tscn")


func _on_aim_button_pressed():
	get_tree().change_scene_to_file("res://scenes/aimtrainer.tscn")


func _on_sequence_button_pressed():
	get_tree().change_scene_to_file("res://scenes/sequencememory.tscn")


func _on_reaction_button_pressed():
	get_tree().change_scene_to_file("res://scenes/reactiontime.tscn")
```

Aim Trainer:

```
extends Node


# Called when the node enters the scene tree for the first time.
func _ready():
	pass # Replace with function body.


# Called every frame. 'delta' is the elapsed time since the previous frame.
func _process(delta):
	pass


func _on_back_button_pressed():
	get_tree().change_scene_to_file("res://scenes/exercises.tscn")


func _on_start_button_pressed():
	get_tree().change_scene_to_file("res://scenes/aimtrainertest.tscn")
```

Aim Trainer Test:

```
extends Node

var targets: int = 30
var try: float = 0 # each individual try to be added together
var total: float = 0 # sum of all tries^
var ave: float = 0 # average of all tries (total/ 30)
var start: bool = false


# Called when the node enters the scene tree for the first time.
func _ready():
	$EndTitle.visible = false
	$Target.visible = false


# Called every frame. 'delta' is the elapsed time since the previous frame.
func _process(_delta):
	if targets < 1:
		end()

	if start == true:
		$Remaining.text = "Remaining: " + str(targets)


func _on_start_button_pressed():
	start = true
	$StartButton.queue_free()
	$BackButton.visible = false
	$BackButton.disabled = true
	$Title.visible = false
	$SubTitle.visible = false
	$Remaining.text = "Remaining: " + str(targets)
	draw_button()


func _on_back_button_pressed():
	get_tree().change_scene_to_file("res://scenes/aimtrainer.tscn")


func draw_button():
```

```
        $TryTimer.start()
        $Target.visible = true
        $Target.position.x = randf_range(($Target.size.x / 2),
(get_window().size.x-$Target.size.x)) # Get x position
        $Target.position.y = randf_range(($Target.size.y / 2),
(get_window().size.y-$Target.size.y)) # Get y position


func _on_target_pressed(): # When target is pressed
        try = int((4096-$TryTimer.time_left)*1000)
        total += try
        targets -= 1
        draw_button()


func end(): # Ends exercises and displays results
        start = false

        $Target.visible = false
        $Target.disabled = true

        ave = int(total/30)
        $SubTitle.visible = true
        $SubTitle.text = "Average: " + str(ave) + " ms"

        $BackButton.visible = true
        $BackButton.disabled = false

        $Remaining.visible = false

        $EndTitle.visible = true

        Global.aim_trainer_test_score(ave)
        Global.save()
```

Global:

```
extends Node

var reaction_time = 10000000000000
var sequence_memory = 0
```

```
var aim_trainer = 10000000000000
var number_memory = 0
var verbal_memory = 0

var save_path = "user://ia_data.save"

# Called when the node enters the scene tree for the first time.
func _ready():
	pass # Replace with function body.


# Called every frame. 'delta' is the elapsed time since the previous frame.
func _process(delta):
	pass

# Called at end of respective exercises
func reaction_time_test_score(i):
	if i < reaction_time:
		reaction_time = i


func sequence_memory_test_score(i):
	if i > sequence_memory:
		sequence_memory = i


func aim_trainer_test_score(i):
	if i < aim_trainer:
		aim_trainer = i


func verbal_memory_test_score(i):
	if i > verbal_memory:
		verbal_memory = i


func number_memory_test_score(i):
	if i > number_memory:
		number_memory = i
```

```
func save():
	save_path = "user://ia_data.save"
	var file = FileAccess.open(Global.save_path, FileAccess.WRITE)
	file.store_var(Global.reaction_time)
	file.store_var(Global.aim_trainer)
	file.store_var(Global.sequence_memory)
	file.store_var(Global.verbal_memory)
	file.store_var(Global.number_memory)
```

Number Memory:

```
extends Node


# Called when the node enters the scene tree for the first time.
func _ready():
	pass # Replace with function body.


# Called every frame. 'delta' is the elapsed time since the previous frame.
func _process(delta):
	pass


func _on_start_button_pressed():
	get_tree().change_scene_to_file("res://scenes/numbermemorytest.tscn")


func _on_back_button_pressed():
	get_tree().change_scene_to_file("res://scenes/exercises.tscn")
```

Number Memory Test:

```
extends Node

var start: bool = false
var take_input: bool = false
var level: int = 1
var num: int
var num_list: Array = []
```

```
# Called when the node enters the scene tree for the first time.
func _ready():
	$Countdown.visible = false


	# Called every frame. 'delta' is the elapsed time since the previous frame.
func _process(delta):
	if start == true:
		$LevelLabel.text = "Level: " + str(level)
		if take_input == true: # Allows or disallows user to input
			input()
		if take_input == false:
			$Countdown.value = $Timer.time_left



func _on_start_button_pressed():
	start = true

	$StartButton.queue_free()
	$BackButton.disabled = true

	$LevelLabel.text = "Level: " + str(level)

	generate_num()


func _on_back_button_pressed():
	get_tree().change_scene_to_file("res://scenes/number_memory.tscn")


func generate_num():
	take_input = false
	$InputLabel.text = ""
	num = randi_range(pow(10, (level-1)), pow(10, level) - 1) # Generates the number
	print(str(num))
	$NumberLabel.text = str(num)
	$Timer.start()
	$Countdown.visible = true
```

