

Appendix A

Extra Information

A.1 Tables & Test Cases

This section contains tables and test cases mentioned in the Evaluation section of this report. They are included here as they remain peripheral to the main body of this report, and would break up the theme and flow of the text if it appeared in the body, as would the source code listings featured in chapter C and the detailed user guide featured in chapter B.

	Noise type				
Cellular Distance Function	Simplex Smooth	Simplex	Perlin	Value	Value Cubic
Euclidean	81ms	83ms	80ms	82ms	92ms
	84ms	88ms	74ms	84ms	105ms
	83ms	83ms	74ms	83ms	74ms
	AVG: 83ms	AVG: 85ms	AVG: 76ms	AVG: 83ms	AVG: 90ms
Euclidean Squared	81ms	74ms	81ms	90ms	76ms
	77ms	91ms	73ms	119ms	109ms
	84ms	79ms	80ms	93ms	82ms
	AVG: 81ms	AVG: 81ms	AVG: 78ms	AVG: 101ms	AVG: 89ms
Manhattan	83ms	93ms	82ms	80ms	81ms
	107ms	101ms	72ms	81ms	72ms
	82ms	87ms	80ms	94ms	101ms
	AVG: 91ms	AVG: 94ms	AVG: 78ms	AVG: 85ms	AVG: 85ms
Hybrid (Euclidean & Manhattan)	77ms	87ms	85ms	85ms	76ms
	96ms	122ms	74ms	85ms	77ms
	82ms	87ms	83ms	87ms	77ms
	AVG: 85ms	AVG: 99ms	AVG: 81ms	AVG: 86ms	AVG: 77ms

Figure A.1: A table denoting some performance tests done with comparing the noise algorithms, the cellular distance functions and the combination pairs between them. This was done on the Noise implementation of the game; the “tests” were simply checking how long it took to create levels on the author of this report’s computer, and all the other script variables were assigned to their default values as described in chapter 4. Each noise type and cellular distance function pair was run 3 times, with the mean time (including potential outliers) calculated afterwards *to the nearest integer*. Be advised that the author of this report did these tests on his computer, so on different computers, results can, and likely *will*, vary.

	Noise type				
Fractal Type	Simplex Smooth	Simplex	Perlin	Value	Value Cubic
None	92ms	89ms	84ms	101ms	77ms
	86ms	152ms	98ms	86ms	88ms
	123ms	99ms	95ms	97ms	86ms
	AVG: 100ms	AVG: 113ms	AVG: 92ms	AVG: 95ms	AVG: 84ms
FBM (Fractional Brownian Motion)	77ms	81ms	73ms	78ms	68ms
	93ms	87ms	79ms	137ms	64ms
	87ms	93ms	73ms	82ms	87ms
	AVG: 86ms	AVG: 87ms	AVG: 75ms	AVG: 99ms	AVG: 73ms
Ridged	23ms	74ms	15ms	27ms	14ms
	25ms	69ms	16ms	28ms	9ms
	23ms	70ms	16ms	26ms	11ms
	AVG: 24ms	AVG: 71ms	AVG: 16ms	AVG: 27ms	AVG: 11ms
Ping Pong	59ms	67ms	108ms	128ms	163ms
	54ms	77ms	105ms	71ms	172ms
	58ms	64ms	111ms	72ms	164ms
	AVG: 57ms	AVG: 69ms	AVG: 108ms	AVG: 90ms	AVG: 166ms

Figure A.2: A table denoting some performance tests done with comparing the noise algorithms, the fractal types and the combination pairs between them. This was done on the Noise implementation of the game; the “tests” were simply checking how long it took to create levels on the author of this report’s computer, and all the other script variables were assigned to their default values as described in chapter 4. Each noise type and fractal type pair was run 3 times, with the mean time (including potential outliers) calculated afterwards *to the nearest integer*. Be advised that the author of this report did these tests on his computer, so on different computers, results can, and likely *will*, vary.

use_custom_axiom = false axiom = "OWB"	use_custom_axiom = true upper_limit = 3	use_custom_axiom = true upper_limit = 10	use_custom_axiom = true upper_limit = 25
21ms	25ms (length = 2)	20ms (length = 4)	9ms (length = 25)
17ms	13ms (length = 2)	11ms (length = 9)	14ms (length = 2)
21ms	21ms (length = 1)	21ms (length = 8)	21ms (length = 21)
20ms	16ms (length = 1)	18ms (length = 5)	20ms (length = 24)
20ms	11ms (length = 2)	11ms (length = 4)	15ms (length = 14)
AVG: 20ms	AVG: 17ms	AVG: 16ms	AVG: 16ms

Figure A.3: A table denoting some performance tests done with comparing the lengths of axioms used in L-Systems. Obviously, this was done on the L-System implementation of the game; the “tests” were simply checking how long it took to create levels on the author of this report’s computer, as well as how long the randomly generated axioms were (where appropriate), and all the other script variables were assigned to their default values. Each of the shown settings were run 5 times, with the mean time (including potential outliers) calculated afterwards *to the nearest integer*. Be advised that the author of this report did these tests on his computer, so on different computers, results can, and likely *will*, vary.

rejection_samples	3	8	13	18
time	170ms	337ms	444ms	503ms
	103ms	224ms	392ms	505ms
	111ms	242ms	388ms	670ms
	AVG: 128ms	AVG: 268ms	AVG: 408ms	AVG: 559ms

Figure A.4: A table denoting some performance tests done with comparing the number of rejection samples used for Poisson Disk Sampling. Obviously, this was done on the Poisson Disk Sampling/Distribution implementation of the game; the “tests” were simply checking how long it took to create levels on the author of this report’s computer, and all the other script variables were assigned to their default values. Each of the shown settings were run 3 times, with the mean time (including potential outliers) calculated afterwards *to the nearest integer*. The bottom cell of the rightmost column, with tests done with 18 rejection samples, is highlighted red because, while testing with 18 rejection samples, at one time the program hung without returning any cell points within 10 seconds. The test had to be retaken another time. Be advised that the author of this report did these tests on his computer, so on different computers, results can, and likely *will*, vary.

	Random Starting Points			
Distance type	15	20	30	40
Euclidean distance	393ms	496ms	775ms	968ms
	385ms	504ms	744ms	970ms
	362ms	497ms	723ms	967ms
	AVG: 380ms	AVG: 499ms	AVG: 747ms	AVG: 968ms
Manhattan Distance	364ms	441ms	645ms	843ms
	346ms	470ms	630ms	835ms
	346ms	437ms	650ms	908ms
	AVG: 352ms	AVG: 449ms	AVG: 642ms	AVG: 862ms

Figure A.5: A table denoting some performance tests done with comparing the distance calculation algorithms, the number of random starting points and the combination pairs between them. This was done on the Voronoi Cells implementation of the game, and thus the number of random starting points corresponds directly to the number of unique Voronoi cells generated for each level. The “tests” were simply checking how long it took to create levels on the author of this report’s computer. Each of the shown settings were run 3 times, with the mean time (including potential outliers) calculated afterwards *to the nearest integer*. Be advised that the author of this report did these tests on his computer, so on different computers, results can, and likely *will*, vary.

L-System	Perlin/Simplex Noise	Poisson Disk Sampling/Distribution	Voronoi Cells
24ms	78ms	176ms	502ms
10ms	78ms	190ms	425ms
14ms	82ms	210ms	455ms
14ms	91ms	216ms	449ms
17ms	75ms	193ms	442ms
AVG: 16ms	AVG: 81ms	AVG: 197ms	AVG: 455ms

Figure A.6: A table denoting some performance tests done with comparing all of the algorithms implemented with the chosen scenario. This was done on every single implementation implementation of the game; the “tests” were simply checking how long it took to create levels on the author of this report’s computer, and every implementation was tested with its default variable values. Each of the implementations were run 5 times, with the mean time (including potential outliers) calculated afterwards *to the nearest integer*. Be advised that the author of this report did these tests on his computer, so on different computers, results can, and likely *will*, vary.

Appendix B

User Guide

B.1 Opening Godot

You will first need to download Godot 4 to run all the provided artefacts.

To run the projects in the .zip file, extract the projects into one folder. Then open Godot 4 (all projects in the source code listings folder are Godot 4 projects, **not** Godot 3 projects). When you start Godot for the first time, the project manager should be completely empty, without any projects, as described in Figure B.1. Projects have to be imported either one-by-one (by clicking “Import” and going to the relevant project and opening it) or by clicking “Scan”, then going to a folder of Godot projects and selecting it. The projects can then be opened in the project manager and edited as needed in Godot. Click “Scan”, then go to the folder where you extracted the projects, then click the “Select Current Folder” button, as shown in Figure B.2, and all the projects should show up in the editor, as shown in Figure B.3. You can then double click on any one project (or click on it once and click the “Edit” button) to open it in the Godot editor, an example of which is shown in Figure B.4. Alternatively, to run the project itself without opening the editor, using the currently saved values for exported script variables where appropriate, click on the project *once* and click the “Run” button.

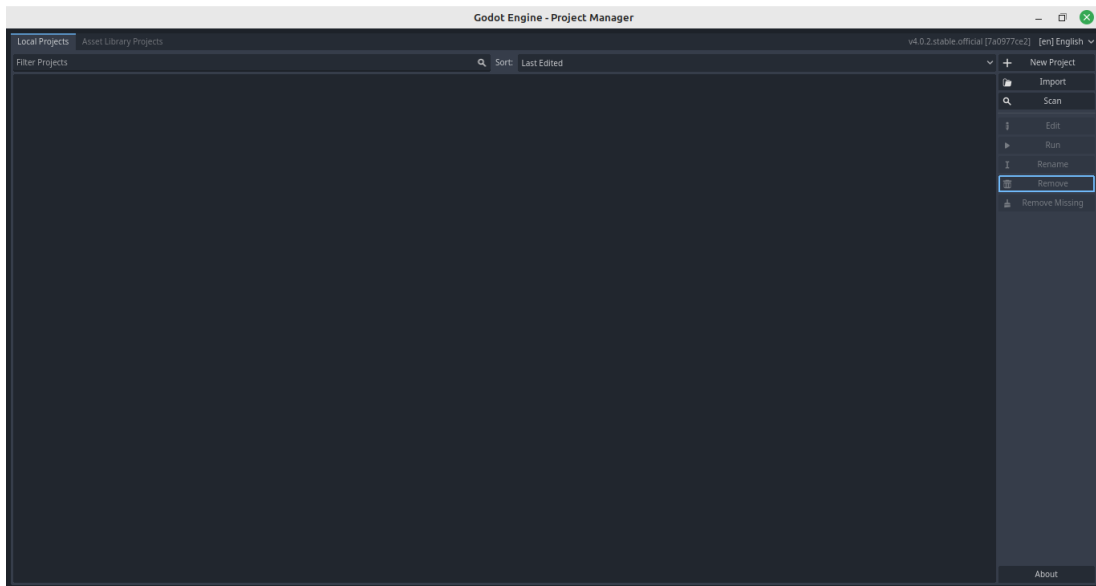


Figure B.1: The Godot editor, when it is opened for the first time, does not show any projects in the editor (the Steam version bundles several example projects). Projects need to be imported either one-by-one or by scanning a folder of Godot projects.

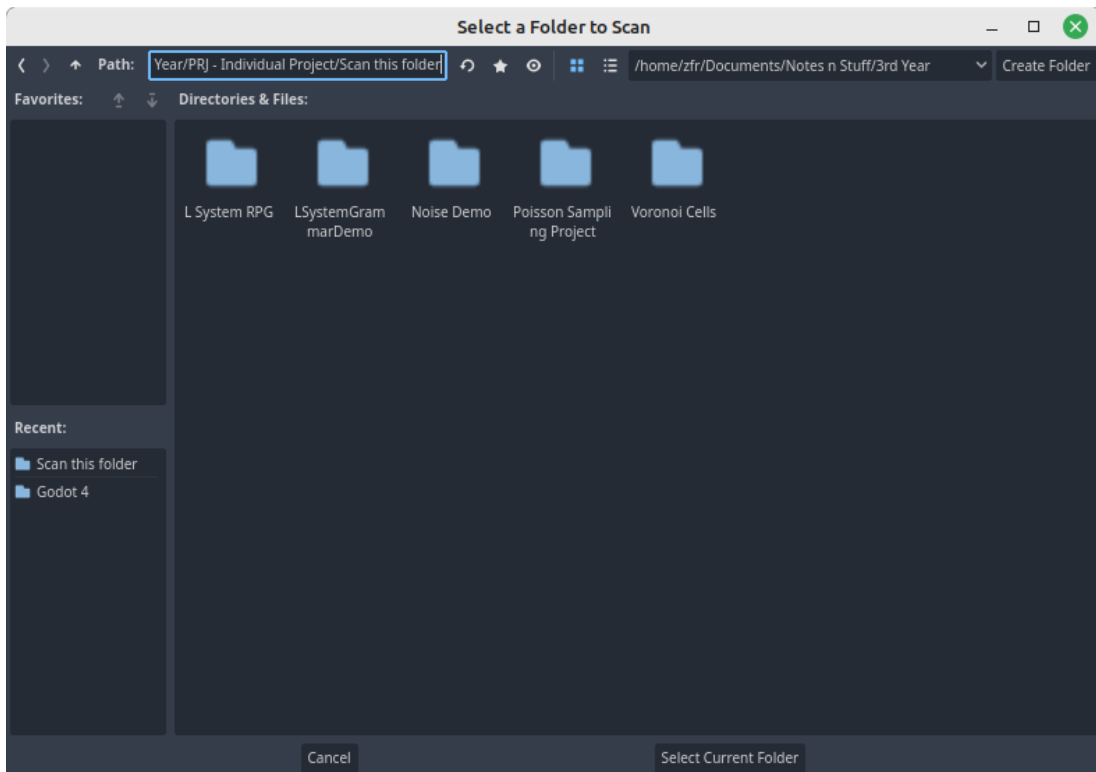


Figure B.2: You can click the "Scan" button in the project manager (in Figure B.1), then go to the relevant folder where your project are in Godot's built-in file explorer. Here, the artefacts behind this project have been exported into a separate folder called "Scan this folder" as an example.

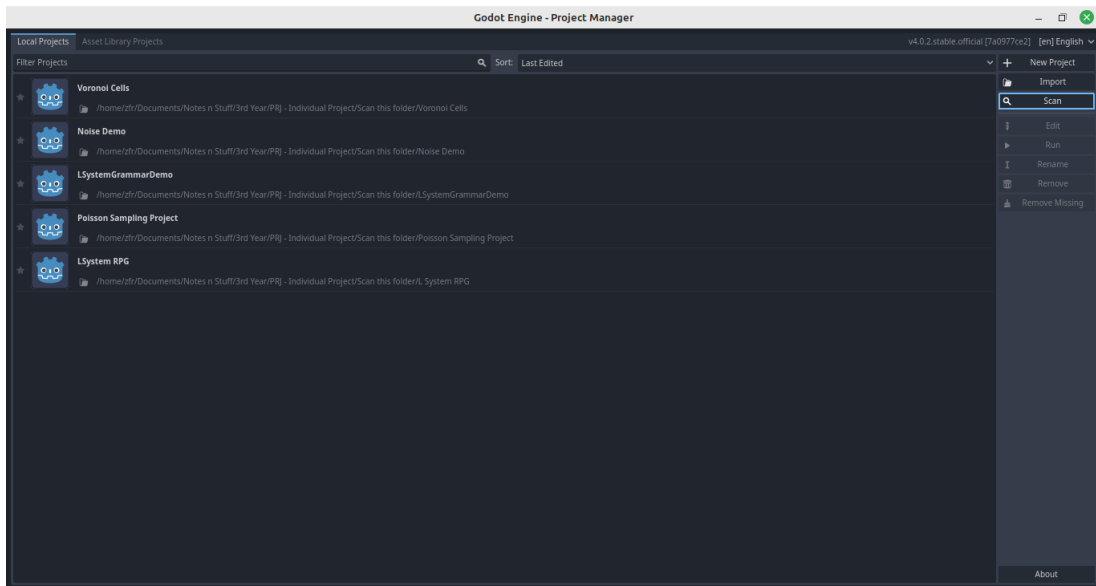


Figure B.3: Once some Godot projects have been imported into the project manager, you should be able to easily view the list and double-click on any one of the projects to edit them, which will open the editor after closing the project manager. You could also click the “Edit” button, or click “Run” to run the game without having to open the editor itself.

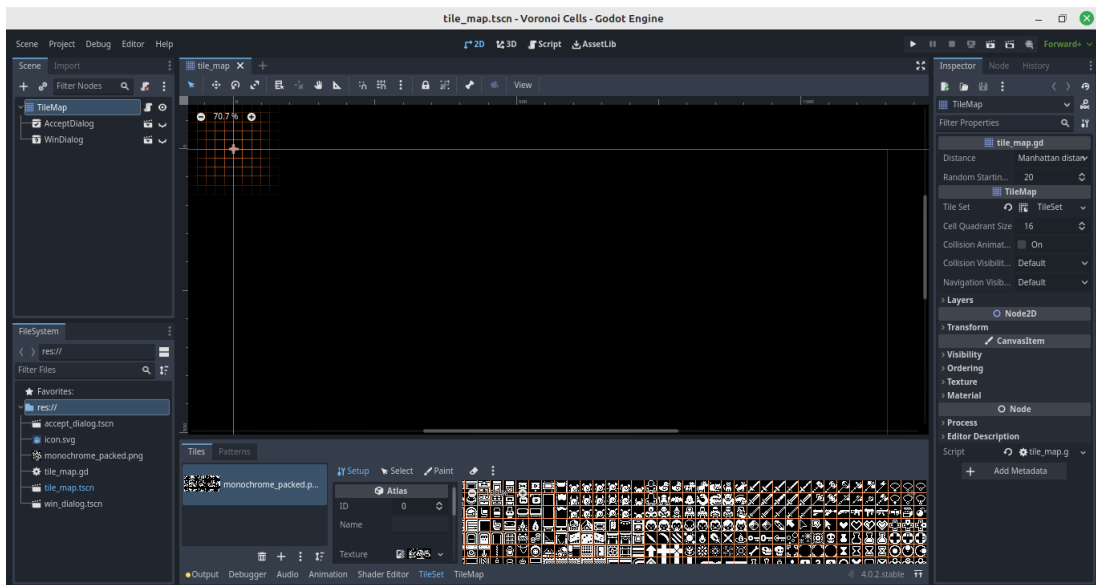


Figure B.4: The Godot editor open with the Voronoi cells project as an example. A visual description of the editor’s contents is in chapter B.2.

B.2 The Godot Editor

As you open up the Godot editor, you will see the main scene view in the center, as shown in Figure B.4, using the Voronoi cells implementation as an example. The left-hand side shows

the scene tree at the top, and the file system (from the root folder of the project) at the bottom. Meanwhile, the right hand side shows the currently selected node's export variables, *including the custom export variables* defined in the node's script file, and two other tabs, "Node" (which shows a list of signals for the scene that could be called in a script) and "History" (which shows the sequence of recent actions performed on the scene during the current session). Above this is also a set of buttons which can be used for playing the project and/or the current scene. We go over how to run the current project in chapter B.4.

B.3 Custom Export Variables

When you click on some of the scenes in the projects, there may be some "exported" variables from scripts that are visible to you in the editor (examples include the "Distance" and "Random Starting Points" variables in the Voronoi Cells project). You can hover over the variable names in the editor and it will show a brief description of what the variable correlates to in the script. We will now go over the different export variables across all of the provided artefacts in this section.

B.3.1 Lindenmayer System

All of the custom export variables defined in scene scripts for your use are in the child node "LSystem" (it is saved into its own scene file, but it is chiefly a child node of the root node "TileMap"). Open the "TileMap" scene (if it is not already opened for you when you launch the Godot editor) and click on the "LSystem" node in the scene tree to edit it.

1. axiom

- **Type:** String
- **Documentation:** The starting string from which the grammar starts applying its rules. Here it may be self defined, or randomly defined when "use_random_axiom" is true.
- **Default value:** "OWB"

2. use_random_axiom

- **Type:** Boolean (bool)
- **Documentation:** Uses a random axiom with the currently set grammar, computed upon runtime, with a length up to (but not strictly) the value of upper_limit. For

example, if `upper_limit` is set to 15, the generated axiom can be 15 characters, or it can be just 5 characters.

- **Default value:** `true`

3. `upper_limit`

- **Type:** Integer (`int`)
- **Documentation:** Defines how many characters a random axiom can have MAXIMUM. Only used when “`use_random_axiom`” is `true`.
- **Default value:** 10

4. `use_custom_ruleset`

- **Type:** Boolean (`bool`)
- **Documentation:** Allows the use of a customly defined ruleset made through amending the rules array in the editor.
- **Default value:** `false`

5. `ruleset`

- **Type:** String enumeration of the following choices:
 - (a) “Default”
 - (b) “More Buildings (IMPOSSIBLE)”
 - (c) “More Trees”
 - (d) “More Space”
- **Documentation:** Denotes a series of pre-defined rulesets for this L-System grammar, of alphabet O (blank space), W (trees and fauna) and B (buildings), that can be chosen and then used on runtime. Can choose between a default ruleset, a ruleset that produces more buildings, a ruleset that produces more trees and a ruleset that produces more empty space.
- **Default value:** “Default”

6. `rules`

- **Type:** Array of dictionaries

- **Documentation:** The set of rules that the L-System grammar uses. Shows the “default” ruleset in the Godot editor for the user to see. If “use_custom_ruleset” is set to true, this array can be edited with a custom defined ruleset that will be used on runtime, so long as it adheres to the alphabet of O (blank space), W (trees and fauna) and B (buildings).
- **Additional information:** The “_get_ruleset” function uses the String value in “ruleset” to set the value for “rules” on runtime, *if* “use_custom_ruleset” is false (which it *is* by default).
- **Default value:** The “Default” grammar, as shown in Figure B.5.

```

1      [
2      {
3          "from": "O",
4          "to": "OWO "
5      },
6      {
7          "from": "W",
8          "to": "WB "
9      },
10     {
11         "from": "B",
12         "to": "BW0 "
13     }
14 ]

```

Figure B.5: The “Default” grammar used for the “rules” export variable, stored in the constant “DEFAULT” in `l_system.gd`. See `l_system.gd` for the other grammars represented as arrays of dictionaries.

B.3.2 Perlin/Simplex Noise

1. noise_type

- **Type:** String enumeration of the following choices:

- (a) “Perlin”
- (b) “Simplex”
- (c) “Simplex Smooth”
- (d) “Value”
- (e) “Value Cubic”

- **Documentation:** Defines the type of noise generation algorithm to use. Equates to the “noise_type” property in FastNoiseLite.
- **Default value:** “Value Cubic”

2. fractal_type

- **Type:** Enumeration of the following choices:
 - (a) “Fractal None”
 - (b) “Fractal FBM”
 - (c) “Fractal Ridged”
 - (d) “Fractal Ping Pong”
- **Documentation:** Defines the type of method used to combine octaves of a noise image into a fractal. Directly equates to the FractalType enumeration in FastNoiseLite.
- **Default value:** “Fractal None”

3. cellular_distance_type

- **Type:** Enumeration of the following choices:
 - (a) “Distance Euclidean”
 - (b) “Distance Euclidean Squared”
 - (c) “Distance Manhattan”
 - (d) “Distance Hybrid”
- **Documentation:** Defines the function used to calculate the distance between the nearest/second-nearest point(s). Directly equates to the CellularDistanceFunction enumeration in FastNoiseLite.
- **Default value:** “Distance Euclidean”

4. noise_frequency

- **Type:** Floating point number (float) between 0.0 and 1.0 inclusive

- **Documentation:** Defines the frequency of the generated noise, the higher the frequency, the rougher and more granular the noise.
- **Additional information:** The default value for “frequency” in the “FastNoise-Class” is 0.01, resulting in very smooth and distinct noise.
- **Default value:** 0.894

5. tree_cap

- **Type:** Floating point number (float) between -1.0 and 1.0 inclusive
- **Documentation:** Defines the upper limit to set for painting a tree tile on a specific noise pixel. If the value returned by the “get_noise_2d” method (in FastNoiseLite) is smaller than this, then it gets painted.
- **Default value:** -0.048

6. building_cap

- **Type:** Floating point number (float) between -1.0 and 1.0 inclusive
- **Documentation:** Defines the upper limit to set for painting a building tile on a specific noise pixel. If the value returned by the “get_noise_2d” method (in FastNoiseLite) is smaller than this, then it gets painted. If the value of “building_cap” is smaller than “tree_cap,” then decide whether or not to paint a building cell there with “building_overtakes_tree.”
- **Default value:** -0.252

7. building_overtakes_tree

- **Type:** Floating point number (float) between 0.0 and 0.5 inclusive
- **Documentation:** Only used when “building_cap” is smaller than “tree_cap.” Determines the probability that a building tile would be painted in a cell where a tree tile was, or could be, also painted. Whether or not the cell actually is painted over is decided on computation time.
- **Default value:** 0.12

B.3.3 Poisson Disk Sampling

1. paint_building_probability

- **Type:** Floating point number (float) between 0.0 and 1.0 inclusive
- **Documentation:** The probability that a building gets painted at a cell in lieu of a tree. The higher this probability, the more likely a building tile gets painted instead of a tree tile.
- **Default value:** 0.125

2. point_radius

- **Type:** Floating point number (float) between 0.5 and 2.5 inclusive
- **Documentation:** The radius value used to measure distances between points for the algorithm. The longer the radius, the further apart points are during the algorithm's processing, and the further apart painted cells are in the game.
- **Default value:** 1.0

3. region_size

- **Type:** Vector2
- **Documentation:** The size of the region in which the algorithm is performed. Set to the "default" tile map size (72, 40) in the script, shown as (0, 0) in the Godot editor. Can be changed to use a smaller region for the algorithm itself, of course resulting in less cells covered within the boundaries set for this game, though the algorithm will perform faster due to less cells being checked.
- **Default value:**
 - **x:** The value in "x_tile_range" (72)
 - **y:** The value in "y_tile_range" (40)

4. rejection_samples

- **Type:** Integer (int) between 1 and 50 inclusive
- **Documentation:** The maximum number of times a cell is checked before it is ignored. A cell can be accepted and painted on before this maximum number is reached. The higher this value, the more times a cell is checked, therefore the higher the algorithm's processing time.
- **Default value:** 8

B.3.4 Voronoi Cells

1. distance

- **Type:** String enumeration of the following choices:
 - (a) “Euclidean distance”
 - (b) “Manhattan distance”
- **Documentation:** Determines whether or not the Euclidean or Manhattan distance formula is used for calculation of the deltas between points within Voronoi cells.
- **Default value:** “Manhattan distance”

2. random_starting_points

- **Type:** Integer (int) between 15 and 40 inclusive
- **Documentation:** Determines the number of points randomly picked from at the start. Therefore, it also determines the number of cells in our Voronoi tessellation.
- **Default value:** 20

B.3.5 The Basic L-System Demo Used to Create the Screenshots in Chapter 3.1.1

There is only one export variable for this: “choices”, which allows you to choose which one of the three provided rulesets to use. “choices” is the default ruleset, and either “deterministic” or “basic” can be chosen. It is in the “DemoNode” scene, the only scene of this Godot project. Quoting the documentation comment, this export variable “Allows you to decide which ruleset to use. See the script file for the sources of said rulesets.” The ruleset is assigned with the “set_values” function.

B.4 Running the Godot Projects

To *run* the current project in the Godot editor, go to the bar above the Inspector, Node and History tabs on the right-hand side. You will find a ► button which will play the main scene of the project (in the artefacts, the main scenes have already been set; if it were not already set, you would have been asked to set one). If closing the window to stop the project does not work, hit the ■ button to end it. If you want to replay the current project without having to stop it, hit the ► button again. Although **both** the icon **and** the colour of the ► button will have changed by then, it will be in the same position as before.

As described in section 3.2.1, only the close button of both the popup dialogs in the game seems to work properly for the moment, but this does not adversely affect the game functioning properly, nor does it adversely affect this project, so this is trivial.

Appendix C

Source Code

Contents

C.1	Instructions	86
C.2	GD4LSystemRPG	86
C.2.1	.gitattributes	86
C.2.2	.gitignore	86
C.2.3	project.godot	86
C.2.4	l_system.tscn	87
C.2.5	l_system.gd	87
C.2.6	tile_map.tscn	93
C.2.7	tile_map.gd	94
C.2.8	accept_dialog.tscn	98
C.2.9	win_dialog.tscn	98
C.2.10	icon.svg.import	99
C.2.11	monochrome_packed.png.import	100
C.2.12	Tiles.tres	101
C.2.13	LICENSE	135
C.3	GD4VoronoiRPG	135
C.3.1	.gitattributes	135
C.3.2	.gitignore	135
C.3.3	project.godot	135
C.3.4	tile_map.tscn	137
C.3.5	tile_map.gd	170
C.3.6	accept_dialog.tscn	178
C.3.7	win_dialog.tscn	179
C.3.8	icon.svg.import	179
C.3.9	monochrome_packed.png.import	181

C.3.10	LICENSE	182
C.4	GD4PoissonRPG	182
C.4.1	.gitattributes	182
C.4.2	.gitignore	182
C.4.3	project.godot	183
C.4.4	tile_map.tscn	183
C.4.5	tile_map.gd	217
C.4.6	accept_dialog.tscn	225
C.4.7	win_dialog.tscn	226
C.4.8	icon.svg.import	226
C.4.9	monochrome_packed.png.import	227
C.4.10	LICENSE	229
C.5	GD4NoiseRPG	229
C.5.1	.gitattributes	229
C.5.2	.gitignore	229
C.5.3	project.godot	229
C.5.4	tile_map.tscn	230
C.5.5	tile_map.gd	264
C.5.6	accept_dialog.tscn	271
C.5.7	win_dialog.tscn	272
C.5.8	icon.svg.import	272
C.5.9	monochrome_packed.png.import	273
C.5.10	LICENSE	275
C.6	LSystemGrammarDemo	275
C.6.1	.gitattributes	275
C.6.2	.gitignore	275
C.6.3	project.godot	275
C.6.4	DemoNode.tscn	276
C.6.5	DemoNode.gd	277
C.6.6	icon.svg.import	279
C.6.7	LICENSE	281
C.7	README	281

C.1 Instructions

If needed, use the table of contents provided to browse through the code listings in this section. Each listing folder will have a short description, a link to its public GitHub repository, and a listing for each readable source file. Use the .zip folder containing the project artefacts to edit and run them in Godot.

“I verify that I am the sole author of the programs contained in this folder, except where explicitly stated to the contrary.” - Zishan Rahman, 21st April 2023

C.2 GD4LSystemRPG

These are the source code listings for the L-System implementation of the 2D tile-map RPG scenario in the Godot game engine. The link to the publicly available source code repository is here: <https://github.com/Zishan-Rahman/GD4LSystemRPG>

C.2.1 .gitattributes

```
1  # Normalize EOL for all files that Git considers text files.
2  * text=auto eol=lf
```

C.2.2 .gitignore

```
1  # Godot 4+ specific ignores
2  .godot/
```

C.2.3 project.godot

```
1  ; Engine configuration file.
2  ; It's best edited using the editor UI and not directly,
3  ; since the parameters that go here are not all obvious.
4  ;
```

```

5   ; Format:
6   ;   [section] ; section goes between []
7   ;   param=value ; assign values to parameters
8
9   config_version=5
10
11  [application]
12
13  config/name="LSystem RPG"
14  run/main_scene="res://tile_map.tscn"
15  config/features=PackedStringArray("4.0", "Forward Plus")
16  config/icon="res://icon.svg"
17
18  [display]
19
20  window/size/viewport_height=640
21
22  [rendering]
23
24  environment/defaults/default_clear_color=Color(0, 0, 0, 1)

```

C.2.4 l_system.tscn

```

1   [gd_scene load_steps=2 format=3 uid="uid://d0v18e7ms571f"]
2
3   [ext_resource type="Script" path="res://l_system.gd" id="1_elydp"]
4
5   [node name="LSystem" type="Node"]
6   script = ExtResource("1_elydp")

```

C.2.5 l_system.gd

```

1  extends Node
2
3  class_name LSystem
4
5  @onready var tile_map: TileMap = get_parent()
6  ## The starting string from which the grammar starts applying its
   rules. Here it may be self defined, or randomly defined when
   use_random_axiom is true.
7  @export var axiom: String = "OWB"
8  ## Uses a random axiom with the currently set grammar, computed
   upon runtime, with a length up to (but not strictly) the value
   of upper_limit. For example, if upper_limit is set to 15, the
   generated axiom can be 15 characters, or it can be just 5
   characters.
9  @export var use_random_axiom: bool = true
10 ## Defines how many characters a random axiom can have MAXIMUM.
   Only used when use_random_axiom is true.
11 @export var upper_limit: int = 10
12 ## Allows the use of a customly defined ruleset made through
   amending the rules array in the editor.
13 @export var use_custom_ruleset: bool = false
14 @onready var string: String = axiom
15 ## Denotes a series of pre-defined rulesets for this L-System
   grammar, of alphabet O (blank space), W (trees and fauna) and B
   (buildings), that can be chosen and then used on runtime. Can
   choose between a default ruleset, a ruleset that produces more
   buildings, a ruleset that produces more trees and a ruleset
   that produces more empty space.
16 @export_enum("Default", "More Buildings (IMPOSSIBLE)", "More Trees"
   , "More Space") var ruleset: String = "Default"
17 ## The set of rules that the L-System grammar uses. Shows the "
   default" ruleset in the Godot editor for the user to see. If
   use_custom_ruleset is set to true, this array can be edited

```

with a custom defined ruleset that will be used on runtime, so long as it adheres to the alphabet of O (blank space), W (trees and fauna) and B (buildings).

```
18 @export var rules: Array[Dictionary] = DEFAULT
19
20 const DEFAULT: Array[Dictionary] = [
21   {
22     "from": "O",
23     "to": "OWO"
24   },
25   {
26     "from": "W",
27     "to": "WB"
28   },
29   {
30     "from": "B",
31     "to": "BWO"
32   }
33 ]
34 const MORE_BUILDINGS: Array[Dictionary] = [
35   {
36     "from": "O",
37     "to": "BWOB"
38   },
39   {
40     "from": "W",
41     "to": "WBOBO"
42   },
43   {
44     "from": "B",
45     "to": "BB"
46   }
47 ]
```

```

48  const MORE_TREES: Array[Dictionary] = [
49      {
50          "from": "O",
51          "to": "WWO"
52      },
53      {
54          "from": "W",
55          "to": "WBWO"
56      },
57      {
58          "from": "B",
59          "to": "BWWO"
60      }
61  ]
62  const MORE_SPACE: Array[Dictionary] = [
63      {
64          "from": "O",
65          "to": "OOBWO"
66      },
67      {
68          "from": "W",
69          "to": "OB"
70      },
71      {
72          "from": "B",
73          "to": "OW"
74      }
75  ]
76
77  const buildings: Array[Vector2i] = [
78      Vector2i(0, 19),
79      Vector2i(1, 19),
80      Vector2i(2, 19),

```



```

81     Vector2i(3, 19),
82     Vector2i(4, 19),
83     Vector2i(5, 19),
84     Vector2i(6, 19),
85     Vector2i(7, 19),
86     Vector2i(8, 20),
87     Vector2i(0, 20),
88     Vector2i(1, 20),
89     Vector2i(2, 20),
90     Vector2i(3, 20),
91     Vector2i(4, 20),
92     Vector2i(5, 20),
93     Vector2i(6, 20),
94     Vector2i(7, 20),
95     Vector2i(8, 20),
96     Vector2i(0, 21),
97     Vector2i(1, 21),
98     Vector2i(2, 21),
99     Vector2i(3, 21),
100    Vector2i(4, 21),
101    Vector2i(5, 21),
102    Vector2i(6, 21),
103    Vector2i(7, 21),
104    Vector2i(8, 21)
105 ]
106 const trees: Array[Vector2i] = [
107     Vector2i(0,1),
108     Vector2i(1,1),
109     Vector2i(2,1),
110     Vector2i(3,1),
111     Vector2i(4,1),
112     Vector2i(5,1),
113     Vector2i(6,1),

```

```

114     Vector2i(7,1),
115     Vector2i(0,2),
116     Vector2i(1,2),
117     Vector2i(2,2),
118     Vector2i(3,2),
119     Vector2i(4,2)
120 ]
121
122 func _get_ruleset() -> Array[Dictionary]:
123     match ruleset:
124         "More Buildings (IMPOSSIBLE)": return MORE_BUILDINGS
125         "More Trees": return MORE_TREES
126         "More Space": return MORE_SPACE
127         _: return DEFAULT
128
129 func get_new_replacement(character: String) -> String:
130     for rule in rules:
131         if rule["from"] == character:
132             return rule["to"]
133     return character
134
135 func _size() -> int:
136     return tile_map.x_tile_range * tile_map.y_tile_range
137
138 func rand_axiom() -> String:
139     var string_buffer: String = ""
140     var limit: int = randi_range(1, upper_limit)
141     for i in range(limit):
142         string_buffer += ["O", "W", "B"].pick_random()
143     return string_buffer
144
145 func parse() -> String:
146     if use_random_axiom:

```

```

147         axiom = rand_axiom()
148         string = axiom
149         if not use_custom_ruleset or ruleset != "Default":
150             rules = _get_ruleset()
151         print("Axiom length: " + str(len(axiom)))
152         var size: int = _size()
153         while len(string) <= size:
154             var new_string = ""
155             for character in string:
156                 new_string += get_new_replacement(character)
157             string = new_string
158         string = string.substr(0, size)
159         return string
160
161     func paint() -> void:
162         string = parse()
163         var i: int = -1
164         for x in range(tile_map.x_tile_range):
165             for y in range(tile_map.y_tile_range):
166                 i += 1
167                 if string[i] == "O": # "O" = BLANK
168                     pass # Do not paint any cell.
169                 elif string[i] == "W": # "W" = TREE
170                     tile_map.set_cell(0, Vector2i(x, y), 0, trees.
171                                     pick_random())
172                 elif string[i] == "B": # "B" = BUILDING
173                     tile_map.set_cell(0, Vector2i(x, y), 0, buildings.
174                                     pick_random())