```
	$Countdown.value = $Timer.time_left


func _on_timer_timeout(): # No more time memorizing
	$InputLabel.text = "Number: "
	take_input = true


func input():
	$NumberLabel.text = ""
	$Countdown.visible = false

	if Input.is_action_just_pressed("0"):
		num_list.append(0)
		$InputLabel.text = "Number: " + array_to_string(num_list)
	if Input.is_action_just_pressed("1"):
		num_list.append(1)
		$InputLabel.text = "Number: " + array_to_string(num_list)
	if Input.is_action_just_pressed("2"):
		num_list.append(2)
		$InputLabel.text = "Number: " + array_to_string(num_list)
	if Input.is_action_just_pressed("3"):
		num_list.append(3)
		$InputLabel.text = "Number: " + array_to_string(num_list)
	if Input.is_action_just_pressed("4"):
		num_list.append(4)
		$InputLabel.text = "Number: " + array_to_string(num_list)
	if Input.is_action_just_pressed("5"):
		num_list.append(5)
		$InputLabel.text = "Number: " + array_to_string(num_list)
	if Input.is_action_just_pressed("6"):
		num_list.append(6)
		$InputLabel.text = "Number: " + array_to_string(num_list)
	if Input.is_action_just_pressed("7"):
		num_list.append(7)
		$InputLabel.text = "Number: " + array_to_string(num_list)
	if Input.is_action_just_pressed("8"):
		num_list.append(8)
		$InputLabel.text = "Number: " + array_to_string(num_list)
	if Input.is_action_just_pressed("9"):
```

```
                num_list.append(9)
                $InputLabel.text = "Number: " + array_to_string(num_list)
        if Input.is_action_just_pressed("backspace"):
                num_list.remove_at(num_list.size() - 1)
                $InputLabel.text = "Number: " + array_to_string(num_list)
        if Input.is_action_just_pressed("enter"):
                answer()


func array_to_string(arr: Array) -> String:
        var s = ""
        for i in arr:
                s += str(i) # Adds each integer in the array to a string
        return s


func answer():
        if array_to_string(num_list) == str(num):
                level += 1
                num_list.clear()
                generate_num()
        else:
                end()


func end():
        start = false
        $LevelLabel.text = "Congradulations you made it to level " + str(level)
        $InputLabel.text = ""
        $NumberLabel.text = "Answer: " + str(num)
        $BackButton.disabled = false
        Global.number_memory_test_score(level)
        Global.save()
```

Reaction Time:

```
extends Node


# Called when the node enters the scene tree for the first time.
```

```
func _ready():
	pass # Replace with function body.


# Called every frame. 'delta' is the elapsed time since the previous frame.
func _process(delta):
	pass


func _on_start_button_pressed():
	get_tree().change_scene_to_file("res://scenes/reactiontimetest.tscn")


func _on_back_button_pressed():
	get_tree().change_scene_to_file("res://scenes/exercises.tscn")
```

Reaction Time Test:

```
extends Node

var start: bool = false # react timer started
var react: bool = false # try timer started
var tries: int = 5 # amount of tries user gets
var try: float = 0 # each individual try to be added to times
var times: Array = [] # array of all times to find average
var total: float = 0 # total of all the times from array times
var ave: float = 0 # average of all the times from array times

# Called when the node enters the scene tree for the first time.
func _ready():
	pass # Replace with function body.


# Called every frame. 'delta' is the elapsed time since the previous frame.
func _process(delta):
	if start == false and tries > 0:
		if Input.is_action_pressed("start"):
			reaction()
	if tries == 0:
		end()
```

```
func reaction():
	$BackButton.disabled = true
	$ReactButton.text = ""
	$ReactTimer.wait_time = randf_range(3.0, 10.0)
	start = true
	$ReactTimer.start()


func _on_react_button_button_down():
	if start == true:
		if react == true:
			try = int((4096-$TryTimer.time_left)*1000) # Calculates time taken
			$ReactButton.text = str(try) + " ms"
			times.append(try)
			tries -= 1
			$ReactTimer.stop()
			$InBetweenTimer.start()
			react = false
		elif react == false:
			$ReactButton.text = "Too Soon!"
			$ReactTimer.stop()
			$InBetweenTimer.start()


func _on_react_timer_timeout():
	$TryTimer.start()
	react = true
	$ReactButton.get_theme_stylebox("hover").bg_color = Color.CRIMSON
	$ReactButton.get_theme_stylebox("normal").bg_color = Color.CRIMSON


func restart():
	$ReactButton.text = "Press Space to Start"
	$ReactButton.get_theme_stylebox("hover").bg_color = Color(1, 0.84, 0, 1)
	$ReactButton.get_theme_stylebox("normal").bg_color = Color(1, 0.84, 0, 1)
	start = false
	$InBetweenTimer.stop()
```

```
func end():
	ave = int((times[0]+times[1]+times[2]+times[3]+times[4])/ len(times))
	$ReactButton.text = "Average: " + str(ave) + " ms"
	Global.reaction_time_test_score(ave)
	Global.save()
	$BackButton.disabled = false


func _on_in_between_timer_timeout():
	restart()


func _on_back_button_pressed():
	get_tree().change_scene_to_file("res://scenes/reactiontime.tscn")
```

Sequence Memory:

```
extends Node

# Called when the node enters the scene tree for the first time.
func _ready():
	pass # Replace with function body.


# Called every frame. 'delta' is the elapsed time since the previous frame.
func _process(delta):
	pass


func _on_start_button_pressed():
	get_tree().change_scene_to_file("res://scenes/sequencememorytest.tscn")


func _on_back_button_pressed():
	get_tree().change_scene_to_file("res://scenes/exercises.tscn")
```

Sequence Memory Test:

```
extends Node

var sequence: Array = []
```

```
var pressed: Array = []
var level: int = 1
var lives: int = 3
var light_up = false # whether or not a button will light up
var done_light_up = false # whether or not the sequence is done, false = no, true = yes and user can begin pressing buttons
var start = false # whether or not the test has started
var user_turn = false # state whether or not it's the user's turn

# Called when the node enters the scene tree for the first time.
func _ready(): # Disables all buttons and generates a random first button of the sequence
	$Buttons/Button1.disabled = true
	$Buttons/Button2.disabled = true
	$Buttons/Button3.disabled = true
	$Buttons/Button4.disabled = true
	$Buttons/Button5.disabled = true
	$Buttons/Button6.disabled = true
	$Buttons/Button7.disabled = true
	$Buttons/Button8.disabled = true
	$Buttons/Button9.disabled = true
	sequence.append(randi_range(1, 9))