```

C.2.6 tile_map.tscn

```

1     [gd_scene load_steps=6 format=3 uid="uid://bwhvtqld3yo8m"]

```

```

2
3  [ext_resource type="TileSet" uid="uid://c168x78r0tful" path="res://
    Tiles.tres" id="1_l3nwg"]
4  [ext_resource type="Script" path="res://tile_map.gd" id="2_wrxl8"]
5  [ext_resource type="PackedScene" uid="uid://d0v18e7ms571f" path="
    res://l_system.tscn" id="3_ktw1n"]
6  [ext_resource type="PackedScene" uid="uid://cau5jgogdnf53" path="
    res://accept_dialog.tscn" id="4_060oh"]
7  [ext_resource type="PackedScene" uid="uid://b5q8ovcigrvyr" path="
    res://win_dialog.tscn" id="5_3s48a"]
8
9  [node name="TileMap" type="TileMap"]
10 tile_set = ExtResource("1_l3nwg")
11 format = 2
12 layer_0/name = "Things"
13 script = ExtResource("2_wrxl8")
14
15 [node name="LSystem" parent="." instance=ExtResource("3_ktw1n")]
16
17 [node name="AcceptDialog" parent="." instance=ExtResource("4_060oh"
    )]
18
19 [node name="WinDialog" parent="." instance=ExtResource("5_3s48a")]

```

C.2.7 tile_map.gd

```

1  extends TileMap
2
3  @onready var l_system: LSystem = $LSystem
4
5  var x_tile_range: int = ProjectSettings.get_setting("display/window
    /size/viewport_width") / tile_set.tile_size.x

```

```

6  var y_tile_range: int = ProjectSettings.get_setting("display/window
    /size/viewport_height") / tile_set.tile_size.y
7
8  const PLAYER_SPRITE: Vector2i = Vector2i(24, 7)
9  var player_placement_cell: Vector2i
10 const rings: Array[Vector2i] = [
11     Vector2i(43, 6),
12     Vector2i(44, 6),
13     Vector2i(45, 6),
14     Vector2i(46, 6)
15 ]
16 var ring_placement_cell: Vector2i
17
18 # Called when the node enters the scene tree for the first time.
19 func _ready() -> void:
20     randomize()
21     var start_time: float = Time.get_ticks_msec()
22     l_system.paint()
23     place_player()
24     place_ring()
25     var new_time: float = Time.get_ticks_msec() - start_time
26     print("Time taken: " + str(new_time) + "ms")
27     $AcceptDialog.dialog_text = "You're a hollow Golem who seeks the
        ultimate treasure; a ring that's got something on top of it
        . It's somewhere in this large village and barely visible to
        your naked eyes, which took us " + str(new_time) + "
        milliseconds to generate (" + str(new_time / 1000.0) + "
        seconds), but you'll stop at nothing to get what you want.
        You can chow down every tree and fauna that stands in your
        way of the ring, but your Achilles heel is any bricks and
        mortar, which WILL make you stop at your tracks. Since it's
        easy to get lost in here, we'll tell you that you're in
        position " + str(player_placement_cell) + " in this big

```

```

    village of size " + str(Vector2i(x_tile_range, y_tile_range)
    ) + ". The ring is in position " + str(ring_placement_cell)
    + ", but it is YOUR job to traverse the village, chow down
    the trees and get it for yourself, so are you ready to
    attain the treasure that is rightfully yours?!"

28     $AcceptDialog.visible = true
29     $AcceptDialog.confirmed.connect(_on_AcceptDialog_closed)
30     $AcceptDialog.canceled.connect(_on_AcceptDialog_closed)
31     $WinDialog.confirmed.connect(_on_WinDialog_confirmed)
32     $WinDialog.canceled.connect(_on_WinDialog_canceled)
33     get_tree().paused = true
34
35     func _on_WinDialog_confirmed() -> void:
36         get_tree().reload_current_scene()
37
38     func _on_WinDialog_canceled() -> void:
39         get_tree().quit()
40
41     func _on_AcceptDialog_closed() -> void:
42         $AcceptDialog.visible = false
43         get_tree().paused = false
44
45     func _get_random_placement_cell() -> Vector2i:
46         return Vector2i(randi() % x_tile_range, randi() % y_tile_range)
47
48     func place_player() -> void:
49         player_placement_cell = _get_random_placement_cell()
50         while get_used_cells(0).has(player_placement_cell):
51             player_placement_cell = _get_random_placement_cell()
52         set_cell(0, player_placement_cell, 0, PLAYER_SPRITE)
53
54     func place_ring() -> void:
55         ring_placement_cell = _get_random_placement_cell()

```

```

56     while get_used_cells(0).has(ring_placement_cell):
57         ring_placement_cell = _get_random_placement_cell()
58     set_cell(0, ring_placement_cell, 0, rings.pick_random())
59
60     func _is_not_out_of_bounds(cell: Vector2i) -> bool:
61         return cell.x >= 0 and cell.x < x_tile_range and cell.y >= 0 and
            cell.y < y_tile_range
62
63     func _physics_process(_delta: float) -> void:
64         var previous_cell: Vector2i = player_placement_cell
65         var direction: Vector2i = Vector2i.ZERO
66         if Input.is_action_pressed("ui_up"): direction = Vector2i.UP
67         elif Input.is_action_pressed("ui_down"): direction = Vector2i.
            DOWN
68         elif Input.is_action_pressed("ui_left"): direction = Vector2i.
            LEFT
69         elif Input.is_action_pressed("ui_right"): direction = Vector2i.
            RIGHT
70         var new_placement_cell: Vector2i = player_placement_cell +
            direction
71         if (not get_used_cells(0).has(new_placement_cell) or l_system.
            trees.has(get_cell_atlas_coords(0, new_placement_cell)) or
            new_placement_cell == ring_placement_cell) and
            _is_not_out_of_bounds(new_placement_cell):
72             player_placement_cell = new_placement_cell
73             set_cell(0, previous_cell, 0) # deletes contents of previous
                cell (atlas_coords = Vector2i(-1, -1))
74             set_cell(0, player_placement_cell, 0, PLAYER_SPRITE)
75             if player_placement_cell == ring_placement_cell:
76                 $WinDialog.visible = true
77                 get_tree().paused = true
78
79     # ALGORITHM AND CUSTOM EXPORT VARIABLES ARE IN LSYSTEM NODE

```

C.2.8 accept_dialog.tscn

```
1  [gd_scene format=3 uid="uid://cau5jgogdnf53"]
2
3  [node name="AcceptDialog" type="AcceptDialog"]
4  title = "Tree-Munching Time!"
5  position = Vector2i(326, 100)
6  size = Vector2i(500, 421)
7  mouse_passthrough = true
8  ok_button_text = "Bring it on!"
9  dialog_text = "You're a hollow Golem who seeks the ultimate
    treasure; a ring that's got something on top of it. It's
    somewhere in this large village and barely visible to your
    naked eyes, but you'll stop at nothing to get what you want.
    You can chow down every tree and fauna that stands in your way
    of the ring, but your Achilles heel is any bricks and mortar,
    which will make you stop at your tracks. Are you ready to
    attain your treasure?w Golem in a black-and-white world, in
    search for your most desired treasure. It's a ring with
    something on top of it. And you'll stop at nothing to get what
    you want. You can chow down every tree and fauna that stands in
    your way of the ring, but your Achilles heel is any bricks and
    mortar, which will make you stop at your tracks. Are you ready
    to attain the treasure that is rightfully yours?!"
10 dialog_autowrap = true
```

C.2.9 win_dialog.tscn

```
1  [gd_scene format=3 uid="uid://b5q8ovcigrvyr"]
2
3  [node name="WinDialog" type="ConfirmationDialog"]
4  title = "You Found the Treasure!"
```



```

5   position = Vector2i(326, 100)
6   size = Vector2i(500, 421)
7   mouse_passthrough = true
8   ok_button_text = "Get Me a New Village"
9   dialog_text = "You found your treasure! Well done, you!"
10
11   Would you like to travel to a new village in the hopes of finding
        another ring? Or would you like to take your treasure home now?
        "
12   dialog_autowrap = true
13   cancel_button_text = "Get Me Out of Here"

```

C.2.10 icon.svg.import

```

1   [remap]
2
3   importer="texture"
4   type="CompressedTexture2D"
5   uid="uid://b45qexb3wmhym"
6   path="res://.godot/imported/icon.svg-218
        a8f2b3041327d8a5756f3a245f83b.ctex"
7   metadata={
8     "vram_texture": false
9   }
10
11   [deps]
12
13   source_file="res://icon.svg"
14   dest_files=["res://.godot/imported/icon.svg-218
        a8f2b3041327d8a5756f3a245f83b.ctex"]
15
16   [params]

```

```

17
18   compress/mode=0
19   compress/high_quality=false
20   compress/lossy_quality=0.7
21   compress/hdr_compression=1
22   compress/normal_map=0
23   compress/channel_pack=0
24   mipmaps/generate=false
25   mipmaps/limit=-1
26   roughness/mode=0
27   roughness/src_normal=""
28   process/fix_alpha_border=true
29   process/premult_alpha=false
30   process/normal_map_invert_y=false
31   process/hdr_as_srgb=false
32   process/hdr_clamp_exposure=false
33   process/size_limit=0
34   detect_3d/compress_to=1
35   svg/scale=1.0
36   editor/scale_with_editor_scale=false
37   editor/convert_colors_with_editor_theme=false

```

C.2.11 monochrome_packed.png.import

```

1   [remap]
2
3   importer="texture"
4   type="CompressedTexture2D"
5   uid="uid://dic8oic1ybjoyq"
6   path="res://.godot/imported/monochrome_packed.png-6
      b9bd1c64dd50f72acd3afd14d1ac34f.ctex"
7   metadata={

```

```

8  "vram_texture": false
9  }
10
11  [deps]
12
13  source_file="res://monochrome_packed.png"
14  dest_files=["res://.godot/imported/monochrome_packed.png-6
           b9bd1c64dd50f72acd3afd14d1ac34f.ctex"]
15
16  [params]
17
18  compress/mode=0
19  compress/high_quality=false
20  compress/lossy_quality=0.7
21  compress/hdr_compression=1
22  compress/normal_map=0
23  compress/channel_pack=0
24  mipmaps/generate=false
25  mipmaps/limit=-1
26  roughness/mode=0
27  roughness/src_normal=""
28  process/fix_alpha_border=true
29  process/premult_alpha=false
30  process/normal_map_invert_y=false
31  process/hdr_as_srgb=false
32  process/hdr_clamp_exposure=false
33  process/size_limit=0
34  detect_3d/compress_to=1

```

C.2.12 Tiles.tres

```

1  [gd_resource type="TileSet" load_steps=3 format=3 uid="uid://"

```

```

    c168x78r0tful"]
2
3  [ext_resource type="Texture2D" uid="uid://dic8oic1ybjyq" path="res
    ://monochrome_packed.png" id="1_fqi6r"]
4
5  [sub_resource type="TileSetAtlasSource" id="
    TileSetAtlasSource_qer06"]
6  texture = ExtResource("1_fqi6r")
7  0:0/0 = 0
8  1:0/0 = 0
9  2:0/0 = 0
10 3:0/0 = 0
11 4:0/0 = 0
12 5:0/0 = 0
13 6:0/0 = 0
14 7:0/0 = 0
15 8:0/0 = 0
16 9:0/0 = 0
17 10:0/0 = 0
18 11:0/0 = 0
19 12:0/0 = 0
20 13:0/0 = 0
21 14:0/0 = 0
22 15:0/0 = 0
23 16:0/0 = 0
24 17:0/0 = 0
25 18:0/0 = 0
26 19:0/0 = 0
27 20:0/0 = 0
28 21:0/0 = 0
29 22:0/0 = 0
30 23:0/0 = 0
31 24:0/0 = 0

```

32 25:0/0 = 0
33 26:0/0 = 0
34 27:0/0 = 0
35 28:0/0 = 0
36 29:0/0 = 0
37 30:0/0 = 0
38 31:0/0 = 0
39 32:0/0 = 0
40 33:0/0 = 0
41 34:0/0 = 0
42 35:0/0 = 0
43 36:0/0 = 0
44 37:0/0 = 0
45 38:0/0 = 0
46 39:0/0 = 0
47 40:0/0 = 0
48 41:0/0 = 0
49 42:0/0 = 0
50 43:0/0 = 0
51 44:0/0 = 0
52 45:0/0 = 0
53 46:0/0 = 0
54 47:0/0 = 0
55 48:0/0 = 0
56 0:1/0 = 0
57 1:1/0 = 0
58 2:1/0 = 0
59 3:1/0 = 0
60 4:1/0 = 0
61 5:1/0 = 0
62 6:1/0 = 0
63 7:1/0 = 0
64 8:1/0 = 0

65 9:1/0 = 0
66 10:1/0 = 0
67 11:1/0 = 0
68 12:1/0 = 0
69 13:1/0 = 0
70 14:1/0 = 0
71 15:1/0 = 0
72 16:1/0 = 0
73 17:1/0 = 0
74 18:1/0 = 0
75 19:1/0 = 0
76 20:1/0 = 0
77 21:1/0 = 0
78 22:1/0 = 0
79 23:1/0 = 0
80 24:1/0 = 0
81 25:1/0 = 0
82 26:1/0 = 0
83 27:1/0 = 0
84 28:1/0 = 0
85 29:1/0 = 0
86 30:1/0 = 0
87 31:1/0 = 0
88 32:1/0 = 0
89 33:1/0 = 0
90 34:1/0 = 0
91 35:1/0 = 0
92 36:1/0 = 0
93 37:1/0 = 0
94 38:1/0 = 0
95 39:1/0 = 0
96 40:1/0 = 0
97 41:1/0 = 0

98	$42:1/0 = 0$
99	$43:1/0 = 0$
100	$44:1/0 = 0$
101	$45:1/0 = 0$
102	$46:1/0 = 0$
103	$47:1/0 = 0$
104	$48:1/0 = 0$
105	$0:2/0 = 0$
106	$1:2/0 = 0$
107	$2:2/0 = 0$
108	$3:2/0 = 0$
109	$4:2/0 = 0$
110	$5:2/0 = 0$
111	$6:2/0 = 0$
112	$7:2/0 = 0$
113	$8:2/0 = 0$
114	$9:2/0 = 0$
115	$10:2/0 = 0$
116	$11:2/0 = 0$
117	$12:2/0 = 0$
118	$13:2/0 = 0$
119	$14:2/0 = 0$
120	$15:2/0 = 0$
121	$16:2/0 = 0$
122	$17:2/0 = 0$
123	$18:2/0 = 0$
124	$19:2/0 = 0$
125	$20:2/0 = 0$
126	$21:2/0 = 0$
127	$22:2/0 = 0$
128	$23:2/0 = 0$
129	$24:2/0 = 0$
130	$25:2/0 = 0$

131	$26:2/0 = 0$
132	$27:2/0 = 0$
133	$28:2/0 = 0$
134	$29:2/0 = 0$
135	$30:2/0 = 0$
136	$31:2/0 = 0$
137	$32:2/0 = 0$
138	$33:2/0 = 0$
139	$34:2/0 = 0$
140	$35:2/0 = 0$
141	$36:2/0 = 0$
142	$37:2/0 = 0$
143	$38:2/0 = 0$
144	$39:2/0 = 0$
145	$40:2/0 = 0$
146	$41:2/0 = 0$
147	$42:2/0 = 0$
148	$43:2/0 = 0$
149	$44:2/0 = 0$
150	$45:2/0 = 0$
151	$46:2/0 = 0$
152	$47:2/0 = 0$
153	$48:2/0 = 0$
154	$0:3/0 = 0$
155	$1:3/0 = 0$
156	$2:3/0 = 0$
157	$3:3/0 = 0$
158	$4:3/0 = 0$
159	$5:3/0 = 0$
160	$6:3/0 = 0$
161	$7:3/0 = 0$
162	$8:3/0 = 0$
163	$9:3/0 = 0$

164 10:3/0 = 0
165 11:3/0 = 0
166 12:3/0 = 0
167 13:3/0 = 0
168 14:3/0 = 0
169 15:3/0 = 0
170 16:3/0 = 0
171 17:3/0 = 0
172 18:3/0 = 0
173 19:3/0 = 0
174 20:3/0 = 0
175 21:3/0 = 0
176 22:3/0 = 0
177 23:3/0 = 0
178 24:3/0 = 0
179 25:3/0 = 0
180 26:3/0 = 0
181 27:3/0 = 0
182 28:3/0 = 0
183 29:3/0 = 0
184 30:3/0 = 0
185 31:3/0 = 0
186 32:3/0 = 0
187 33:3/0 = 0
188 34:3/0 = 0
189 35:3/0 = 0
190 36:3/0 = 0
191 37:3/0 = 0
192 38:3/0 = 0
193 39:3/0 = 0
194 40:3/0 = 0
195 41:3/0 = 0
196 42:3/0 = 0

197 $43:3/0 = 0$
198 $44:3/0 = 0$
199 $45:3/0 = 0$
200 $46:3/0 = 0$
201 $47:3/0 = 0$
202 $48:3/0 = 0$
203 $0:4/0 = 0$
204 $1:4/0 = 0$
205 $2:4/0 = 0$
206 $3:4/0 = 0$
207 $4:4/0 = 0$
208 $5:4/0 = 0$
209 $6:4/0 = 0$
210 $7:4/0 = 0$
211 $8:4/0 = 0$
212 $9:4/0 = 0$
213 $10:4/0 = 0$
214 $11:4/0 = 0$
215 $12:4/0 = 0$
216 $13:4/0 = 0$
217 $14:4/0 = 0$
218 $15:4/0 = 0$
219 $16:4/0 = 0$
220 $17:4/0 = 0$
221 $18:4/0 = 0$
222 $19:4/0 = 0$
223 $20:4/0 = 0$
224 $21:4/0 = 0$
225 $22:4/0 = 0$
226 $23:4/0 = 0$
227 $24:4/0 = 0$
228 $25:4/0 = 0$
229 $26:4/0 = 0$