# Called every frame. 'delta' is the elapsed time since the previous frame.
func _process(delta):
	$LivesTitle.text = "Lives: " + str(lives)
	$SubTitle.text = "Level: " + str(level)

	if lives <= 0:
		end(level)


func _on_back_button_pressed():
	get_tree().change_scene_to_file("res://scenes/sequencememory.tscn")

# In case of any button being pressed
func _on_button_1_pressed(): # top left
	if sequence[0] != 1:
		lives -= 1
	else:
```

```
                pressed.append(sequence[0])
                sequence.remove_at(0)
        if sequence.size() == 0:
                user_turn = false
                new_level()

func _on_button_2_pressed(): # top mid
        if sequence[0] != 2:
                lives -= 1
        else:
                pressed.append(sequence[0])
                sequence.remove_at(0)
        if sequence.size() == 0:
                user_turn = false
                new_level()

func _on_button_3_pressed(): # top right
        if sequence[0] != 3:
                lives -= 1
        else:
                pressed.append(sequence[0])
                sequence.remove_at(0)
        if sequence.size() == 0:
                user_turn = false
                new_level()

func _on_button_4_pressed(): # mid left
        if sequence[0] != 4:
                lives -= 1
        else:
                pressed.append(sequence[0])
                sequence.remove_at(0)
        if sequence.size() == 0:
                user_turn = false
                new_level()

func _on_button_5_pressed(): # mid
        if sequence[0] != 5:
                lives -= 1
        else:
```

```
                pressed.append(sequence[0])
                sequence.remove_at(0)
        if sequence.size() == 0:
                user_turn = false
                new_level()

func _on_button_6_pressed(): # mid right
        if sequence[0] != 6:
                lives -= 1
        else:
                pressed.append(sequence[0])
                sequence.remove_at(0)
        if sequence.size() == 0:
                user_turn = false
                new_level()

func _on_button_7_pressed(): # bottom left
        if sequence[0] != 7:
                lives -= 1
        else:
                pressed.append(sequence[0])
                sequence.remove_at(0)
        if sequence.size() == 0:
                user_turn = false
                new_level()


func _on_button_8_pressed(): # bottom mid
        if sequence[0] != 8:
                lives -= 1
        else:
                pressed.append(sequence[0])
                sequence.remove_at(0)
        if sequence.size() == 0:
                user_turn = false
                new_level()


func _on_button_9_pressed(): # bottom right
        if sequence[0] != 9:
```

```
            lives -= 1
        else:
            pressed.append(sequence[0])
            sequence.remove_at(0)
        if sequence.size() == 0:
            user_turn = false
            new_level()


func light_up_button(i): # call everytime a button needs to light up
        done_light_up = false
        var style = StyleBoxFlat.new()
        style.bg_color = Color.FIREBRICK
        $LightUpTimer.one_shot = true
        $LightUpTimer.start()
        if i == 1:
            $Buttons/Button1.add_theme_stylebox_override("normal", style)
            $Buttons/Button1.add_theme_stylebox_override("hover", style)
            $Buttons/Button1.add_theme_stylebox_override("pressed", style)
            $Buttons/Button1.add_theme_stylebox_override("disabled", style)
        elif i == 2:
            $Buttons/Button2.add_theme_stylebox_override("normal", style)
            $Buttons/Button2.add_theme_stylebox_override("hover", style)
            $Buttons/Button2.add_theme_stylebox_override("pressed", style)
            $Buttons/Button2.add_theme_stylebox_override("disabled", style)
        elif i == 3:
            $Buttons/Button3.add_theme_stylebox_override("normal", style)
            $Buttons/Button3.add_theme_stylebox_override("hover", style)
            $Buttons/Button3.add_theme_stylebox_override("pressed", style)
            $Buttons/Button3.add_theme_stylebox_override("disabled", style)
        elif i == 4:
            $Buttons/Button4.add_theme_stylebox_override("normal", style)
            $Buttons/Button4.add_theme_stylebox_override("hover", style)
            $Buttons/Button4.add_theme_stylebox_override("pressed", style)
            $Buttons/Button4.add_theme_stylebox_override("disabled", style)
        elif i == 5:
            $Buttons/Button5.add_theme_stylebox_override("normal", style)
            $Buttons/Button5.add_theme_stylebox_override("hover", style)
            $Buttons/Button5.add_theme_stylebox_override("pressed", style)
            $Buttons/Button5.add_theme_stylebox_override("disabled", style)
```

```
	elif i == 6:
		$Buttons/Button6.add_theme_stylebox_override("normal", style)
		$Buttons/Button6.add_theme_stylebox_override("hover", style)
		$Buttons/Button6.add_theme_stylebox_override("pressed", style)
		$Buttons/Button6.add_theme_stylebox_override("disabled", style)
	elif i == 7:
		$Buttons/Button7.add_theme_stylebox_override("normal", style)
		$Buttons/Button7.add_theme_stylebox_override("hover", style)
		$Buttons/Button7.add_theme_stylebox_override("pressed", style)
		$Buttons/Button7.add_theme_stylebox_override("disabled", style)
	elif i == 8:
		$Buttons/Button8.add_theme_stylebox_override("normal", style)
		$Buttons/Button8.add_theme_stylebox_override("hover", style)
		$Buttons/Button8.add_theme_stylebox_override("pressed", style)
		$Buttons/Button8.add_theme_stylebox_override("disabled", style)
	elif i == 9:
		$Buttons/Button9.add_theme_stylebox_override("normal", style)
		$Buttons/Button9.add_theme_stylebox_override("hover", style)
		$Buttons/Button9.add_theme_stylebox_override("pressed", style)
		$Buttons/Button9.add_theme_stylebox_override("disabled", style)



func turn_off_button(i): # Changes color
	var style = StyleBoxFlat.new()
	style.bg_color = Color(1, 0.84, 0)
	if i == 1:
		$Buttons/Button1.add_theme_stylebox_override("normal", style)
		$Buttons/Button1.add_theme_stylebox_override("hover", style)
		$Buttons/Button1.add_theme_stylebox_override("pressed", style)
		$Buttons/Button1.add_theme_stylebox_override("disabled", style)
	elif i == 2:
		$Buttons/Button2.add_theme_stylebox_override("normal", style)
		$Buttons/Button2.add_theme_stylebox_override("hover", style)
		$Buttons/Button2.add_theme_stylebox_override("pressed", style)
		$Buttons/Button2.add_theme_stylebox_override("disabled", style)
	elif i == 3:
		$Buttons/Button3.add_theme_stylebox_override("normal", style)
		$Buttons/Button3.add_theme_stylebox_override("hover", style)
		$Buttons/Button3.add_theme_stylebox_override("pressed", style)
```