230	$27:4/0 = 0$
231	$28:4/0 = 0$
232	$29:4/0 = 0$
233	$30:4/0 = 0$
234	$31:4/0 = 0$
235	$32:4/0 = 0$
236	$33:4/0 = 0$
237	$34:4/0 = 0$
238	$35:4/0 = 0$
239	$36:4/0 = 0$
240	$37:4/0 = 0$
241	$38:4/0 = 0$
242	$39:4/0 = 0$
243	$40:4/0 = 0$
244	$41:4/0 = 0$
245	$42:4/0 = 0$
246	$43:4/0 = 0$
247	$44:4/0 = 0$
248	$45:4/0 = 0$
249	$46:4/0 = 0$
250	$47:4/0 = 0$
251	$48:4/0 = 0$
252	$0:5/0 = 0$
253	$1:5/0 = 0$
254	$2:5/0 = 0$
255	$3:5/0 = 0$
256	$4:5/0 = 0$
257	$5:5/0 = 0$
258	$6:5/0 = 0$
259	$7:5/0 = 0$
260	$8:5/0 = 0$
261	$9:5/0 = 0$
262	$10:5/0 = 0$

263	$11:5/0 = 0$
264	$12:5/0 = 0$
265	$13:5/0 = 0$
266	$14:5/0 = 0$
267	$15:5/0 = 0$
268	$16:5/0 = 0$
269	$17:5/0 = 0$
270	$18:5/0 = 0$
271	$19:5/0 = 0$
272	$20:5/0 = 0$
273	$21:5/0 = 0$
274	$22:5/0 = 0$
275	$23:5/0 = 0$
276	$24:5/0 = 0$
277	$25:5/0 = 0$
278	$26:5/0 = 0$
279	$27:5/0 = 0$
280	$28:5/0 = 0$
281	$29:5/0 = 0$
282	$30:5/0 = 0$
283	$31:5/0 = 0$
284	$32:5/0 = 0$
285	$33:5/0 = 0$
286	$34:5/0 = 0$
287	$35:5/0 = 0$
288	$36:5/0 = 0$
289	$37:5/0 = 0$
290	$38:5/0 = 0$
291	$39:5/0 = 0$
292	$40:5/0 = 0$
293	$41:5/0 = 0$
294	$42:5/0 = 0$
295	$43:5/0 = 0$

296	$44:5/0 = 0$
297	$45:5/0 = 0$
298	$46:5/0 = 0$
299	$47:5/0 = 0$
300	$48:5/0 = 0$
301	$0:6/0 = 0$
302	$1:6/0 = 0$
303	$2:6/0 = 0$
304	$3:6/0 = 0$
305	$4:6/0 = 0$
306	$5:6/0 = 0$
307	$6:6/0 = 0$
308	$7:6/0 = 0$
309	$8:6/0 = 0$
310	$9:6/0 = 0$
311	$10:6/0 = 0$
312	$11:6/0 = 0$
313	$12:6/0 = 0$
314	$13:6/0 = 0$
315	$14:6/0 = 0$
316	$15:6/0 = 0$
317	$16:6/0 = 0$
318	$17:6/0 = 0$
319	$18:6/0 = 0$
320	$19:6/0 = 0$
321	$20:6/0 = 0$
322	$21:6/0 = 0$
323	$22:6/0 = 0$
324	$23:6/0 = 0$
325	$24:6/0 = 0$
326	$25:6/0 = 0$
327	$26:6/0 = 0$
328	$27:6/0 = 0$

329 28:6/0 = 0
330 29:6/0 = 0
331 30:6/0 = 0
332 31:6/0 = 0
333 32:6/0 = 0
334 33:6/0 = 0
335 34:6/0 = 0
336 35:6/0 = 0
337 36:6/0 = 0
338 37:6/0 = 0
339 38:6/0 = 0
340 39:6/0 = 0
341 40:6/0 = 0
342 41:6/0 = 0
343 42:6/0 = 0
344 43:6/0 = 0
345 44:6/0 = 0
346 45:6/0 = 0
347 46:6/0 = 0
348 47:6/0 = 0
349 48:6/0 = 0
350 0:7/0 = 0
351 1:7/0 = 0
352 2:7/0 = 0
353 3:7/0 = 0
354 4:7/0 = 0
355 5:7/0 = 0
356 6:7/0 = 0
357 7:7/0 = 0
358 8:7/0 = 0
359 9:7/0 = 0
360 10:7/0 = 0
361 11:7/0 = 0

362 12:7/0 = 0
363 13:7/0 = 0
364 14:7/0 = 0
365 15:7/0 = 0
366 16:7/0 = 0
367 17:7/0 = 0
368 18:7/0 = 0
369 19:7/0 = 0
370 20:7/0 = 0
371 21:7/0 = 0
372 22:7/0 = 0
373 23:7/0 = 0
374 24:7/0 = 0
375 25:7/0 = 0
376 26:7/0 = 0
377 27:7/0 = 0
378 28:7/0 = 0
379 29:7/0 = 0
380 30:7/0 = 0
381 31:7/0 = 0
382 32:7/0 = 0
383 33:7/0 = 0
384 34:7/0 = 0
385 35:7/0 = 0
386 36:7/0 = 0
387 37:7/0 = 0
388 38:7/0 = 0
389 39:7/0 = 0
390 40:7/0 = 0
391 41:7/0 = 0
392 42:7/0 = 0
393 43:7/0 = 0
394 44:7/0 = 0

395 $45:7/0 = 0$
396 $46:7/0 = 0$
397 $47:7/0 = 0$
398 $48:7/0 = 0$
399 $0:8/0 = 0$
400 $1:8/0 = 0$
401 $2:8/0 = 0$
402 $3:8/0 = 0$
403 $4:8/0 = 0$
404 $5:8/0 = 0$
405 $6:8/0 = 0$
406 $7:8/0 = 0$
407 $8:8/0 = 0$
408 $9:8/0 = 0$
409 $10:8/0 = 0$
410 $11:8/0 = 0$
411 $12:8/0 = 0$
412 $13:8/0 = 0$
413 $14:8/0 = 0$
414 $15:8/0 = 0$
415 $16:8/0 = 0$
416 $17:8/0 = 0$
417 $18:8/0 = 0$
418 $19:8/0 = 0$
419 $20:8/0 = 0$
420 $21:8/0 = 0$
421 $22:8/0 = 0$
422 $23:8/0 = 0$
423 $24:8/0 = 0$
424 $25:8/0 = 0$
425 $26:8/0 = 0$
426 $27:8/0 = 0$
427 $28:8/0 = 0$

428 $29:8/0 = 0$
429 $30:8/0 = 0$
430 $31:8/0 = 0$
431 $32:8/0 = 0$
432 $33:8/0 = 0$
433 $34:8/0 = 0$
434 $35:8/0 = 0$
435 $36:8/0 = 0$
436 $37:8/0 = 0$
437 $38:8/0 = 0$
438 $39:8/0 = 0$
439 $40:8/0 = 0$
440 $41:8/0 = 0$
441 $42:8/0 = 0$
442 $43:8/0 = 0$
443 $44:8/0 = 0$
444 $45:8/0 = 0$
445 $46:8/0 = 0$
446 $47:8/0 = 0$
447 $48:8/0 = 0$
448 $0:9/0 = 0$
449 $1:9/0 = 0$
450 $2:9/0 = 0$
451 $3:9/0 = 0$
452 $4:9/0 = 0$
453 $5:9/0 = 0$
454 $6:9/0 = 0$
455 $7:9/0 = 0$
456 $8:9/0 = 0$
457 $9:9/0 = 0$
458 $10:9/0 = 0$
459 $11:9/0 = 0$
460 $12:9/0 = 0$

461 13:9/0 = 0
462 14:9/0 = 0
463 15:9/0 = 0
464 16:9/0 = 0
465 17:9/0 = 0
466 18:9/0 = 0
467 19:9/0 = 0
468 20:9/0 = 0
469 21:9/0 = 0
470 22:9/0 = 0
471 23:9/0 = 0
472 24:9/0 = 0
473 25:9/0 = 0
474 26:9/0 = 0
475 27:9/0 = 0
476 28:9/0 = 0
477 29:9/0 = 0
478 30:9/0 = 0
479 31:9/0 = 0
480 32:9/0 = 0
481 33:9/0 = 0
482 34:9/0 = 0
483 35:9/0 = 0
484 36:9/0 = 0
485 37:9/0 = 0
486 38:9/0 = 0
487 39:9/0 = 0
488 40:9/0 = 0
489 41:9/0 = 0
490 42:9/0 = 0
491 43:9/0 = 0
492 44:9/0 = 0
493 45:9/0 = 0

494 46:9/0 = 0
495 47:9/0 = 0
496 48:9/0 = 0
497 0:10/0 = 0
498 1:10/0 = 0
499 2:10/0 = 0
500 3:10/0 = 0
501 4:10/0 = 0
502 5:10/0 = 0
503 6:10/0 = 0
504 7:10/0 = 0
505 8:10/0 = 0
506 9:10/0 = 0
507 10:10/0 = 0
508 11:10/0 = 0
509 12:10/0 = 0
510 13:10/0 = 0
511 14:10/0 = 0
512 15:10/0 = 0
513 16:10/0 = 0
514 17:10/0 = 0
515 18:10/0 = 0
516 19:10/0 = 0
517 20:10/0 = 0
518 21:10/0 = 0
519 22:10/0 = 0
520 23:10/0 = 0
521 24:10/0 = 0
522 25:10/0 = 0
523 26:10/0 = 0
524 27:10/0 = 0
525 28:10/0 = 0
526 29:10/0 = 0

527 30:10/0 = 0
528 31:10/0 = 0
529 32:10/0 = 0
530 33:10/0 = 0
531 34:10/0 = 0
532 35:10/0 = 0
533 36:10/0 = 0
534 37:10/0 = 0
535 38:10/0 = 0
536 39:10/0 = 0
537 40:10/0 = 0
538 41:10/0 = 0
539 42:10/0 = 0
540 43:10/0 = 0
541 44:10/0 = 0
542 45:10/0 = 0
543 46:10/0 = 0
544 47:10/0 = 0
545 48:10/0 = 0
546 0:11/0 = 0
547 1:11/0 = 0
548 2:11/0 = 0
549 3:11/0 = 0
550 4:11/0 = 0
551 5:11/0 = 0
552 6:11/0 = 0
553 7:11/0 = 0
554 8:11/0 = 0
555 9:11/0 = 0
556 10:11/0 = 0
557 11:11/0 = 0
558 12:11/0 = 0
559 13:11/0 = 0

560 14:11/0 = 0
561 15:11/0 = 0
562 16:11/0 = 0
563 17:11/0 = 0
564 18:11/0 = 0
565 19:11/0 = 0
566 20:11/0 = 0
567 21:11/0 = 0
568 22:11/0 = 0
569 23:11/0 = 0
570 24:11/0 = 0
571 25:11/0 = 0
572 26:11/0 = 0
573 27:11/0 = 0
574 28:11/0 = 0
575 29:11/0 = 0
576 30:11/0 = 0
577 31:11/0 = 0
578 32:11/0 = 0
579 33:11/0 = 0
580 34:11/0 = 0
581 35:11/0 = 0
582 36:11/0 = 0
583 37:11/0 = 0
584 38:11/0 = 0
585 39:11/0 = 0
586 40:11/0 = 0
587 41:11/0 = 0
588 42:11/0 = 0
589 43:11/0 = 0
590 44:11/0 = 0
591 45:11/0 = 0
592 46:11/0 = 0

593 47:11/0 = 0
594 48:11/0 = 0
595 0:12/0 = 0
596 1:12/0 = 0
597 2:12/0 = 0
598 3:12/0 = 0
599 4:12/0 = 0
600 5:12/0 = 0
601 6:12/0 = 0
602 7:12/0 = 0
603 8:12/0 = 0
604 9:12/0 = 0
605 10:12/0 = 0
606 11:12/0 = 0
607 12:12/0 = 0
608 13:12/0 = 0
609 14:12/0 = 0
610 15:12/0 = 0
611 16:12/0 = 0
612 17:12/0 = 0
613 18:12/0 = 0
614 19:12/0 = 0
615 20:12/0 = 0
616 21:12/0 = 0
617 22:12/0 = 0
618 23:12/0 = 0
619 24:12/0 = 0
620 25:12/0 = 0
621 26:12/0 = 0
622 27:12/0 = 0
623 28:12/0 = 0
624 29:12/0 = 0
625 30:12/0 = 0

626 31:12/0 = 0
627 32:12/0 = 0
628 33:12/0 = 0
629 34:12/0 = 0
630 35:12/0 = 0
631 36:12/0 = 0
632 37:12/0 = 0
633 38:12/0 = 0
634 39:12/0 = 0
635 40:12/0 = 0
636 41:12/0 = 0
637 42:12/0 = 0
638 43:12/0 = 0
639 44:12/0 = 0
640 45:12/0 = 0
641 46:12/0 = 0
642 47:12/0 = 0
643 48:12/0 = 0
644 0:13/0 = 0
645 1:13/0 = 0
646 2:13/0 = 0
647 3:13/0 = 0
648 4:13/0 = 0
649 5:13/0 = 0
650 6:13/0 = 0
651 7:13/0 = 0
652 8:13/0 = 0
653 9:13/0 = 0
654 10:13/0 = 0
655 11:13/0 = 0
656 12:13/0 = 0
657 13:13/0 = 0
658 14:13/0 = 0

659 15:13/0 = 0
660 16:13/0 = 0
661 17:13/0 = 0
662 18:13/0 = 0
663 19:13/0 = 0
664 20:13/0 = 0
665 21:13/0 = 0
666 22:13/0 = 0
667 23:13/0 = 0
668 24:13/0 = 0
669 25:13/0 = 0
670 26:13/0 = 0
671 27:13/0 = 0
672 28:13/0 = 0
673 29:13/0 = 0
674 30:13/0 = 0
675 31:13/0 = 0
676 32:13/0 = 0
677 33:13/0 = 0
678 34:13/0 = 0
679 35:13/0 = 0
680 36:13/0 = 0
681 37:13/0 = 0
682 38:13/0 = 0
683 39:13/0 = 0
684 40:13/0 = 0
685 41:13/0 = 0
686 42:13/0 = 0
687 43:13/0 = 0
688 44:13/0 = 0
689 45:13/0 = 0
690 46:13/0 = 0
691 47:13/0 = 0

692	$48:13/0 = 0$
693	$0:14/0 = 0$
694	$1:14/0 = 0$
695	$2:14/0 = 0$
696	$3:14/0 = 0$
697	$4:14/0 = 0$
698	$5:14/0 = 0$
699	$6:14/0 = 0$
700	$7:14/0 = 0$
701	$8:14/0 = 0$
702	$9:14/0 = 0$
703	$10:14/0 = 0$
704	$11:14/0 = 0$
705	$12:14/0 = 0$
706	$13:14/0 = 0$
707	$14:14/0 = 0$
708	$15:14/0 = 0$
709	$16:14/0 = 0$
710	$17:14/0 = 0$
711	$18:14/0 = 0$
712	$19:14/0 = 0$
713	$20:14/0 = 0$
714	$21:14/0 = 0$
715	$22:14/0 = 0$
716	$23:14/0 = 0$
717	$24:14/0 = 0$
718	$25:14/0 = 0$
719	$26:14/0 = 0$
720	$27:14/0 = 0$
721	$28:14/0 = 0$
722	$29:14/0 = 0$
723	$30:14/0 = 0$
724	$31:14/0 = 0$

725 32:14/0 = 0
726 33:14/0 = 0
727 34:14/0 = 0
728 35:14/0 = 0
729 36:14/0 = 0
730 37:14/0 = 0
731 38:14/0 = 0
732 39:14/0 = 0
733 40:14/0 = 0
734 41:14/0 = 0
735 42:14/0 = 0
736 43:14/0 = 0
737 44:14/0 = 0
738 45:14/0 = 0
739 46:14/0 = 0
740 47:14/0 = 0
741 48:14/0 = 0
742 0:15/0 = 0
743 1:15/0 = 0
744 2:15/0 = 0
745 3:15/0 = 0
746 4:15/0 = 0
747 5:15/0 = 0
748 6:15/0 = 0
749 7:15/0 = 0
750 8:15/0 = 0
751 9:15/0 = 0
752 10:15/0 = 0
753 11:15/0 = 0
754 12:15/0 = 0
755 13:15/0 = 0
756 14:15/0 = 0
757 15:15/0 = 0

758 16:15/0 = 0
759 17:15/0 = 0
760 18:15/0 = 0
761 19:15/0 = 0
762 20:15/0 = 0
763 21:15/0 = 0
764 22:15/0 = 0
765 23:15/0 = 0
766 24:15/0 = 0
767 25:15/0 = 0
768 26:15/0 = 0
769 27:15/0 = 0
770 28:15/0 = 0
771 29:15/0 = 0
772 30:15/0 = 0
773 31:15/0 = 0
774 32:15/0 = 0
775 33:15/0 = 0
776 34:15/0 = 0
777 35:15/0 = 0
778 36:15/0 = 0
779 37:15/0 = 0
780 38:15/0 = 0
781 39:15/0 = 0
782 40:15/0 = 0
783 41:15/0 = 0
784 42:15/0 = 0
785 43:15/0 = 0
786 44:15/0 = 0
787 45:15/0 = 0
788 46:15/0 = 0
789 47:15/0 = 0
790 48:15/0 = 0

791 0:16/0 = 0
792 1:16/0 = 0
793 2:16/0 = 0
794 3:16/0 = 0
795 4:16/0 = 0
796 5:16/0 = 0
797 6:16/0 = 0
798 7:16/0 = 0
799 8:16/0 = 0
800 9:16/0 = 0
801 10:16/0 = 0
802 11:16/0 = 0
803 12:16/0 = 0
804 13:16/0 = 0
805 14:16/0 = 0
806 15:16/0 = 0
807 16:16/0 = 0
808 17:16/0 = 0
809 18:16/0 = 0
810 19:16/0 = 0
811 20:16/0 = 0
812 21:16/0 = 0
813 22:16/0 = 0
814 23:16/0 = 0
815 24:16/0 = 0
816 25:16/0 = 0
817 26:16/0 = 0
818 27:16/0 = 0
819 28:16/0 = 0
820 29:16/0 = 0
821 30:16/0 = 0
822 31:16/0 = 0
823 32:16/0 = 0

824 33:16/0 = 0
825 34:16/0 = 0
826 35:16/0 = 0
827 36:16/0 = 0
828 37:16/0 = 0
829 38:16/0 = 0
830 39:16/0 = 0
831 40:16/0 = 0
832 41:16/0 = 0
833 42:16/0 = 0
834 43:16/0 = 0
835 44:16/0 = 0
836 45:16/0 = 0
837 46:16/0 = 0
838 47:16/0 = 0
839 48:16/0 = 0
840 0:17/0 = 0
841 1:17/0 = 0
842 2:17/0 = 0
843 3:17/0 = 0
844 4:17/0 = 0
845 5:17/0 = 0
846 6:17/0 = 0
847 7:17/0 = 0
848 8:17/0 = 0
849 9:17/0 = 0
850 10:17/0 = 0
851 11:17/0 = 0
852 12:17/0 = 0
853 13:17/0 = 0
854 14:17/0 = 0
855 15:17/0 = 0
856 16:17/0 = 0