```
                $Buttons/Button3.add_theme_stylebox_override("disabled", style)
        elif i == 4:
                $Buttons/Button4.add_theme_stylebox_override("normal", style)
                $Buttons/Button4.add_theme_stylebox_override("hover", style)
                $Buttons/Button4.add_theme_stylebox_override("pressed", style)
                $Buttons/Button4.add_theme_stylebox_override("disabled", style)
        elif i == 5:
                $Buttons/Button5.add_theme_stylebox_override("normal", style)
                $Buttons/Button5.add_theme_stylebox_override("hover", style)
                $Buttons/Button5.add_theme_stylebox_override("pressed", style)
                $Buttons/Button5.add_theme_stylebox_override("disabled", style)
        elif i == 6:
                $Buttons/Button6.add_theme_stylebox_override("normal", style)
                $Buttons/Button6.add_theme_stylebox_override("hover", style)
                $Buttons/Button6.add_theme_stylebox_override("pressed", style)
                $Buttons/Button6.add_theme_stylebox_override("disabled", style)
        elif i == 7:
                $Buttons/Button7.add_theme_stylebox_override("normal", style)
                $Buttons/Button7.add_theme_stylebox_override("hover", style)
                $Buttons/Button7.add_theme_stylebox_override("pressed", style)
                $Buttons/Button7.add_theme_stylebox_override("disabled", style)
        elif i == 8:
                $Buttons/Button8.add_theme_stylebox_override("normal", style)
                $Buttons/Button8.add_theme_stylebox_override("hover", style)
                $Buttons/Button8.add_theme_stylebox_override("pressed", style)
                $Buttons/Button8.add_theme_stylebox_override("disabled", style)
        elif i == 9:
                $Buttons/Button9.add_theme_stylebox_override("normal", style)
                $Buttons/Button9.add_theme_stylebox_override("hover", style)
                $Buttons/Button9.add_theme_stylebox_override("pressed", style)
                $Buttons/Button9.add_theme_stylebox_override("disabled", style)

        $InBetweenTimer.one_shot = true
        $InBetweenTimer.start()


func _on_start_button_pressed():
        $Buttons/StartButton.queue_free()
        $Buttons/BackButton2.disabled = true
        $SubTitle.text = "Level: " + str(level)
```

```
	play()


func _on_light_up_timer_timeout():
	done_light_up = true


func new_level():
	level += 1
	$Buttons/Button1.disabled = true
	$Buttons/Button2.disabled = true
	$Buttons/Button3.disabled = true
	$Buttons/Button4.disabled = true
	$Buttons/Button5.disabled = true
	$Buttons/Button6.disabled = true
	$Buttons/Button7.disabled = true
	$Buttons/Button8.disabled = true
	$Buttons/Button9.disabled = true
	for index in pressed.size():
		sequence.append(pressed[index])
	sequence.append(randi_range(1, 9))
	pressed.clear()
	play()


func play():
	start = true
	light_up = true
	for index in sequence.size():
		light_up_button(sequence[index])
		await $LightUpTimer.timeout
		turn_off_button(sequence[index])
		await $InBetweenTimer.timeout
	$Buttons/Button1.disabled = false
	$Buttons/Button2.disabled = false
	$Buttons/Button3.disabled = false
	$Buttons/Button4.disabled = false
	$Buttons/Button5.disabled = false
	$Buttons/Button6.disabled = false
	$Buttons/Button7.disabled = false
```

```
        $Buttons/Button8.disabled = false
        $Buttons/Button9.disabled = false
        user_turn = true


func end(le):
        $Buttons/BackButton2.disabled = false
        $SubTitle.text = "Congradulations! You survived until level " + str(le)
        $LivesTitle.text = "Go back and come back in to try again!"
        Global.sequence_memory_test_score(le)
        Global.save()
```

Language Memory:

```
extends Node


# Called when the node enters the scene tree for the first time.
func _ready():
        pass # Replace with function body.


# Called every frame. 'delta' is the elapsed time since the previous frame.
func _process(delta):
        pass


func _on_back_button_pressed():
        get_tree().change_scene_to_file("res://scenes/exercises.tscn")


func _on_start_button_pressed():
        get_tree().change_scene_to_file("res://scenes/verbalmemorytest.tscn")
```

Language Memory Test:

```
extends Node

var lives: int = 3
var score: int = 0
```

var new: Array =
["aback","abaft","abandoned","abashed","aberrant","abhorrent","abiding","abject","ablaze","able
","abnormal","aboard","aboriginal","abortive","abounding","abrasive","abrupt","absent","absorb
ed","absorbing","abstracted","absurd","abundant","abusive","accept","acceptable","accessible","
accidental","account","accurate","achiever","acid","acidic","acoustic","acoustics","acrid","act","
action","activity","actor","actually","ad
hoc","adamant","adaptable","add","addicted","addition","adhesive","adjoining","adjustment","ad
mire","admit","adorable","adventurous","advertisement","advice","advise","afford","afraid","aft
ermath","afternoon","afterthought","aggressive","agonizing","agree","agreeable","agreement","a
head","air","airplane","airport","ajar","alarm","alcoholic","alert","alike","alive","alleged","allow
","alluring","aloof","amazing","ambiguous","ambitious","amount","amuck","amuse","amused","
amusement","amusing","analyze","ancient","anger","angle","angry","animal","animated","annou
nce","annoy","annoyed","annoying","answer","ants","anxious","apathetic","apologise","apparatu
s","apparel","appear","applaud","appliance","appreciate","approval","approve","aquatic","arch",
"argue","argument","arithmetic","arm","army","aromatic","arrange","arrest","arrive","arrogant",
"art","ashamed","ask","aspiring","assorted","astonishing","attach","attack","attempt","attend","at
tract","attraction","attractive","aunt","auspicious","authority","automatic","available","average",
"avoid","awake","aware","awesome","awful","axiomatic","babies","baby","back","bad","badge"
,"bag","bait","bake","balance","ball","ban","bang","barbarous","bare","base","baseball","bashful
","basin","basket","basketball","bat","bath","bathe","battle","bawdy","bead","beam","bear","bea
utiful","bed","bedroom","beds","bee","beef","befitting","beg","beginner","behave","behavior","b
elief","believe","bell","belligerent","bells","belong","beneficial","bent","berry","berserk","best",
"better","bewildered","big","bike","bikes","billowy","bird","birds","birth","birthday","bit","bite"
,"bite-sized","bitter","bizarre","black","black-and-white","blade","bleach","bless","blind","blink"
,"blood","bloody","blot","blow","blue","blue-eyed","blush","blushing","board","boast","boat","b
oil","boiling","bolt","bomb","bone","book","books","boorish","boot","border","bore","bored","b
oring","borrow","bottle","bounce","bouncy","boundary","boundless","bow","box","boy","brainy
","brake","branch","brash","brass","brave","brawny","breakable","breath","breathe","breezy","br
ick","bridge","brief","bright","broad","broken","brother","brown","bruise","brush","bubble","bu
cket","building","bulb","bump","bumpy","burly","burn","burst","bury","bushes","business","bus
tling","busy","butter","button","buzz","cabbage","cable","cactus","cagey","cake","cakes","calcul
ate","calculating","calculator","calendar","call","callous","calm","camera","camp","can","canno
n","canvas","cap","capable","capricious","caption","car","card","care","careful","careless","carin
g","carpenter","carriage","carry","cars","cart","carve","cast","cat","cats","cattle","cause","cautio
us","cave","ceaseless","celery","cellar","cemetery","cent","certain","chalk","challenge","chance"
,"change","changeable","channel","charge","charming","chase","cheap","cheat","check","cheer",
"cheerful","cheese","chemical","cherries","cherry","chess","chew","chicken","chickens","chief",
"childlike","children","chilly","chin","chivalrous","choke","chop","chubby","chunky","church","
circle","claim","clam","clammy","clap","class","classy","clean","clear","clever","clip","cloistere
d","close","closed","cloth","cloudy","clover","club","clumsy","cluttered","coach","coal","coast",

"coat","cobweb","coherent","coil","cold","collar","collect","color","colorful","colossal","colour"
,"comb","combative","comfortable","command","committee","common","communicate","comp
any","compare","comparison","compete","competition","complain","complete","complex","conc
entrate","concern","concerned","condemned","condition","confess","confuse","confused","conn
ect","connection","conscious","consider","consist","contain","continue","control","cooing","coo
k","cool","cooperative","coordinated","copper","copy","corn","correct","cough","count","countr
y","courageous","cover","cow","cowardly","cows","crabby","crack","cracker","crash","crate","c
raven","crawl","crayon","crazy","cream","creator","creature","credit","creepy","crib","crime","c
rook","crooked","cross","crow","crowd","crowded","crown","cruel","crush","cry","cub","cuddly
","cultured","cumbersome","cup","cure","curious","curl","curly","current","curtain","curve","cur
ved","curvy","cushion","cut","cute","cycle","cynical","dad","daffy","daily","dam","damage","da
maged","damaging","damp","dance","dangerous","dapper","dare","dark","dashing","daughter","
day","dazzling","dead","deadpan","deafening","dear","death","debonair","debt","decay","deceiv
e","decide","decision","decisive","decorate","decorous","deep","deeply","deer","defeated","defe
ctive","defiant","degree","delay","delicate","delicious","delight","delightful","delirious","deliver
","demonic","depend","dependent","depressed","deranged","describe","descriptive","desert","de
serted","deserve","design","desire","desk","destroy","destruction","detail","detailed","detect","d
etermined","develop","development","devilish","didactic","different","difficult","digestion","dili
gent","dime","dinner","dinosaurs","direction","direful","dirt","dirty","disagree","disagreeable","
disappear","disapprove","disarm","disastrous","discover","discovery","discreet","discussion","di
sgusted","disgusting","disillusioned","dislike","dispensable","distance","distinct","distribution","
disturbed","divergent","divide","division","dizzy","dock","doctor","dog","dogs","doll","dolls","d
omineering","donkey","door","double","doubt","doubtful","downtown","drab","draconian","dra
g","drain","dramatic","drawer","dream","dreary","dress","drink","drip","driving","drop","drown
","drum","drunk","dry","duck","ducks","dull","dust","dusty","dynamic","dysfunctional","eager",
"ear","early","earn","earsplitting","earth","earthquake","earthy","easy","eatable","economic","ed
ge","educate","educated","education","effect","efficacious","efficient","egg","eggnog","eggs","ei
ght","elastic","elated","elbow","elderly","electric","elegant","elfin","elite","embarrass","embarra
ssed","eminent","employ","empty","enchanted","enchanting","encourage","encouraging","end","
endurable","energetic","engine","enjoy","enormous","enter","entertain","entertaining","enthusias
tic","envious","equable","equal","erect","erratic","error","escape","ethereal","evanescent","evasi
ve","even","event","examine","example","excellent","exchange","excite","excited","exciting","e
xclusive","excuse","exercise","exist","existence","exotic","expand","expansion","expect","expen
sive","experience","expert","explain","explode","extend","extra-large","extra-small","exuberant"
,"exultant","eye","eyes","fabulous","face","fact","fade","faded","fail","faint","fair","fairies","fait
hful","fall","fallacious","false","familiar","famous","fanatical","fancy","fang","fantastic","far","f
ar-flung","farm","fascinated","fast","fasten","fat","faulty","fax","fear","fearful","fearless","feebl
e","feeling","feigned","female","fence","fertile","festive","fetch","few","field","fierce","file","fil
l","film","filthy","fine","finger","finicky","fire","fireman","first","fish","fit","five","fix","fixed",
"flag","flagrant","flaky","flame","flap","flash","flashy","flat","flavor","flawless","flesh","flight",

"flimsy","flippant","float","flock","flood","floor","flow","flower","flowers","flowery","fluffy","f
luttering","fly","foamy","fog","fold","follow","food","fool","foolish","foot","force","foregoing",
"forgetful","fork","form","fortunate","found","four","fowl","fragile","frail","frame","frantic","fre
e","freezing","frequent","fresh","fretful","friction","friend","friendly","friends","frighten","fright
ened","frightening","frog","frogs","front","fruit","fry","fuel","full","fumbling","functional","fun
ny","furniture","furry","furtive","future","futuristic","fuzzy","gabby","gainful","gamy","gaping",
"garrulous","gate","gather","gaudy","gaze","geese","general","gentle","ghost","giant","giants","
giddy","gifted","gigantic","giraffe","girl","girls","glamorous","glass","gleaming","glib","glisteni
ng","glorious","glossy","glove","glow","glue","godly","gold","good","goofy","gorgeous","gover
nment","governor","grab","graceful","grade","grain","grandfather","grandiose","grandmother","
grape","grass","grate","grateful","gratis","gray","grease","greasy","great","greedy","green","gree
t","grey","grieving","grin","grip","groan","groovy","grotesque","grouchy","ground","group","gr
owth","grubby","gruesome","grumpy","guarantee","guard","guarded","guess","guide","guiltless"
,"guitar","gullible","gun","gusty","guttural","habitual","hair","haircut","half","hall","hallowed","
halting","hammer","hand","handle","hands","handsome","handsomely","handy","hang","hangin
g","hapless","happen","happy","harass","harbor","hard","hard-to-find","harm","harmonious","ha
rmony","harsh","hat","hate","hateful","haunt","head","heady","heal","health","healthy","heap","
heartbreaking","heat","heavenly","heavy","hellish","help","helpful","helpless","hesitant","hideo
us","high","high-pitched","highfalutin","hilarious","hill","hissing","historical","history","hobbies
","hole","holiday","holistic","hollow","home","homeless","homely","honey","honorable","hook"
,"hop","hope","horn","horrible","horse","horses","hose","hospitable","hospital","hot","hour","ho
use","houses","hover","hug","huge","hulking","hum","humdrum","humor","humorous","hungry
","hunt","hurried","hurry","hurt","hushed","husky","hydrant","hypnotic","hysterical","ice","icicl
e","icky","icy","idea","identify","idiotic","ignorant","ignore","ill","ill-fated","ill-informed","illeg
al","illustrious","imaginary","imagine","immense","imminent","impartial","imperfect","impolite
","important","imported","impossible","impress","improve","impulse","incandescent","include",
"income","incompetent","inconclusive","increase","incredible","industrious","industry","inexpen
sive","infamous","influence","inform","inject","injure","ink","innate","innocent","inquisitive","i
nsect","insidious","instinctive","instruct","instrument","insurance","intelligent","intend","interest
","interesting","interfere","internal","interrupt","introduce","invent","invention","invincible","in
vite","irate","iron","irritate","irritating","island","itch","itchy","jaded","jagged","jail","jam","jar"
,"jazzy","jealous","jeans","jelly","jellyfish","jewel","jittery","jobless","jog","join","joke","jolly",
"joyous","judge","judicious","juggle","juice","juicy","jumbled","jump","jumpy","juvenile","kap
ut","keen","kettle","key","kick","kill","kind","kindhearted","kindly","kiss","kittens","kitty","kne
e","kneel","knife","knit","knock","knot","knotty","knowing","knowledge","knowledgeable","kn
own","label","labored","laborer","lace","lackadaisical","lacking","ladybug","lake","lame","lame
ntable","lamp","land","language","languid","large","last","late","laugh","laughable","launch","la
vish","lazy","lean","learn","learned","leather","left","leg","legal","legs","lethal","letter","letters",
"lettuce","level","lewd","library","license","lick","lie","light","lighten","like","likeable","limit","
limping","line","linen","lip","liquid","list","listen","literate","little","live","lively","living","load

","loaf","lock","locket","lonely","long","long-term","longing","look","loose","lopsided","loss","loud","loutish","love","lovely","loving","low","lowly","lucky","ludicrous","lumber","lumpy","lunch","lunchroom","lush","luxuriant","lying","lyrical","macabre","machine","macho","maddening","madly","magenta","magic","magical","magnificent","maid","mailbox","majestic","makeshift","male","malicious","mammoth","man","manage","maniacal","many","marble","march","mark","marked","market","married","marry","marvelous","mask","mass","massive","match","mate","material","materialistic","matter","mature","meal","mean","measly","measure","meat","meaty","meddle","medical","meek","meeting","mellow","melodic","melt","melted","memorize","memory","men","mend","merciful","mere","mess
up","messy","metal","mice","middle","mighty","military","milk","milky","mind","mindless","mine","miniature","minister","minor","mint","minute","miscreant","miss","mist","misty","mitten","mix","mixed","moan","moaning","modern","moldy","mom","momentous","money","monkey","month","moon","moor","morning","mother","motion","motionless","mountain","mountainous","mourn","mouth","move","muddle","muddled","mug","multiply","mundane","murder","murky","muscle","mushy","mute","mysterious","nail","naive","name","nappy","narrow","nasty","nation","natural","naughty","nauseating","near","neat","nebulous","necessary","neck","need","needle","needless","needy","neighborly","nerve","nervous","nest","new","next","nice","nifty","night","nimble","nine","nippy","nod","noise","noiseless","noisy","nonchalant","nondescript","nonstop","normal","north","nose","nostalgic","nosy","note","notebook","notice","noxious","null","number","numberless","numerous","nut","nutritious","nutty","oafish","oatmeal","obedient","obeisant","obese","obey","object","obnoxious","obscene","obsequious","observant","observation","observe","obsolete","obtain","obtainable","occur","ocean","oceanic","odd","offbeat","offend","offer","office","oil","old","old-fashioned","omniscient","one","onerous","open","opposite","optimal","orange","oranges","order","ordinary","organic","ossified","outgoing","outrageous","outstanding","oval","oven","overconfident","overflow","overjoyed","overrated","overt","overwrought","owe","own","pack","paddle","page","pail","painful","painstaking","paint","pale","paltry","pan","pancake","panicky","panoramic","paper","parallel","parcel","parched","park","parsimonious","part","partner","party","pass","passenger","past","paste","pastoral","pat","pathetic","pause","payment","peace","peaceful","pear","peck","pedal","peel","peep","pen","pencil","penitent","perfect","perform","periodic","permissible","permit","perpetual","person","pest","pet","petite","pets","phobic","phone","physical","picayune","pick","pickle","picture","pie","pies","pig","pigs","pin","pinch","pine","pink","pipe","piquant","pizzas","place","placid","plain","plan","plane","planes","plant","plantation","plants","plastic","plate","plausible","play","playground","pleasant","please","pleasure","plot","plough","plucky","plug","pocket","point","pointless","poised","poison","poke","polish","polite","political","pollution","poor","pop","popcorn","porter","position","possess","possessive","possible","post","pot","potato","pour","powder","power","powerful","practice","pray","preach","precede","precious","prefer","premium","prepare","present","preserve","press","pretend","pretty","prevent","previous","price","pricey","prick","prickly","print","private","probable","produce","productive","profit","profuse","program","promise","property","prose","protect","protective","protest","proud","provide","psychedelic","psychotic","public","puffy","pull","pump","pumpe