857 17:17/0 = 0
858 18:17/0 = 0
859 19:17/0 = 0
860 20:17/0 = 0
861 21:17/0 = 0
862 22:17/0 = 0
863 23:17/0 = 0
864 24:17/0 = 0
865 25:17/0 = 0
866 26:17/0 = 0
867 27:17/0 = 0
868 28:17/0 = 0
869 29:17/0 = 0
870 30:17/0 = 0
871 31:17/0 = 0
872 32:17/0 = 0
873 33:17/0 = 0
874 34:17/0 = 0
875 35:17/0 = 0
876 36:17/0 = 0
877 37:17/0 = 0
878 38:17/0 = 0
879 39:17/0 = 0
880 40:17/0 = 0
881 41:17/0 = 0
882 42:17/0 = 0
883 43:17/0 = 0
884 44:17/0 = 0
885 45:17/0 = 0
886 46:17/0 = 0
887 47:17/0 = 0
888 48:17/0 = 0
889 0:18/0 = 0

890 1:18/0 = 0
891 2:18/0 = 0
892 3:18/0 = 0
893 4:18/0 = 0
894 5:18/0 = 0
895 6:18/0 = 0
896 7:18/0 = 0
897 8:18/0 = 0
898 9:18/0 = 0
899 10:18/0 = 0
900 11:18/0 = 0
901 12:18/0 = 0
902 13:18/0 = 0
903 14:18/0 = 0
904 15:18/0 = 0
905 16:18/0 = 0
906 17:18/0 = 0
907 18:18/0 = 0
908 19:18/0 = 0
909 20:18/0 = 0
910 21:18/0 = 0
911 22:18/0 = 0
912 23:18/0 = 0
913 24:18/0 = 0
914 25:18/0 = 0
915 26:18/0 = 0
916 27:18/0 = 0
917 28:18/0 = 0
918 29:18/0 = 0
919 30:18/0 = 0
920 31:18/0 = 0
921 32:18/0 = 0
922 33:18/0 = 0

923 34:18/0 = 0
924 35:18/0 = 0
925 36:18/0 = 0
926 37:18/0 = 0
927 38:18/0 = 0
928 39:18/0 = 0
929 40:18/0 = 0
930 41:18/0 = 0
931 42:18/0 = 0
932 43:18/0 = 0
933 44:18/0 = 0
934 45:18/0 = 0
935 46:18/0 = 0
936 47:18/0 = 0
937 48:18/0 = 0
938 0:19/0 = 0
939 1:19/0 = 0
940 2:19/0 = 0
941 3:19/0 = 0
942 4:19/0 = 0
943 5:19/0 = 0
944 6:19/0 = 0
945 7:19/0 = 0
946 8:19/0 = 0
947 9:19/0 = 0
948 10:19/0 = 0
949 11:19/0 = 0
950 12:19/0 = 0
951 13:19/0 = 0
952 14:19/0 = 0
953 15:19/0 = 0
954 16:19/0 = 0
955 17:19/0 = 0

956 18:19/0 = 0
957 19:19/0 = 0
958 20:19/0 = 0
959 21:19/0 = 0
960 22:19/0 = 0
961 23:19/0 = 0
962 24:19/0 = 0
963 25:19/0 = 0
964 26:19/0 = 0
965 27:19/0 = 0
966 28:19/0 = 0
967 29:19/0 = 0
968 30:19/0 = 0
969 31:19/0 = 0
970 32:19/0 = 0
971 33:19/0 = 0
972 34:19/0 = 0
973 35:19/0 = 0
974 36:19/0 = 0
975 37:19/0 = 0
976 38:19/0 = 0
977 39:19/0 = 0
978 40:19/0 = 0
979 41:19/0 = 0
980 42:19/0 = 0
981 43:19/0 = 0
982 44:19/0 = 0
983 45:19/0 = 0
984 46:19/0 = 0
985 47:19/0 = 0
986 48:19/0 = 0
987 0:20/0 = 0
988 1:20/0 = 0

989 2:20/0 = 0
990 3:20/0 = 0
991 4:20/0 = 0
992 5:20/0 = 0
993 6:20/0 = 0
994 7:20/0 = 0
995 8:20/0 = 0
996 9:20/0 = 0
997 10:20/0 = 0
998 11:20/0 = 0
999 12:20/0 = 0
1000 13:20/0 = 0
1001 14:20/0 = 0
1002 15:20/0 = 0
1003 16:20/0 = 0
1004 17:20/0 = 0
1005 18:20/0 = 0
1006 19:20/0 = 0
1007 20:20/0 = 0
1008 21:20/0 = 0
1009 22:20/0 = 0
1010 23:20/0 = 0
1011 24:20/0 = 0
1012 25:20/0 = 0
1013 26:20/0 = 0
1014 27:20/0 = 0
1015 28:20/0 = 0
1016 29:20/0 = 0
1017 30:20/0 = 0
1018 31:20/0 = 0
1019 32:20/0 = 0
1020 33:20/0 = 0
1021 34:20/0 = 0

1022	$35:20/0 = 0$
1023	$36:20/0 = 0$
1024	$37:20/0 = 0$
1025	$38:20/0 = 0$
1026	$39:20/0 = 0$
1027	$40:20/0 = 0$
1028	$41:20/0 = 0$
1029	$42:20/0 = 0$
1030	$43:20/0 = 0$
1031	$44:20/0 = 0$
1032	$45:20/0 = 0$
1033	$46:20/0 = 0$
1034	$47:20/0 = 0$
1035	$48:20/0 = 0$
1036	$0:21/0 = 0$
1037	$1:21/0 = 0$
1038	$2:21/0 = 0$
1039	$3:21/0 = 0$
1040	$4:21/0 = 0$
1041	$5:21/0 = 0$
1042	$6:21/0 = 0$
1043	$7:21/0 = 0$
1044	$8:21/0 = 0$
1045	$9:21/0 = 0$
1046	$10:21/0 = 0$
1047	$11:21/0 = 0$
1048	$12:21/0 = 0$
1049	$13:21/0 = 0$
1050	$14:21/0 = 0$
1051	$15:21/0 = 0$
1052	$16:21/0 = 0$
1053	$17:21/0 = 0$
1054	$18:21/0 = 0$

```

1055    19:21/0 = 0
1056    20:21/0 = 0
1057    21:21/0 = 0
1058    22:21/0 = 0
1059    23:21/0 = 0
1060    24:21/0 = 0
1061    25:21/0 = 0
1062    26:21/0 = 0
1063    27:21/0 = 0
1064    28:21/0 = 0
1065    29:21/0 = 0
1066    30:21/0 = 0
1067    31:21/0 = 0
1068    32:21/0 = 0
1069    33:21/0 = 0
1070    34:21/0 = 0
1071    35:21/0 = 0
1072    36:21/0 = 0
1073    37:21/0 = 0
1074    38:21/0 = 0
1075    39:21/0 = 0
1076    40:21/0 = 0
1077    41:21/0 = 0
1078    42:21/0 = 0
1079    43:21/0 = 0
1080    44:21/0 = 0
1081    45:21/0 = 0
1082    46:21/0 = 0
1083    47:21/0 = 0
1084    48:21/0 = 0
1085
1086    [resource]
1087    sources/0 = SubResource("TileSetAtlasSource_qer06")

```

C.2.13 LICENSE

```
1   This work is licensed under the Creative Commons Attribution-
    ShareAlike 4.0 International License. To view a copy of this
    license, visit http://creativecommons.org/licenses/by-sa/4.0/
    or send a letter to Creative Commons, PO Box 1866, Mountain
    View, CA 94042, USA.
```

C.3 GD4VoronoiRPG

These are the source code listings for the Voronoi Cells implementation of the 2D tile-map RPG scenario in the Godot game engine. The link to the publicly available source code repository is here: <https://github.com/Zishan-Rahman/GD4VoronoiRPG>

C.3.1 .gitattributes

```
1   # Normalize EOL for all files that Git considers text files.
2   * text=auto eol=lf
```

C.3.2 .gitignore

```
1   # Godot 4+ specific ignores
2   .godot/
```

C.3.3 project.godot

```
1   ; Engine configuration file.
2   ; It's best edited using the editor UI and not directly,
3   ; since the parameters that go here are not all obvious.
4   ;
```

```

5 ; Format:
6 ; [section] ; section goes between []
7 ; param=value ; assign values to parameters
8
9 config_version=5
10
11 [application]
12
13 config/name="Voronoi Cells"
14 run/main_scene="res://tile_map.tscn"
15 config/features=PackedStringArray("4.0", "Forward Plus")
16 config/icon="res://icon.svg"
17
18 [display]
19
20 window/size/viewport_height=640
21
22 [input]
23
24 reset_position={
25 "deadzone": 0.5,
26 "events": [Object(InputEventKey,"resource_local_to_scene":false,"
27             resource_name":"","device":-1,"window_id":0,"alt_pressed":false,
28             "shift_pressed":false,"ctrl_pressed":false,"meta_pressed":
29             false,"pressed":false,"keycode":71,"physical_keycode":0,"
30             key_label":0,"unicode":103,"echo":false,"script":null)
31 , Object(InputEventMouseButton,"resource_local_to_scene":false,"
32           resource_name":"","device":-1,"window_id":0,"alt_pressed":false,
33           "shift_pressed":false,"ctrl_pressed":false,"meta_pressed":
34           false,"button_mask":2,"position":Vector2(75, 12),"
35           global_position":Vector2(78, 44),"factor":1.0,"button_index
36           ":2,"pressed":true,"double_click":false,"script":null)
37 ]

```

```

29     }
30
31     [rendering]
32
33     environment/defaults/default_clear_color=Color(0, 0, 0, 1)

```

C.3.4 tile_map.tscn

```

1     [gd_scene load_steps=7 format=3 uid="uid://d6lxn73sfbh1w"]
2
3     [ext_resource type="Texture2D" uid="uid://cpign73sfbh1w" path="res
        ://monochrome_packed.png" id="1_o183d"]
4     [ext_resource type="Script" path="res://tile_map.gd" id="2_lf4lw"]
5     [ext_resource type="PackedScene" uid="uid://cau5jgogdnf53" path="
        res://accept_dialog.tscn" id="3_y08lj"]
6     [ext_resource type="PackedScene" uid="uid://b5q8ovcigrvyr" path="
        res://win_dialog.tscn" id="4_fkys0"]
7
8     [sub_resource type="TileSetAtlasSource" id="
        TileSetAtlasSource_6h0bd"]
9     texture = ExtResource("1_o183d")
10    0:0/0 = 0
11    1:0/0 = 0
12    2:0/0 = 0
13    3:0/0 = 0
14    4:0/0 = 0
15    5:0/0 = 0
16    6:0/0 = 0
17    7:0/0 = 0
18    8:0/0 = 0
19    9:0/0 = 0
20    10:0/0 = 0

```

21 11:0/0 = 0
22 12:0/0 = 0
23 13:0/0 = 0
24 14:0/0 = 0
25 15:0/0 = 0
26 16:0/0 = 0
27 17:0/0 = 0
28 18:0/0 = 0
29 19:0/0 = 0
30 20:0/0 = 0
31 21:0/0 = 0
32 22:0/0 = 0
33 23:0/0 = 0
34 24:0/0 = 0
35 25:0/0 = 0
36 26:0/0 = 0
37 27:0/0 = 0
38 28:0/0 = 0
39 29:0/0 = 0
40 30:0/0 = 0
41 31:0/0 = 0
42 32:0/0 = 0
43 33:0/0 = 0
44 34:0/0 = 0
45 35:0/0 = 0
46 36:0/0 = 0
47 37:0/0 = 0
48 38:0/0 = 0
49 39:0/0 = 0
50 40:0/0 = 0
51 41:0/0 = 0
52 42:0/0 = 0
53 43:0/0 = 0

54 $44:0/0 = 0$
55 $45:0/0 = 0$
56 $46:0/0 = 0$
57 $47:0/0 = 0$
58 $48:0/0 = 0$
59 $0:1/0 = 0$
60 $1:1/0 = 0$
61 $2:1/0 = 0$
62 $3:1/0 = 0$
63 $4:1/0 = 0$
64 $5:1/0 = 0$
65 $6:1/0 = 0$
66 $7:1/0 = 0$
67 $8:1/0 = 0$
68 $9:1/0 = 0$
69 $10:1/0 = 0$
70 $11:1/0 = 0$
71 $12:1/0 = 0$
72 $13:1/0 = 0$
73 $14:1/0 = 0$
74 $15:1/0 = 0$
75 $16:1/0 = 0$
76 $17:1/0 = 0$
77 $18:1/0 = 0$
78 $19:1/0 = 0$
79 $20:1/0 = 0$
80 $21:1/0 = 0$
81 $22:1/0 = 0$
82 $23:1/0 = 0$
83 $24:1/0 = 0$
84 $25:1/0 = 0$
85 $26:1/0 = 0$
86 $27:1/0 = 0$

87 28:1/0 = 0
88 29:1/0 = 0
89 30:1/0 = 0
90 31:1/0 = 0
91 32:1/0 = 0
92 33:1/0 = 0
93 34:1/0 = 0
94 35:1/0 = 0
95 36:1/0 = 0
96 37:1/0 = 0
97 38:1/0 = 0
98 39:1/0 = 0
99 40:1/0 = 0
100 41:1/0 = 0
101 42:1/0 = 0
102 43:1/0 = 0
103 44:1/0 = 0
104 45:1/0 = 0
105 46:1/0 = 0
106 47:1/0 = 0
107 48:1/0 = 0
108 0:2/0 = 0
109 1:2/0 = 0
110 2:2/0 = 0
111 3:2/0 = 0
112 4:2/0 = 0
113 5:2/0 = 0
114 6:2/0 = 0
115 7:2/0 = 0
116 8:2/0 = 0
117 9:2/0 = 0
118 10:2/0 = 0
119 11:2/0 = 0

120	$12:2/0 = 0$
121	$13:2/0 = 0$
122	$14:2/0 = 0$
123	$15:2/0 = 0$
124	$16:2/0 = 0$
125	$17:2/0 = 0$
126	$18:2/0 = 0$
127	$19:2/0 = 0$
128	$20:2/0 = 0$
129	$21:2/0 = 0$
130	$22:2/0 = 0$
131	$23:2/0 = 0$
132	$24:2/0 = 0$
133	$25:2/0 = 0$
134	$26:2/0 = 0$
135	$27:2/0 = 0$
136	$28:2/0 = 0$
137	$29:2/0 = 0$
138	$30:2/0 = 0$
139	$31:2/0 = 0$
140	$32:2/0 = 0$
141	$33:2/0 = 0$
142	$34:2/0 = 0$
143	$35:2/0 = 0$
144	$36:2/0 = 0$
145	$37:2/0 = 0$
146	$38:2/0 = 0$
147	$39:2/0 = 0$
148	$40:2/0 = 0$
149	$41:2/0 = 0$
150	$42:2/0 = 0$
151	$43:2/0 = 0$
152	$44:2/0 = 0$

153	$45:2/0 = 0$
154	$46:2/0 = 0$
155	$47:2/0 = 0$
156	$48:2/0 = 0$
157	$0:3/0 = 0$
158	$1:3/0 = 0$
159	$2:3/0 = 0$
160	$3:3/0 = 0$
161	$4:3/0 = 0$
162	$5:3/0 = 0$
163	$6:3/0 = 0$
164	$7:3/0 = 0$
165	$8:3/0 = 0$
166	$9:3/0 = 0$
167	$10:3/0 = 0$
168	$11:3/0 = 0$
169	$12:3/0 = 0$
170	$13:3/0 = 0$
171	$14:3/0 = 0$
172	$15:3/0 = 0$
173	$16:3/0 = 0$
174	$17:3/0 = 0$
175	$18:3/0 = 0$
176	$19:3/0 = 0$
177	$20:3/0 = 0$
178	$21:3/0 = 0$
179	$22:3/0 = 0$
180	$23:3/0 = 0$
181	$24:3/0 = 0$
182	$25:3/0 = 0$
183	$26:3/0 = 0$
184	$27:3/0 = 0$
185	$28:3/0 = 0$

186 29:3/0 = 0
187 30:3/0 = 0
188 31:3/0 = 0
189 32:3/0 = 0
190 33:3/0 = 0
191 34:3/0 = 0
192 35:3/0 = 0
193 36:3/0 = 0
194 37:3/0 = 0
195 38:3/0 = 0
196 39:3/0 = 0
197 40:3/0 = 0
198 41:3/0 = 0
199 42:3/0 = 0
200 43:3/0 = 0
201 44:3/0 = 0
202 45:3/0 = 0
203 46:3/0 = 0
204 47:3/0 = 0
205 48:3/0 = 0
206 0:4/0 = 0
207 1:4/0 = 0
208 2:4/0 = 0
209 3:4/0 = 0
210 4:4/0 = 0
211 5:4/0 = 0
212 6:4/0 = 0
213 7:4/0 = 0
214 8:4/0 = 0
215 9:4/0 = 0
216 10:4/0 = 0
217 11:4/0 = 0
218 12:4/0 = 0

219	$13:4/0 = 0$
220	$14:4/0 = 0$
221	$15:4/0 = 0$
222	$16:4/0 = 0$
223	$17:4/0 = 0$
224	$18:4/0 = 0$
225	$19:4/0 = 0$
226	$20:4/0 = 0$
227	$21:4/0 = 0$
228	$22:4/0 = 0$
229	$23:4/0 = 0$
230	$24:4/0 = 0$
231	$25:4/0 = 0$
232	$26:4/0 = 0$
233	$27:4/0 = 0$
234	$28:4/0 = 0$
235	$29:4/0 = 0$
236	$30:4/0 = 0$
237	$31:4/0 = 0$
238	$32:4/0 = 0$
239	$33:4/0 = 0$
240	$34:4/0 = 0$
241	$35:4/0 = 0$
242	$36:4/0 = 0$
243	$37:4/0 = 0$
244	$38:4/0 = 0$
245	$39:4/0 = 0$
246	$40:4/0 = 0$
247	$41:4/0 = 0$
248	$42:4/0 = 0$
249	$43:4/0 = 0$
250	$44:4/0 = 0$
251	$45:4/0 = 0$

252	$46:4/0 = 0$
253	$47:4/0 = 0$
254	$48:4/0 = 0$
255	$0:5/0 = 0$
256	$1:5/0 = 0$
257	$2:5/0 = 0$
258	$3:5/0 = 0$
259	$4:5/0 = 0$
260	$5:5/0 = 0$
261	$6:5/0 = 0$
262	$7:5/0 = 0$
263	$8:5/0 = 0$
264	$9:5/0 = 0$
265	$10:5/0 = 0$
266	$11:5/0 = 0$
267	$12:5/0 = 0$
268	$13:5/0 = 0$
269	$14:5/0 = 0$
270	$15:5/0 = 0$
271	$16:5/0 = 0$
272	$17:5/0 = 0$
273	$18:5/0 = 0$
274	$19:5/0 = 0$
275	$20:5/0 = 0$
276	$21:5/0 = 0$
277	$22:5/0 = 0$
278	$23:5/0 = 0$
279	$24:5/0 = 0$
280	$25:5/0 = 0$
281	$26:5/0 = 0$
282	$27:5/0 = 0$
283	$28:5/0 = 0$
284	$29:5/0 = 0$

285	$30:5/0 = 0$
286	$31:5/0 = 0$
287	$32:5/0 = 0$
288	$33:5/0 = 0$
289	$34:5/0 = 0$
290	$35:5/0 = 0$
291	$36:5/0 = 0$
292	$37:5/0 = 0$
293	$38:5/0 = 0$
294	$39:5/0 = 0$
295	$40:5/0 = 0$
296	$41:5/0 = 0$
297	$42:5/0 = 0$
298	$43:5/0 = 0$
299	$44:5/0 = 0$
300	$45:5/0 = 0$
301	$46:5/0 = 0$
302	$47:5/0 = 0$
303	$48:5/0 = 0$
304	$0:6/0 = 0$
305	$1:6/0 = 0$
306	$2:6/0 = 0$
307	$3:6/0 = 0$
308	$4:6/0 = 0$
309	$5:6/0 = 0$
310	$6:6/0 = 0$
311	$7:6/0 = 0$
312	$8:6/0 = 0$
313	$9:6/0 = 0$
314	$10:6/0 = 0$
315	$11:6/0 = 0$
316	$12:6/0 = 0$
317	$13:6/0 = 0$

318 14:6/0 = 0
319 15:6/0 = 0
320 16:6/0 = 0
321 17:6/0 = 0
322 18:6/0 = 0
323 19:6/0 = 0
324 20:6/0 = 0
325 21:6/0 = 0
326 22:6/0 = 0
327 23:6/0 = 0
328 24:6/0 = 0
329 25:6/0 = 0
330 26:6/0 = 0
331 27:6/0 = 0
332 28:6/0 = 0
333 29:6/0 = 0
334 30:6/0 = 0
335 31:6/0 = 0
336 32:6/0 = 0
337 33:6/0 = 0
338 34:6/0 = 0
339 35:6/0 = 0
340 36:6/0 = 0
341 37:6/0 = 0
342 38:6/0 = 0
343 39:6/0 = 0
344 40:6/0 = 0
345 41:6/0 = 0
346 42:6/0 = 0
347 43:6/0 = 0
348 44:6/0 = 0
349 45:6/0 = 0
350 46:6/0 = 0

351 47:6/0 = 0
352 48:6/0 = 0
353 0:7/0 = 0
354 1:7/0 = 0
355 2:7/0 = 0
356 3:7/0 = 0
357 4:7/0 = 0
358 5:7/0 = 0
359 6:7/0 = 0
360 7:7/0 = 0
361 8:7/0 = 0
362 9:7/0 = 0
363 10:7/0 = 0
364 11:7/0 = 0
365 12:7/0 = 0
366 13:7/0 = 0
367 14:7/0 = 0
368 15:7/0 = 0
369 16:7/0 = 0
370 17:7/0 = 0
371 18:7/0 = 0
372 19:7/0 = 0
373 20:7/0 = 0
374 21:7/0 = 0
375 22:7/0 = 0
376 23:7/0 = 0
377 24:7/0 = 0
378 25:7/0 = 0
379 26:7/0 = 0
380 27:7/0 = 0
381 28:7/0 = 0
382 29:7/0 = 0
383 30:7/0 = 0

384 31:7/0 = 0
385 32:7/0 = 0
386 33:7/0 = 0
387 34:7/0 = 0
388 35:7/0 = 0
389 36:7/0 = 0
390 37:7/0 = 0
391 38:7/0 = 0
392 39:7/0 = 0
393 40:7/0 = 0
394 41:7/0 = 0
395 42:7/0 = 0
396 43:7/0 = 0
397 44:7/0 = 0
398 45:7/0 = 0
399 46:7/0 = 0
400 47:7/0 = 0
401 48:7/0 = 0
402 0:8/0 = 0
403 1:8/0 = 0
404 2:8/0 = 0
405 3:8/0 = 0
406 4:8/0 = 0
407 5:8/0 = 0
408 6:8/0 = 0
409 7:8/0 = 0
410 8:8/0 = 0
411 9:8/0 = 0
412 10:8/0 = 0
413 11:8/0 = 0
414 12:8/0 = 0
415 13:8/0 = 0
416 14:8/0 = 0

417 15:8/0 = 0
418 16:8/0 = 0
419 17:8/0 = 0
420 18:8/0 = 0
421 19:8/0 = 0
422 20:8/0 = 0
423 21:8/0 = 0
424 22:8/0 = 0
425 23:8/0 = 0
426 24:8/0 = 0
427 25:8/0 = 0
428 26:8/0 = 0
429 27:8/0 = 0
430 28:8/0 = 0
431 29:8/0 = 0
432 30:8/0 = 0
433 31:8/0 = 0
434 32:8/0 = 0
435 33:8/0 = 0
436 34:8/0 = 0
437 35:8/0 = 0
438 36:8/0 = 0
439 37:8/0 = 0
440 38:8/0 = 0
441 39:8/0 = 0
442 40:8/0 = 0
443 41:8/0 = 0
444 42:8/0 = 0
445 43:8/0 = 0
446 44:8/0 = 0
447 45:8/0 = 0
448 46:8/0 = 0
449 47:8/0 = 0

450	$48:8/0 = 0$
451	$0:9/0 = 0$
452	$1:9/0 = 0$
453	$2:9/0 = 0$
454	$3:9/0 = 0$
455	$4:9/0 = 0$
456	$5:9/0 = 0$
457	$6:9/0 = 0$
458	$7:9/0 = 0$
459	$8:9/0 = 0$
460	$9:9/0 = 0$
461	$10:9/0 = 0$
462	$11:9/0 = 0$
463	$12:9/0 = 0$
464	$13:9/0 = 0$
465	$14:9/0 = 0$
466	$15:9/0 = 0$
467	$16:9/0 = 0$
468	$17:9/0 = 0$
469	$18:9/0 = 0$
470	$19:9/0 = 0$
471	$20:9/0 = 0$
472	$21:9/0 = 0$
473	$22:9/0 = 0$
474	$23:9/0 = 0$
475	$24:9/0 = 0$
476	$25:9/0 = 0$
477	$26:9/0 = 0$
478	$27:9/0 = 0$
479	$28:9/0 = 0$
480	$29:9/0 = 0$
481	$30:9/0 = 0$
482	$31:9/0 = 0$

483 32:9/0 = 0
484 33:9/0 = 0
485 34:9/0 = 0
486 35:9/0 = 0
487 36:9/0 = 0
488 37:9/0 = 0
489 38:9/0 = 0
490 39:9/0 = 0
491 40:9/0 = 0
492 41:9/0 = 0
493 42:9/0 = 0
494 43:9/0 = 0
495 44:9/0 = 0
496 45:9/0 = 0
497 46:9/0 = 0
498 47:9/0 = 0
499 48:9/0 = 0
500 0:10/0 = 0
501 1:10/0 = 0
502 2:10/0 = 0
503 3:10/0 = 0
504 4:10/0 = 0
505 5:10/0 = 0
506 6:10/0 = 0
507 7:10/0 = 0
508 8:10/0 = 0
509 9:10/0 = 0
510 10:10/0 = 0
511 11:10/0 = 0
512 12:10/0 = 0
513 13:10/0 = 0
514 14:10/0 = 0
515 15:10/0 = 0

516 16:10/0 = 0
517 17:10/0 = 0
518 18:10/0 = 0
519 19:10/0 = 0
520 20:10/0 = 0
521 21:10/0 = 0
522 22:10/0 = 0
523 23:10/0 = 0
524 24:10/0 = 0
525 25:10/0 = 0
526 26:10/0 = 0
527 27:10/0 = 0
528 28:10/0 = 0
529 29:10/0 = 0
530 30:10/0 = 0
531 31:10/0 = 0
532 32:10/0 = 0
533 33:10/0 = 0
534 34:10/0 = 0
535 35:10/0 = 0
536 36:10/0 = 0
537 37:10/0 = 0
538 38:10/0 = 0
539 39:10/0 = 0
540 40:10/0 = 0
541 41:10/0 = 0
542 42:10/0 = 0
543 43:10/0 = 0
544 44:10/0 = 0
545 45:10/0 = 0
546 46:10/0 = 0
547 47:10/0 = 0
548 48:10/0 = 0

549 0:11/0 = 0
550 1:11/0 = 0
551 2:11/0 = 0
552 3:11/0 = 0
553 4:11/0 = 0
554 5:11/0 = 0
555 6:11/0 = 0
556 7:11/0 = 0
557 8:11/0 = 0
558 9:11/0 = 0
559 10:11/0 = 0
560 11:11/0 = 0
561 12:11/0 = 0
562 13:11/0 = 0
563 14:11/0 = 0
564 15:11/0 = 0
565 16:11/0 = 0
566 17:11/0 = 0
567 18:11/0 = 0
568 19:11/0 = 0
569 20:11/0 = 0
570 21:11/0 = 0
571 22:11/0 = 0
572 23:11/0 = 0
573 24:11/0 = 0
574 25:11/0 = 0
575 26:11/0 = 0
576 27:11/0 = 0
577 28:11/0 = 0
578 29:11/0 = 0
579 30:11/0 = 0
580 31:11/0 = 0
581 32:11/0 = 0

582 33:11/0 = 0
583 34:11/0 = 0
584 35:11/0 = 0
585 36:11/0 = 0
586 37:11/0 = 0
587 38:11/0 = 0
588 39:11/0 = 0
589 40:11/0 = 0
590 41:11/0 = 0
591 42:11/0 = 0
592 43:11/0 = 0
593 44:11/0 = 0
594 45:11/0 = 0
595 46:11/0 = 0
596 47:11/0 = 0
597 48:11/0 = 0
598 0:12/0 = 0
599 1:12/0 = 0
600 2:12/0 = 0
601 3:12/0 = 0
602 4:12/0 = 0
603 5:12/0 = 0
604 6:12/0 = 0
605 7:12/0 = 0
606 8:12/0 = 0
607 9:12/0 = 0
608 10:12/0 = 0
609 11:12/0 = 0
610 12:12/0 = 0
611 13:12/0 = 0
612 14:12/0 = 0
613 15:12/0 = 0
614 16:12/0 = 0

615 17:12/0 = 0
616 18:12/0 = 0
617 19:12/0 = 0
618 20:12/0 = 0
619 21:12/0 = 0
620 22:12/0 = 0
621 23:12/0 = 0
622 24:12/0 = 0
623 25:12/0 = 0
624 26:12/0 = 0
625 27:12/0 = 0
626 28:12/0 = 0
627 29:12/0 = 0
628 30:12/0 = 0
629 31:12/0 = 0
630 32:12/0 = 0
631 33:12/0 = 0
632 34:12/0 = 0
633 35:12/0 = 0
634 36:12/0 = 0
635 37:12/0 = 0
636 38:12/0 = 0
637 39:12/0 = 0
638 40:12/0 = 0
639 41:12/0 = 0
640 42:12/0 = 0
641 43:12/0 = 0
642 44:12/0 = 0
643 45:12/0 = 0
644 46:12/0 = 0
645 47:12/0 = 0
646 48:12/0 = 0
647 0:13/0 = 0

648 1:13/0 = 0
649 2:13/0 = 0
650 3:13/0 = 0
651 4:13/0 = 0
652 5:13/0 = 0
653 6:13/0 = 0
654 7:13/0 = 0
655 8:13/0 = 0
656 9:13/0 = 0
657 10:13/0 = 0
658 11:13/0 = 0
659 12:13/0 = 0
660 13:13/0 = 0
661 14:13/0 = 0
662 15:13/0 = 0
663 16:13/0 = 0
664 17:13/0 = 0
665 18:13/0 = 0
666 19:13/0 = 0
667 20:13/0 = 0
668 21:13/0 = 0
669 22:13/0 = 0
670 23:13/0 = 0
671 24:13/0 = 0
672 25:13/0 = 0
673 26:13/0 = 0
674 27:13/0 = 0
675 28:13/0 = 0
676 29:13/0 = 0
677 30:13/0 = 0
678 31:13/0 = 0
679 32:13/0 = 0
680 33:13/0 = 0

681 34:13/0 = 0
682 35:13/0 = 0
683 36:13/0 = 0
684 37:13/0 = 0
685 38:13/0 = 0
686 39:13/0 = 0
687 40:13/0 = 0
688 41:13/0 = 0
689 42:13/0 = 0
690 43:13/0 = 0
691 44:13/0 = 0
692 45:13/0 = 0
693 46:13/0 = 0
694 47:13/0 = 0
695 48:13/0 = 0
696 0:14/0 = 0
697 1:14/0 = 0
698 2:14/0 = 0
699 3:14/0 = 0
700 4:14/0 = 0
701 5:14/0 = 0
702 6:14/0 = 0
703 7:14/0 = 0
704 8:14/0 = 0
705 9:14/0 = 0
706 10:14/0 = 0
707 11:14/0 = 0
708 12:14/0 = 0
709 13:14/0 = 0
710 14:14/0 = 0
711 15:14/0 = 0
712 16:14/0 = 0
713 17:14/0 = 0

714	$18:14/0 = 0$
715	$19:14/0 = 0$
716	$20:14/0 = 0$
717	$21:14/0 = 0$
718	$22:14/0 = 0$
719	$23:14/0 = 0$
720	$24:14/0 = 0$
721	$25:14/0 = 0$
722	$26:14/0 = 0$
723	$27:14/0 = 0$
724	$28:14/0 = 0$
725	$29:14/0 = 0$
726	$30:14/0 = 0$
727	$31:14/0 = 0$
728	$32:14/0 = 0$
729	$33:14/0 = 0$
730	$34:14/0 = 0$
731	$35:14/0 = 0$
732	$36:14/0 = 0$
733	$37:14/0 = 0$
734	$38:14/0 = 0$
735	$39:14/0 = 0$
736	$40:14/0 = 0$
737	$41:14/0 = 0$
738	$42:14/0 = 0$
739	$43:14/0 = 0$
740	$44:14/0 = 0$
741	$45:14/0 = 0$
742	$46:14/0 = 0$
743	$47:14/0 = 0$
744	$48:14/0 = 0$
745	$0:15/0 = 0$
746	$1:15/0 = 0$

747 2:15/0 = 0
748 3:15/0 = 0
749 4:15/0 = 0
750 5:15/0 = 0
751 6:15/0 = 0
752 7:15/0 = 0
753 8:15/0 = 0
754 9:15/0 = 0
755 10:15/0 = 0
756 11:15/0 = 0
757 12:15/0 = 0
758 13:15/0 = 0
759 14:15/0 = 0
760 15:15/0 = 0
761 16:15/0 = 0
762 17:15/0 = 0
763 18:15/0 = 0
764 19:15/0 = 0
765 20:15/0 = 0
766 21:15/0 = 0
767 22:15/0 = 0
768 23:15/0 = 0
769 24:15/0 = 0
770 25:15/0 = 0
771 26:15/0 = 0
772 27:15/0 = 0
773 28:15/0 = 0
774 29:15/0 = 0
775 30:15/0 = 0
776 31:15/0 = 0
777 32:15/0 = 0
778 33:15/0 = 0
779 34:15/0 = 0

780	$35:15/0 = 0$
781	$36:15/0 = 0$
782	$37:15/0 = 0$
783	$38:15/0 = 0$
784	$39:15/0 = 0$
785	$40:15/0 = 0$
786	$41:15/0 = 0$
787	$42:15/0 = 0$
788	$43:15/0 = 0$
789	$44:15/0 = 0$
790	$45:15/0 = 0$
791	$46:15/0 = 0$
792	$47:15/0 = 0$
793	$48:15/0 = 0$
794	$0:16/0 = 0$
795	$1:16/0 = 0$
796	$2:16/0 = 0$
797	$3:16/0 = 0$
798	$4:16/0 = 0$
799	$5:16/0 = 0$
800	$6:16/0 = 0$
801	$7:16/0 = 0$
802	$8:16/0 = 0$
803	$9:16/0 = 0$
804	$10:16/0 = 0$
805	$11:16/0 = 0$
806	$12:16/0 = 0$
807	$13:16/0 = 0$
808	$14:16/0 = 0$
809	$15:16/0 = 0$
810	$16:16/0 = 0$
811	$17:16/0 = 0$
812	$18:16/0 = 0$

813 19:16/0 = 0
814 20:16/0 = 0
815 21:16/0 = 0
816 22:16/0 = 0
817 23:16/0 = 0
818 24:16/0 = 0
819 25:16/0 = 0
820 26:16/0 = 0
821 27:16/0 = 0
822 28:16/0 = 0
823 29:16/0 = 0
824 30:16/0 = 0
825 31:16/0 = 0
826 32:16/0 = 0
827 33:16/0 = 0
828 34:16/0 = 0
829 35:16/0 = 0
830 36:16/0 = 0
831 37:16/0 = 0
832 38:16/0 = 0
833 39:16/0 = 0
834 40:16/0 = 0
835 41:16/0 = 0
836 42:16/0 = 0
837 43:16/0 = 0
838 44:16/0 = 0
839 45:16/0 = 0
840 46:16/0 = 0
841 47:16/0 = 0
842 48:16/0 = 0
843 0:17/0 = 0
844 1:17/0 = 0
845 2:17/0 = 0

846 3:17/0 = 0
847 4:17/0 = 0
848 5:17/0 = 0
849 6:17/0 = 0
850 7:17/0 = 0
851 8:17/0 = 0
852 9:17/0 = 0
853 10:17/0 = 0
854 11:17/0 = 0
855 12:17/0 = 0
856 13:17/0 = 0
857 14:17/0 = 0
858 15:17/0 = 0
859 16:17/0 = 0
860 17:17/0 = 0
861 18:17/0 = 0
862 19:17/0 = 0
863 20:17/0 = 0
864 21:17/0 = 0
865 22:17/0 = 0
866 23:17/0 = 0
867 24:17/0 = 0
868 25:17/0 = 0
869 26:17/0 = 0
870 27:17/0 = 0
871 28:17/0 = 0
872 29:17/0 = 0
873 30:17/0 = 0
874 31:17/0 = 0
875 32:17/0 = 0
876 33:17/0 = 0
877 34:17/0 = 0
878 35:17/0 = 0

879 36:17/0 = 0
880 37:17/0 = 0
881 38:17/0 = 0
882 39:17/0 = 0
883 40:17/0 = 0
884 41:17/0 = 0
885 42:17/0 = 0
886 43:17/0 = 0
887 44:17/0 = 0
888 45:17/0 = 0
889 46:17/0 = 0
890 47:17/0 = 0
891 48:17/0 = 0
892 0:18/0 = 0
893 1:18/0 = 0
894 2:18/0 = 0
895 3:18/0 = 0
896 4:18/0 = 0
897 5:18/0 = 0
898 6:18/0 = 0
899 7:18/0 = 0
900 8:18/0 = 0
901 9:18/0 = 0
902 10:18/0 = 0
903 11:18/0 = 0
904 12:18/0 = 0
905 13:18/0 = 0
906 14:18/0 = 0
907 15:18/0 = 0
908 16:18/0 = 0
909 17:18/0 = 0
910 18:18/0 = 0
911 19:18/0 = 0