d","punch","puncture","punish","punishment","puny","purple","purpose","purring","push","push
y","puzzled","puzzling","quack","quaint","quarrelsome","quarter","quartz","queen","question","
questionable","queue","quick","quickest","quicksand","quiet","quill","quilt","quince","quirky","
quiver","quixotic","quizzical","rabbit","rabbits","rabid","race","racial","radiate","ragged","rail","
railway","rain","rainstorm","rainy","raise","rake","rambunctious","rampant","range","rapid","rar
e","raspy","rat","rate","ratty","ray","reach","reaction","reading","ready","real","realize","reason"
,"rebel","receipt","receive","receptive","recess","recognise","recondite","record","red","reduce",
"redundant","reflect","reflective","refuse","regret","regular","reign","reject","rejoice","relation",
"relax","release","relieved","religion","rely","remain","remarkable","remember","remind","remi
niscent","remove","repair","repeat","replace","reply","report","representative","reproduce","repu
lsive","request","rescue","resolute","resonant","respect","responsible","rest","retire","return","re
ward","rhetorical","rhyme","rhythm","rice","rich","riddle","rifle","right","righteous","rightful","r
igid","ring","rings","rinse","ripe","risk","ritzy","river","road","roasted","rob","robin","robust","r
ock","rod","roll","romantic","roof","room","roomy","root","rose","rot","rotten","rough","round",
"route","royal","rub","ruddy","rude","ruin","rule","run","rural","rush","rustic","ruthless","sable",
"sack","sad","safe","sail","salt","salty","same","sand","sassy","satisfy","satisfying","save","savo
ry","saw","scale","scandalous","scarce","scare","scarecrow","scared","scarf","scary","scatter","s
cattered","scene","scent","school","science","scientific","scintillating","scissors","scold","scorch
","scrape","scratch","scrawny","scream","screeching","screw","scribble","scrub","sea","seal","se
arch","seashore","seat","second","second-hand","secret","secretary","secretive","sedate","seed","
seemly","selection","selective","self","selfish","sense","separate","serious","servant","serve","set
tle","shade","shaggy","shake","shaky","shallow","shame","shape","share","sharp","shave","shee
p","sheet","shelf","shelter","shiny","ship","shirt","shiver","shivering","shock","shocking","shoe"
,"shoes","shop","short","show","shrill","shrug","shut","shy","sick","side","sidewalk","sigh","sig
n","signal","silent","silk","silky","silly","silver","simple","simplistic","sin","sincere","sink","sip"
,"sister","sisters","six","size","skate","ski","skillful","skin","skinny","skip","skirt","sky","slap","
slave","sleep","sleepy","sleet","slim","slimy","slip","slippery","slope","sloppy","slow","small","
smart","smash","smell","smelly","smile","smiling","smoggy","smoke","smooth","snail","snails",
"snake","snakes","snatch","sneaky","sneeze","sniff","snobbish","snore","snotty","snow","soak",
"soap","society","sock","soda","sofa","soft","soggy","solid","somber","son","song","songs","soo
the","sophisticated","sordid","sore","sort","sound","soup","sour","space","spade","spare","spark
","sparkle","sparkling","special","spectacular","spell","spicy","spiders","spiffy","spiky","spill","
spiritual","spiteful","splendid","spoil","sponge","spooky","spoon","spot","spotless","spotted","sp
otty","spray","spring","sprout","spurious","spy","squalid","square","squash","squeak","squeal","
squealing","squeamish","squeeze","squirrel","stage","stain","staking","stale","stamp","standing",
"star","stare","start","statement","station","statuesque","stay","steadfast","steady","steam","steel
","steep","steer","stem","step","stereotyped","stew","stick","sticks","sticky","stiff","stimulating",
"stingy","stir","stitch","stocking","stomach","stone","stop","store","stormy","story","stove","stra
ight","strange","stranger","strap","straw","stream","street","strengthen","stretch","string","strip",
"striped","stroke","strong","structure","stuff","stupendous","stupid","sturdy","subdued","subsequ

ent","substance","substantial","subtract","succeed","successful","succinct","suck","sudden","suffer","sugar","suggest","suggestion","suit","sulky","summer","sun","super","superb","superficial","supply","support","suppose","supreme","surprise","surround","suspect","suspend","swanky","sweater","sweet","sweltering","swift","swim","swing","switch","symptomatic","synonymous","system","table","taboo","tacit","tacky","tail","talented","talk","tall","tame","tan","tangible","tangy","tank","tap","tart","taste","tasteful","tasteless","tasty","tawdry","tax","teaching","team","tearful","tease","tedious","teeny","teeny-tiny","teeth","telephone","telling","temper","temporary","tempt","ten","tendency","tender","tense","tent","tenuous","terrible","terrific","terrify","territory","test","tested","testy","texture","thank","thankful","thaw","theory","therapeutic","thick","thin","thing","things","thinkable","third","thirsty","thought","thoughtful","thoughtless","thread","threatening","three","thrill","throat","throne","thumb","thunder","thundering","tick","ticket","tickle","tidy","tie","tiger","tight","tightfisted","time","tin","tiny","tip","tire","tired","tiresome","title","toad","toe","toes","tomatoes","tongue","tooth","toothbrush","toothpaste","toothsome","top","torpid","touch","tough","tour","tow","towering","town","toy","toys","trace","trade","trail","train","trains","tramp","tranquil","transport","trap","trashy","travel","tray","treat","treatment","tree","trees","tremble","tremendous","trick","tricky","trip","trite","trot","trouble","troubled","trousers","truck","trucks","truculent","true","trust","truthful","try","tub","tug","tumble","turkey","turn","twig","twist","two","type","typical","ubiquitous","ugliest","ugly","ultra","umbrella","unable","unaccountable","unadvised","unarmed","unbecoming","unbiased","uncle","uncovered","understood","underwear","undesirable","undress","unequal","unequaled","uneven","unfasten","unhealthy","uninterested","unique","unit","unite","unkempt","unknown","unlock","unnatural","unpack","unruly","unsightly","unsuitable","untidy","unused","unusual","unwieldy","unwritten","upbeat","uppity","upset","uptight","use","used","useful","useless","utopian","utter","uttermost","vacation","vacuous","vagabond","vague","valuable","value","van","vanish","various","vase","vast","vegetable","veil","vein","vengeful","venomous","verdant","verse","versed","vessel","vest","victorious","view","vigorous","violent","violet","visit","visitor","vivacious","voice","voiceless","volatile","volcano","volleyball","voracious","voyage","vulgar","wacky","waggish","wail","wait","waiting","wakeful","walk","wall","wander","wandering","want","wanting","war","warlike","warm","warn","wary","wash","waste","wasteful","watch","water","watery","wave","waves","wax","way","weak","wealth","wealthy","weary","weather","week","weigh","weight","welcome","well-groomed","well-made","well-off","well-to-do","wet","wheel","whimsical","whine","whip","whirl","whisper","whispering","whistle","white","whole","wholesale","wicked","wide","wide-eyed","wiggly","wild","wilderness","willing","wind","window","windy","wine","wing","wink","winter","wipe","wire","wiry","wise","wish","wistful","witty","wobble","woebegone","woman","womanly","women","wonder","wonderful","wood","wooden","wool","woozy","word","work","workable","worm","worried","worry","worthless","wound","wrap","wrathful","wreck","wren","wrench","wrestle","wretched","wriggle","wrist","writer","writing","wrong","wry","x-ray","yak","yam","yard","yarn","yawn","year","yell","yellow","yielding","yoke","young","youthful","yummy","zany","zealous","zebra","zephyr","zesty","zinc","zip","zipper","zippy","zonked","zoo","zoom"]
var seen: Array = []