912 20:18/0 = 0
913 21:18/0 = 0
914 22:18/0 = 0
915 23:18/0 = 0
916 24:18/0 = 0
917 25:18/0 = 0
918 26:18/0 = 0
919 27:18/0 = 0
920 28:18/0 = 0
921 29:18/0 = 0
922 30:18/0 = 0
923 31:18/0 = 0
924 32:18/0 = 0
925 33:18/0 = 0
926 34:18/0 = 0
927 35:18/0 = 0
928 36:18/0 = 0
929 37:18/0 = 0
930 38:18/0 = 0
931 39:18/0 = 0
932 40:18/0 = 0
933 41:18/0 = 0
934 42:18/0 = 0
935 43:18/0 = 0
936 44:18/0 = 0
937 45:18/0 = 0
938 46:18/0 = 0
939 47:18/0 = 0
940 48:18/0 = 0
941 0:19/0 = 0
942 1:19/0 = 0
943 2:19/0 = 0
944 3:19/0 = 0

945 4:19/0 = 0
946 5:19/0 = 0
947 6:19/0 = 0
948 7:19/0 = 0
949 8:19/0 = 0
950 9:19/0 = 0
951 10:19/0 = 0
952 11:19/0 = 0
953 12:19/0 = 0
954 13:19/0 = 0
955 14:19/0 = 0
956 15:19/0 = 0
957 16:19/0 = 0
958 17:19/0 = 0
959 18:19/0 = 0
960 19:19/0 = 0
961 20:19/0 = 0
962 21:19/0 = 0
963 22:19/0 = 0
964 23:19/0 = 0
965 24:19/0 = 0
966 25:19/0 = 0
967 26:19/0 = 0
968 27:19/0 = 0
969 28:19/0 = 0
970 29:19/0 = 0
971 30:19/0 = 0
972 31:19/0 = 0
973 32:19/0 = 0
974 33:19/0 = 0
975 34:19/0 = 0
976 35:19/0 = 0
977 36:19/0 = 0

978	$37:19/0 = 0$
979	$38:19/0 = 0$
980	$39:19/0 = 0$
981	$40:19/0 = 0$
982	$41:19/0 = 0$
983	$42:19/0 = 0$
984	$43:19/0 = 0$
985	$44:19/0 = 0$
986	$45:19/0 = 0$
987	$46:19/0 = 0$
988	$47:19/0 = 0$
989	$48:19/0 = 0$
990	$0:20/0 = 0$
991	$1:20/0 = 0$
992	$2:20/0 = 0$
993	$3:20/0 = 0$
994	$4:20/0 = 0$
995	$5:20/0 = 0$
996	$6:20/0 = 0$
997	$7:20/0 = 0$
998	$8:20/0 = 0$
999	$9:20/0 = 0$
1000	$10:20/0 = 0$
1001	$11:20/0 = 0$
1002	$12:20/0 = 0$
1003	$13:20/0 = 0$
1004	$14:20/0 = 0$
1005	$15:20/0 = 0$
1006	$16:20/0 = 0$
1007	$17:20/0 = 0$
1008	$18:20/0 = 0$
1009	$19:20/0 = 0$
1010	$20:20/0 = 0$

1011	$21:20/0 = 0$
1012	$22:20/0 = 0$
1013	$23:20/0 = 0$
1014	$24:20/0 = 0$
1015	$25:20/0 = 0$
1016	$26:20/0 = 0$
1017	$27:20/0 = 0$
1018	$28:20/0 = 0$
1019	$29:20/0 = 0$
1020	$30:20/0 = 0$
1021	$31:20/0 = 0$
1022	$32:20/0 = 0$
1023	$33:20/0 = 0$
1024	$34:20/0 = 0$
1025	$35:20/0 = 0$
1026	$36:20/0 = 0$
1027	$37:20/0 = 0$
1028	$38:20/0 = 0$
1029	$39:20/0 = 0$
1030	$40:20/0 = 0$
1031	$41:20/0 = 0$
1032	$42:20/0 = 0$
1033	$43:20/0 = 0$
1034	$44:20/0 = 0$
1035	$45:20/0 = 0$
1036	$46:20/0 = 0$
1037	$47:20/0 = 0$
1038	$48:20/0 = 0$
1039	$0:21/0 = 0$
1040	$1:21/0 = 0$
1041	$2:21/0 = 0$
1042	$3:21/0 = 0$
1043	$4:21/0 = 0$

1044	$5:21/0 = 0$
1045	$6:21/0 = 0$
1046	$7:21/0 = 0$
1047	$8:21/0 = 0$
1048	$9:21/0 = 0$
1049	$10:21/0 = 0$
1050	$11:21/0 = 0$
1051	$12:21/0 = 0$
1052	$13:21/0 = 0$
1053	$14:21/0 = 0$
1054	$15:21/0 = 0$
1055	$16:21/0 = 0$
1056	$17:21/0 = 0$
1057	$18:21/0 = 0$
1058	$19:21/0 = 0$
1059	$20:21/0 = 0$
1060	$21:21/0 = 0$
1061	$22:21/0 = 0$
1062	$23:21/0 = 0$
1063	$24:21/0 = 0$
1064	$25:21/0 = 0$
1065	$26:21/0 = 0$
1066	$27:21/0 = 0$
1067	$28:21/0 = 0$
1068	$29:21/0 = 0$
1069	$30:21/0 = 0$
1070	$31:21/0 = 0$
1071	$32:21/0 = 0$
1072	$33:21/0 = 0$
1073	$34:21/0 = 0$
1074	$35:21/0 = 0$
1075	$36:21/0 = 0$
1076	$37:21/0 = 0$

```

1077 38:21/0 = 0
1078 39:21/0 = 0
1079 40:21/0 = 0
1080 41:21/0 = 0
1081 42:21/0 = 0
1082 43:21/0 = 0
1083 44:21/0 = 0
1084 45:21/0 = 0
1085 46:21/0 = 0
1086 47:21/0 = 0
1087 48:21/0 = 0
1088
1089 [sub_resource type="TileSet" id="TileSet_3drs5"]
1090 sources/0 = SubResource("TileSetAtlasSource_6h0bd")
1091
1092 [node name="TileMap" type="TileMap"]
1093 tile_set = SubResource("TileSet_3drs5")
1094 format = 2
1095 script = ExtResource("2_lf4lw")
1096
1097 [node name="AcceptDialog" parent="." instance=ExtResource("3_y08lj"
    )]
1098
1099 [node name="WinDialog" parent="." instance=ExtResource("4_fkys0")]

```

C.3.5 tile_map.gd

```

1 extends TileMap
2
3 const buildings: Array[Vector2i] = [
4     Vector2i(0, 19),
5     Vector2i(1, 19),

```



```

6      Vector2i(2, 19),
7      Vector2i(3, 19),
8      Vector2i(4, 19),
9      Vector2i(5, 19),
10     Vector2i(6, 19),
11     Vector2i(7, 19),
12     Vector2i(8, 20),
13     Vector2i(0, 20),
14     Vector2i(1, 20),
15     Vector2i(2, 20),
16     Vector2i(3, 20),
17     Vector2i(4, 20),
18     Vector2i(5, 20),
19     Vector2i(6, 20),
20     Vector2i(7, 20),
21     Vector2i(8, 20),
22     Vector2i(0, 21),
23     Vector2i(1, 21),
24     Vector2i(2, 21),
25     Vector2i(3, 21),
26     Vector2i(4, 21),
27     Vector2i(5, 21),
28     Vector2i(6, 21),
29     Vector2i(7, 21),
30     Vector2i(8, 21)
31 ]
32 const trees: Array[Vector2i] = [
33     Vector2i(0,1),
34     Vector2i(1,1),
35     Vector2i(2,1),
36     Vector2i(3,1),
37     Vector2i(4,1),
38     Vector2i(5,1),

```

```

39     Vector2i(6,1),
40     Vector2i(7,1),
41     Vector2i(0,2),
42     Vector2i(1,2),
43     Vector2i(2,2),
44     Vector2i(3,2),
45     Vector2i(4,2)
46 ]
47 const PLAYER_SPRITE: Vector2i = Vector2i(24, 7)
48 var player_placement_cell: Vector2i
49 const rings: Array[Vector2i] = [
50     Vector2i(43, 6),
51     Vector2i(44, 6),
52     Vector2i(45, 6),
53     Vector2i(46, 6)
54 ]
55 var ring_placement_cell: Vector2i
56
57 var points: Array[Dictionary] = []
58 const EUCLIDEAN: String = "Euclidean distance"
59 const MANHATTAN: String = "Manhattan distance"
60 ## Determines whether or not the Euclidean or Manhattan distance
    formula is used for calculation of the deltas between points
    within Voronoi cells.
61 @export_enum(EUCLIDEAN, MANHATTAN) var distance: String = MANHATTAN
62 ## Determines the number of points randomly picked from at the
    start. Therefore, it also determines the number of cells in our
    Voronoi tessellation.
63 @export_range(15, 40, 1) var random_starting_points: int = 20
64 var x_tile_range: int = ProjectSettings.get_setting("display/window
    /size/viewport_width") / tile_set.tile_size.x
65 var y_tile_range: int = ProjectSettings.get_setting("display/window
    /size/viewport_height") / tile_set.tile_size.y

```

```

66
67 # Called when the node enters the scene tree for the first time.
68 func _ready() -> void:
69     randomize()
70     var start_time: float = Time.get_ticks_msec()
71     define_points(random_starting_points)
72     paint_points()
73     place_player()
74     place_ring()
75     var new_time: float = Time.get_ticks_msec() - start_time
76     print("Time taken: " + str(new_time) + "ms")
77     $AcceptDialog.dialog_text = "You're a hollow Golem who seeks the
        ultimate treasure; a ring that's got something on top of it
        . It's somewhere in this large village and barely visible to
        your naked eyes, which took us " + str(new_time) + "
        milliseconds to generate (" + str(new_time / 1000.0) + "
        seconds), but you'll stop at nothing to get what you want.
        You can chow down every tree and fauna that stands in your
        way of the ring, but your Achilles heel is any bricks and
        mortar, which WILL make you stop at your tracks. Since it's
        easy to get lost in here, we'll tell you that you're in
        position " + str(player_placement_cell) + " in this big
        village of size " + str(Vector2i(x_tile_range, y_tile_range)
        ) + ". It's also easy to get stuck here, so either press the
        G key or right click to teleport somewhere else where there
        is fauna (or even the ring!!), because this game actually
        WANTS you to win it. Ultimately, though, it is YOUR job to
        find the ring, so are you ready to attain the treasure that
        is rightfully yours?!"
78     $AcceptDialog.visible = true
79     $AcceptDialog.confirmed.connect(_on_AcceptDialog_closed)
80     $AcceptDialog.canceled.connect(_on_AcceptDialog_closed)
81     $WinDialog.confirmed.connect(_on_WinDialog_confirmed)

```

```

82     $WinDialog.canceled.connect(_on_WinDialog_canceled)
83     get_tree().paused = true
84
85     func _on_WinDialog_confirmed() -> void:
86         get_tree().reload_current_scene()
87
88     func _on_WinDialog_canceled() -> void:
89         get_tree().quit()
90
91     func _on_AcceptDialog_closed() -> void:
92         $AcceptDialog.visible = false
93         get_tree().paused = false
94
95     func _get_random_placement_cell() -> Vector2i:
96         return Vector2i(randi() % x_tile_range, randi() % y_tile_range)
97
98     func place_player() -> void:
99         player_placement_cell = _get_random_placement_cell()
100         while buildings.has(get_cell_atlas_coords(0,
101             player_placement_cell)) or player_placement_cell ==
102             ring_placement_cell:
103             player_placement_cell = _get_random_placement_cell()
104         set_cell(0, player_placement_cell, 0, PLAYER_SPRITE)
105
106     func place_ring() -> void:
107         ring_placement_cell = _get_random_placement_cell()
108         while buildings.has(get_cell_atlas_coords(0, ring_placement_cell
109             )) or ring_placement_cell == player_placement_cell:
110             ring_placement_cell = _get_random_placement_cell()
111         set_cell(0, ring_placement_cell, 0, rings.pick_random())
112
113     func _is_not_out_of_bounds(cell: Vector2i) -> bool:
114         return cell.x >= 0 and cell.x < x_tile_range and cell.y >= 0 and

```

```

        cell.y < y_tile_range
112
113 func _physics_process(_delta) -> void:
114     var previous_cell: Vector2i = player_placement_cell
115     var direction: Vector2i = Vector2i.ZERO
116     if Input.is_action_pressed("ui_up"): direction = Vector2i.UP
117     elif Input.is_action_pressed("ui_down"): direction = Vector2i.
        DOWN
118     elif Input.is_action_pressed("ui_left"): direction = Vector2i.
        LEFT
119     elif Input.is_action_pressed("ui_right"): direction = Vector2i.
        RIGHT
120     elif Input.is_action_just_pressed("reset_position"): # Respawn
        player in a different part of the map
121     player_placement_cell = _get_random_placement_cell()
122     while buildings.has(get_cell_atlas_coords(0,
        player_placement_cell)): # This time, since we're not
        STARTING the game, we don't care whether or not the
        player magically lands on the ring
123     player_placement_cell = _get_random_placement_cell()
124     set_cell(0, player_placement_cell, 0, PLAYER_SPRITE)
125     set_cell(0, previous_cell, 0) # replace the previous sprite
126     _win_game_if_won()
127     return
128     var new_placement_cell: Vector2i = player_placement_cell +
        direction
129     if (not get_used_cells(0).has(new_placement_cell) or trees.has(
        get_cell_atlas_coords(0, new_placement_cell)) or
        new_placement_cell == ring_placement_cell) and
        _is_not_out_of_bounds(new_placement_cell):
130     player_placement_cell = new_placement_cell
131     set_cell(0, previous_cell, 0) # deletes contents of previous
        cell (atlas_coords = Vector2i(-1, -1))

```

```

132         set_cell(0, player_placement_cell, 0, PLAYER_SPRITE)
133         _win_game_if_won()
134
135     func _win_game_if_won() -> void:
136         if player_placement_cell == ring_placement_cell:
137             $WinDialog.visible = true
138             get_tree().paused = true
139
140     # ALGORITHM BEGINS HERE
141
142     # Used as inspiration: http://pcg.wikidot.com/pcg-algorithm:voronoi
143     # -diagram
144     # (brute-force implementation in JavaScript, here adapted to
145     # GDScript)
146
147     func paint_points() -> void:
148         for point in points:
149             set_cell(0, Vector2(point["x"], point["y"]), 0, point["type"]
150                 ])
151             for citizen in point["citizens"]:
152                 if _is_in_bounds(point["x"], citizen["dx"], point["y"],
153                     citizen["dy"]):
154                     set_cell(0, Vector2(point["x"] + citizen["dx"], point["
155                         y"] + citizen["dy"]), 0, point["type"])
156
157     func _is_in_bounds(x: int, dx: int, y: int, dy: int) -> bool:
158         return x + dx >= 0 and x + dx < x_tile_range and y + dy >= 0 and
159             y + dy < y_tile_range
160
161     func _squared(x: int) -> int:
162         return x ** 2
163
164     func calculate_points_delta(x: int, y: int, p: int) -> float:

```

```

159     if distance == EUCLIDEAN:
160         return sqrt(_squared(points[p]["x"] - x) + _squared(points[p]
161             ["y"] - y))
162
163     return abs(points[p]["x"] - x) + abs(points[p]["y"] - y)
164
165 func define_points(num_points: int) -> void:
166     var types: Array[Vector2i] = trees.duplicate()
167     types.append_array(buildings)
168     for i in range(num_points):
169         var x: int = randi_range(0, x_tile_range)
170         var y: int = randi_range(0, y_tile_range)
171         var type: Vector2i = types.pick_random()
172         types.erase(type)
173         points.append(
174             {
175                 "type": type,
176                 "x": x,
177                 "y": y,
178                 "citizens": []
179             }
180         )
181     for x in range(x_tile_range):
182         for y in range(y_tile_range):
183             var lowest_delta: Dictionary = {
184                 "point_id": 0,
185                 "delta": x_tile_range * y_tile_range
186             }
187             for p in range(len(points)):
188                 var delta: float = calculate_points_delta(x, y, p)
189                 if delta < lowest_delta["delta"]:
190                     lowest_delta = {
191                         "point_id": p,
192                         "delta": delta

```

```

191         }
192         var active_point: Dictionary = points[lowest_delta["
            point_id"]]
193         var dx: int = x - active_point["x"]
194         var dy: int = y - active_point["y"]
195         active_point["citizens"].append(
196             {
197                 "dx": dx,
198                 "dy": dy
199             }
200         )

```

C.3.6 accept_dialog.tscn

```

1  [gd_scene format=3 uid="uid://cau5jgogdnf53"]
2
3  [node name="AcceptDialog" type="AcceptDialog"]
4  title = "Tree-Munching Time!"
5  position = Vector2i(326, 100)
6  size = Vector2i(500, 421)
7  mouse_passthrough = true
8  ok_button_text = "Bring it on!"
9  dialog_text = "You're a hollow Golem who seeks the ultimate
    treasure; a ring that's got something on top of it. It's
    somewhere in this large village and barely visible to your
    naked eyes, but you'll stop at nothing to get what you want.
    You can chow down every tree and fauna that stands in your way
    of the ring, but your Achilles heel is any bricks and mortar,
    which will make you stop at your tracks. Are you ready to
    attain your treasure?w Golem in a black-and-white world, in
    search for your most desired treasure. It's a ring with
    something on top of it. And you'll stop at nothing to get what

```



```

    you want. You can chow down every tree and fauna that stands in
    your way of the ring, but your Achilles heel is any bricks and
    mortar, which will make you stop at your tracks. Are you ready
    to attain the treasure that is rightfully yours?!"
10  dialog_autowrap = true

```

C.3.7 win_dialog.tscn

```

1  [gd_scene format=3 uid="uid://b5q8ovcigrvyr"]
2
3  [node name="WinDialog" type="ConfirmationDialog"]
4  title = "You Found the Treasure!"
5  position = Vector2i(326, 100)
6  size = Vector2i(500, 421)
7  mouse_passthrough = true
8  ok_button_text = "Get Me a New Village"
9  dialog_text = "You found your treasure! Well done, you!"
10
11  Would you like to travel to a new village in the hopes of finding
    another ring? Or would you like to take your treasure home now?
    "
12  dialog_autowrap = true
13  cancel_button_text = "Get Me Out of Here"

```

C.3.8 icon.svg.import

```

1  [remap]
2
3  importer="texture"
4  type="CompressedTexture2D"
5  uid="uid://du4v6taw8ssax"

```

```

6   path="res://.godot/imported/icon.svg-218
    a8f2b3041327d8a5756f3a245f83b.ctex"
7   metadata={
8     "vram_texture": false
9   }
10
11   [deps]
12
13   source_file="res://icon.svg"
14   dest_files=["res://.godot/imported/icon.svg-218
    a8f2b3041327d8a5756f3a245f83b.ctex"]
15
16   [params]
17
18   compress/mode=0
19   compress/high_quality=false
20   compress/lossy_quality=0.7
21   compress/hdr_compression=1
22   compress/normal_map=0
23   compress/channel_pack=0
24   mipmaps/generate=false
25   mipmaps/limit=-1
26   roughness/mode=0
27   roughness/src_normal=""
28   process/fix_alpha_border=true
29   process/premult_alpha=false
30   process/normal_map_invert_y=false
31   process/hdr_as_srgb=false
32   process/hdr_clamp_exposure=false
33   process/size_limit=0
34   detect_3d/compress_to=1
35   svg/scale=1.0
36   editor/scale_with_editor_scale=false

```

```
37 editor/convert_colors_with_editor_theme=false
```

C.3.9 monochrome_packed.png.import

```
1  [remap]
2
3  importer="texture"
4  type="CompressedTexture2D"
5  uid="uid://cpign73sfbsrt"
6  path="res://.godot/imported/monochrome_packed.png-6
    b9bd1c64dd50f72acd3afd14d1ac34f.ctex"
7  metadata={
8    "vram_texture": false
9  }
10
11  [deps]
12
13  source_file="res://monochrome_packed.png"
14  dest_files=["res://.godot/imported/monochrome_packed.png-6
    b9bd1c64dd50f72acd3afd14d1ac34f.ctex"]
15
16  [params]
17
18  compress/mode=0
19  compress/high_quality=false
20  compress/lossy_quality=0.7
21  compress/hdr_compression=1
22  compress/normal_map=0
23  compress/channel_pack=0
24  mipmaps/generate=false
25  mipmaps/limit=-1
26  roughness/mode=0
```

```

27  roughness/src_normal=""
28  process/fix_alpha_border=true
29  process/premult_alpha=false
30  process/normal_map_invert_y=false
31  process/hdr_as_srgb=false
32  process/hdr_clamp_exposure=false
33  process/size_limit=0
34  detect_3d/compress_to=1

```

C.3.10 LICENSE

```

1  This work is licensed under the Creative Commons Attribution-
    ShareAlike 4.0 International License. To view a copy of this
    license, visit http://creativecommons.org/licenses/by-sa/4.0/
    or send a letter to Creative Commons, PO Box 1866, Mountain
    View, CA 94042, USA.