```
var start: bool = false
var word: String = ""
var num: int = randi_range(0, new.size())
var over: bool = false

# Called when the node enters the scene tree for the first time.
func _ready():
	print(str(new.size()))
	$SeenButton.disabled = true
	$NewButton.disabled = true


# Called every frame. 'delta' is the elapsed time since the previous frame.
func _process(_delta):
	if over == false:
		$LivesLabel.text = "Lives: " + str(lives)
		$ScoreLabel.text = "Score: " + str(score)
	else:
		$WordLabel.text = "Final Score: " + str(score)


func _on_back_button_pressed():
	get_tree().change_scene_to_file("res://scenes/verbalmemory.tscn")

# If user is right / wrong
func _on_seen_button_pressed():
	if word == "seen":
		score += 1
	elif word == "new":
		lives -= 1
		if lives < 1:
			end()

	new_word() # Either way it gets a new word

func _on_new_button_pressed():
	if word == "new":
		score += 1
		seen.append(new[num])
		new.erase(new[num])
```

```
        elif word == "seen":
                lives -= 1
                if lives < 1:
                        end()

        new_word()


func _on_start_button_pressed():
        $SeenButton.disabled = false
        $NewButton.disabled = false
        $BackButton.disabled = true
        $StartButton.queue_free()
        $WordLabel.text = new[num]
        word = "new"


func new_word():
        var choice = randi_range(1, 2)
        if choice == 1:
                word = "new"
                num = randi_range(0, new.size() - 1)
                $WordLabel.text = new[num]
        elif choice == 2:
                word = "seen"
                num = randi_range(0, seen.size() - 1)
                $WordLabel.text = seen[num]


func end():
        over = true

        $SeenButton.queue_free()
        $NewButton.queue_free()
        $ScoreLabel.queue_free()
        $LivesLabel.queue_free()

        $BackButton.disabled = false
        $WordLabel.text = "Final Score: " + str(score)
        Global.verbal_memory_test_score(score)
```

```
Global.save()
```