```

C.4 GD4PoissonRPG

These are the source code listings for the Poisson Disk Sampling implementation of the 2D tile-map RPG scenario in the Godot game engine. The link to the publicly available source code repository is here: <https://github.com/Zishan-Rahman/GD4PoissonRPG>

C.4.1 .gitattributes

```

1  # Normalize EOL for all files that Git considers text files.
2  * text=auto eol=lf

```

C.4.2 .gitignore

```
1  # Godot 4+ specific ignores
2  .godot/
```

C.4.3 project.godot

```
1  ; Engine configuration file.
2  ; It's best edited using the editor UI and not directly,
3  ; since the parameters that go here are not all obvious.
4  ;
5  ; Format:
6  ; [section] ; section goes between []
7  ; param=value ; assign values to parameters
8
9  config_version=5
10
11 [application]
12
13 config/name="Poisson Sampling Project"
14 run/main_scene="res://tile_map.tscn"
15 config/features=PackedStringArray("4.0", "Forward Plus")
16 config/icon="res://icon.svg"
17
18 [display]
19
20 window/size/viewport_height=640
21
22 [rendering]
23
24 environment/defaults/default_clear_color=Color(0, 0, 0, 1)
```

C.4.4 tile_map.tscn

```

1  [gd_scene load_steps=7 format=3 uid="uid://f2kv7fett7"]
2
3  [ext_resource type="Texture2D" uid="uid://c3bpsm4r8t504" path="res
    ://monochrome_packed.png" id="1_uucm3"]
4  [ext_resource type="Script" path="res://tile_map.gd" id="2_iyhvf"]
5  [ext_resource type="PackedScene" uid="uid://cau5jgogdnf53" path="
    res://accept_dialog.tscn" id="3_bk3rg"]
6  [ext_resource type="PackedScene" uid="uid://b5q8ovcigrvyr" path="
    res://win_dialog.tscn" id="4_4hdc7"]
7
8  [sub_resource type="TileSetAtlasSource" id="
    TileSetAtlasSource_j4usm"]
9  texture = ExtResource("1_uucm3")
10 0:0/0 = 0
11 1:0/0 = 0
12 2:0/0 = 0
13 3:0/0 = 0
14 4:0/0 = 0
15 5:0/0 = 0
16 6:0/0 = 0
17 7:0/0 = 0
18 8:0/0 = 0
19 9:0/0 = 0
20 10:0/0 = 0
21 11:0/0 = 0
22 12:0/0 = 0
23 13:0/0 = 0
24 14:0/0 = 0
25 15:0/0 = 0
26 16:0/0 = 0
27 17:0/0 = 0
28 18:0/0 = 0
29 19:0/0 = 0

```

30 20:0/0 = 0
31 21:0/0 = 0
32 22:0/0 = 0
33 23:0/0 = 0
34 24:0/0 = 0
35 25:0/0 = 0
36 26:0/0 = 0
37 27:0/0 = 0
38 28:0/0 = 0
39 29:0/0 = 0
40 30:0/0 = 0
41 31:0/0 = 0
42 32:0/0 = 0
43 33:0/0 = 0
44 34:0/0 = 0
45 35:0/0 = 0
46 36:0/0 = 0
47 37:0/0 = 0
48 38:0/0 = 0
49 39:0/0 = 0
50 40:0/0 = 0
51 41:0/0 = 0
52 42:0/0 = 0
53 43:0/0 = 0
54 44:0/0 = 0
55 45:0/0 = 0
56 46:0/0 = 0
57 47:0/0 = 0
58 48:0/0 = 0
59 0:1/0 = 0
60 1:1/0 = 0
61 2:1/0 = 0
62 3:1/0 = 0

63 4:1/0 = 0
64 5:1/0 = 0
65 6:1/0 = 0
66 7:1/0 = 0
67 8:1/0 = 0
68 9:1/0 = 0
69 10:1/0 = 0
70 11:1/0 = 0
71 12:1/0 = 0
72 13:1/0 = 0
73 14:1/0 = 0
74 15:1/0 = 0
75 16:1/0 = 0
76 17:1/0 = 0
77 18:1/0 = 0
78 19:1/0 = 0
79 20:1/0 = 0
80 21:1/0 = 0
81 22:1/0 = 0
82 23:1/0 = 0
83 24:1/0 = 0
84 25:1/0 = 0
85 26:1/0 = 0
86 27:1/0 = 0
87 28:1/0 = 0
88 29:1/0 = 0
89 30:1/0 = 0
90 31:1/0 = 0
91 32:1/0 = 0
92 33:1/0 = 0
93 34:1/0 = 0
94 35:1/0 = 0
95 36:1/0 = 0

96	$37:1/0 = 0$
97	$38:1/0 = 0$
98	$39:1/0 = 0$
99	$40:1/0 = 0$
100	$41:1/0 = 0$
101	$42:1/0 = 0$
102	$43:1/0 = 0$
103	$44:1/0 = 0$
104	$45:1/0 = 0$
105	$46:1/0 = 0$
106	$47:1/0 = 0$
107	$48:1/0 = 0$
108	$0:2/0 = 0$
109	$1:2/0 = 0$
110	$2:2/0 = 0$
111	$3:2/0 = 0$
112	$4:2/0 = 0$
113	$5:2/0 = 0$
114	$6:2/0 = 0$
115	$7:2/0 = 0$
116	$8:2/0 = 0$
117	$9:2/0 = 0$
118	$10:2/0 = 0$
119	$11:2/0 = 0$
120	$12:2/0 = 0$
121	$13:2/0 = 0$
122	$14:2/0 = 0$
123	$15:2/0 = 0$
124	$16:2/0 = 0$
125	$17:2/0 = 0$
126	$18:2/0 = 0$
127	$19:2/0 = 0$
128	$20:2/0 = 0$

129	$21:2/0 = 0$
130	$22:2/0 = 0$
131	$23:2/0 = 0$
132	$24:2/0 = 0$
133	$25:2/0 = 0$
134	$26:2/0 = 0$
135	$27:2/0 = 0$
136	$28:2/0 = 0$
137	$29:2/0 = 0$
138	$30:2/0 = 0$
139	$31:2/0 = 0$
140	$32:2/0 = 0$
141	$33:2/0 = 0$
142	$34:2/0 = 0$
143	$35:2/0 = 0$
144	$36:2/0 = 0$
145	$37:2/0 = 0$
146	$38:2/0 = 0$
147	$39:2/0 = 0$
148	$40:2/0 = 0$
149	$41:2/0 = 0$
150	$42:2/0 = 0$
151	$43:2/0 = 0$
152	$44:2/0 = 0$
153	$45:2/0 = 0$
154	$46:2/0 = 0$
155	$47:2/0 = 0$
156	$48:2/0 = 0$
157	$0:3/0 = 0$
158	$1:3/0 = 0$
159	$2:3/0 = 0$
160	$3:3/0 = 0$
161	$4:3/0 = 0$

162	$5:3/0 = 0$
163	$6:3/0 = 0$
164	$7:3/0 = 0$
165	$8:3/0 = 0$
166	$9:3/0 = 0$
167	$10:3/0 = 0$
168	$11:3/0 = 0$
169	$12:3/0 = 0$
170	$13:3/0 = 0$
171	$14:3/0 = 0$
172	$15:3/0 = 0$
173	$16:3/0 = 0$
174	$17:3/0 = 0$
175	$18:3/0 = 0$
176	$19:3/0 = 0$
177	$20:3/0 = 0$
178	$21:3/0 = 0$
179	$22:3/0 = 0$
180	$23:3/0 = 0$
181	$24:3/0 = 0$
182	$25:3/0 = 0$
183	$26:3/0 = 0$
184	$27:3/0 = 0$
185	$28:3/0 = 0$
186	$29:3/0 = 0$
187	$30:3/0 = 0$
188	$31:3/0 = 0$
189	$32:3/0 = 0$
190	$33:3/0 = 0$
191	$34:3/0 = 0$
192	$35:3/0 = 0$
193	$36:3/0 = 0$
194	$37:3/0 = 0$

195 38:3/0 = 0
196 39:3/0 = 0
197 40:3/0 = 0
198 41:3/0 = 0
199 42:3/0 = 0
200 43:3/0 = 0
201 44:3/0 = 0
202 45:3/0 = 0
203 46:3/0 = 0
204 47:3/0 = 0
205 48:3/0 = 0
206 0:4/0 = 0
207 1:4/0 = 0
208 2:4/0 = 0
209 3:4/0 = 0
210 4:4/0 = 0
211 5:4/0 = 0
212 6:4/0 = 0
213 7:4/0 = 0
214 8:4/0 = 0
215 9:4/0 = 0
216 10:4/0 = 0
217 11:4/0 = 0
218 12:4/0 = 0
219 13:4/0 = 0
220 14:4/0 = 0
221 15:4/0 = 0
222 16:4/0 = 0
223 17:4/0 = 0
224 18:4/0 = 0
225 19:4/0 = 0
226 20:4/0 = 0
227 21:4/0 = 0

228	$22:4/0 = 0$
229	$23:4/0 = 0$
230	$24:4/0 = 0$
231	$25:4/0 = 0$
232	$26:4/0 = 0$
233	$27:4/0 = 0$
234	$28:4/0 = 0$
235	$29:4/0 = 0$
236	$30:4/0 = 0$
237	$31:4/0 = 0$
238	$32:4/0 = 0$
239	$33:4/0 = 0$
240	$34:4/0 = 0$
241	$35:4/0 = 0$
242	$36:4/0 = 0$
243	$37:4/0 = 0$
244	$38:4/0 = 0$
245	$39:4/0 = 0$
246	$40:4/0 = 0$
247	$41:4/0 = 0$
248	$42:4/0 = 0$
249	$43:4/0 = 0$
250	$44:4/0 = 0$
251	$45:4/0 = 0$
252	$46:4/0 = 0$
253	$47:4/0 = 0$
254	$48:4/0 = 0$
255	$0:5/0 = 0$
256	$1:5/0 = 0$
257	$2:5/0 = 0$
258	$3:5/0 = 0$
259	$4:5/0 = 0$
260	$5:5/0 = 0$

261 6:5/0 = 0
262 7:5/0 = 0
263 8:5/0 = 0
264 9:5/0 = 0
265 10:5/0 = 0
266 11:5/0 = 0
267 12:5/0 = 0
268 13:5/0 = 0
269 14:5/0 = 0
270 15:5/0 = 0
271 16:5/0 = 0
272 17:5/0 = 0
273 18:5/0 = 0
274 19:5/0 = 0
275 20:5/0 = 0
276 21:5/0 = 0
277 22:5/0 = 0
278 23:5/0 = 0
279 24:5/0 = 0
280 25:5/0 = 0
281 26:5/0 = 0
282 27:5/0 = 0
283 28:5/0 = 0
284 29:5/0 = 0
285 30:5/0 = 0
286 31:5/0 = 0
287 32:5/0 = 0
288 33:5/0 = 0
289 34:5/0 = 0
290 35:5/0 = 0
291 36:5/0 = 0
292 37:5/0 = 0
293 38:5/0 = 0

294	$39:5/0 = 0$
295	$40:5/0 = 0$
296	$41:5/0 = 0$
297	$42:5/0 = 0$
298	$43:5/0 = 0$
299	$44:5/0 = 0$
300	$45:5/0 = 0$
301	$46:5/0 = 0$
302	$47:5/0 = 0$
303	$48:5/0 = 0$
304	$0:6/0 = 0$
305	$1:6/0 = 0$
306	$2:6/0 = 0$
307	$3:6/0 = 0$
308	$4:6/0 = 0$
309	$5:6/0 = 0$
310	$6:6/0 = 0$
311	$7:6/0 = 0$
312	$8:6/0 = 0$
313	$9:6/0 = 0$
314	$10:6/0 = 0$
315	$11:6/0 = 0$
316	$12:6/0 = 0$
317	$13:6/0 = 0$
318	$14:6/0 = 0$
319	$15:6/0 = 0$
320	$16:6/0 = 0$
321	$17:6/0 = 0$
322	$18:6/0 = 0$
323	$19:6/0 = 0$
324	$20:6/0 = 0$
325	$21:6/0 = 0$
326	$22:6/0 = 0$

327 23:6/0 = 0
328 24:6/0 = 0
329 25:6/0 = 0
330 26:6/0 = 0
331 27:6/0 = 0
332 28:6/0 = 0
333 29:6/0 = 0
334 30:6/0 = 0
335 31:6/0 = 0
336 32:6/0 = 0
337 33:6/0 = 0
338 34:6/0 = 0
339 35:6/0 = 0
340 36:6/0 = 0
341 37:6/0 = 0
342 38:6/0 = 0
343 39:6/0 = 0
344 40:6/0 = 0
345 41:6/0 = 0
346 42:6/0 = 0
347 43:6/0 = 0
348 44:6/0 = 0
349 45:6/0 = 0
350 46:6/0 = 0
351 47:6/0 = 0
352 48:6/0 = 0
353 0:7/0 = 0
354 1:7/0 = 0
355 2:7/0 = 0
356 3:7/0 = 0
357 4:7/0 = 0
358 5:7/0 = 0
359 6:7/0 = 0

360 7:7/0 = 0
361 8:7/0 = 0
362 9:7/0 = 0
363 10:7/0 = 0
364 11:7/0 = 0
365 12:7/0 = 0
366 13:7/0 = 0
367 14:7/0 = 0
368 15:7/0 = 0
369 16:7/0 = 0
370 17:7/0 = 0
371 18:7/0 = 0
372 19:7/0 = 0
373 20:7/0 = 0
374 21:7/0 = 0
375 22:7/0 = 0
376 23:7/0 = 0
377 24:7/0 = 0
378 25:7/0 = 0
379 26:7/0 = 0
380 27:7/0 = 0
381 28:7/0 = 0
382 29:7/0 = 0
383 30:7/0 = 0
384 31:7/0 = 0
385 32:7/0 = 0
386 33:7/0 = 0
387 34:7/0 = 0
388 35:7/0 = 0
389 36:7/0 = 0
390 37:7/0 = 0
391 38:7/0 = 0
392 39:7/0 = 0

393 $40:7/0 = 0$
394 $41:7/0 = 0$
395 $42:7/0 = 0$
396 $43:7/0 = 0$
397 $44:7/0 = 0$
398 $45:7/0 = 0$
399 $46:7/0 = 0$
400 $47:7/0 = 0$
401 $48:7/0 = 0$
402 $0:8/0 = 0$
403 $1:8/0 = 0$
404 $2:8/0 = 0$
405 $3:8/0 = 0$
406 $4:8/0 = 0$
407 $5:8/0 = 0$
408 $6:8/0 = 0$
409 $7:8/0 = 0$
410 $8:8/0 = 0$
411 $9:8/0 = 0$
412 $10:8/0 = 0$
413 $11:8/0 = 0$
414 $12:8/0 = 0$
415 $13:8/0 = 0$
416 $14:8/0 = 0$
417 $15:8/0 = 0$
418 $16:8/0 = 0$
419 $17:8/0 = 0$
420 $18:8/0 = 0$
421 $19:8/0 = 0$
422 $20:8/0 = 0$
423 $21:8/0 = 0$
424 $22:8/0 = 0$
425 $23:8/0 = 0$

426 24:8/0 = 0
427 25:8/0 = 0
428 26:8/0 = 0
429 27:8/0 = 0
430 28:8/0 = 0
431 29:8/0 = 0
432 30:8/0 = 0
433 31:8/0 = 0
434 32:8/0 = 0
435 33:8/0 = 0
436 34:8/0 = 0
437 35:8/0 = 0
438 36:8/0 = 0
439 37:8/0 = 0
440 38:8/0 = 0
441 39:8/0 = 0
442 40:8/0 = 0
443 41:8/0 = 0
444 42:8/0 = 0
445 43:8/0 = 0
446 44:8/0 = 0
447 45:8/0 = 0
448 46:8/0 = 0
449 47:8/0 = 0
450 48:8/0 = 0
451 0:9/0 = 0
452 1:9/0 = 0
453 2:9/0 = 0
454 3:9/0 = 0
455 4:9/0 = 0
456 5:9/0 = 0
457 6:9/0 = 0
458 7:9/0 = 0

459 $8:9/0 = 0$
460 $9:9/0 = 0$
461 $10:9/0 = 0$
462 $11:9/0 = 0$
463 $12:9/0 = 0$
464 $13:9/0 = 0$
465 $14:9/0 = 0$
466 $15:9/0 = 0$
467 $16:9/0 = 0$
468 $17:9/0 = 0$
469 $18:9/0 = 0$
470 $19:9/0 = 0$
471 $20:9/0 = 0$
472 $21:9/0 = 0$
473 $22:9/0 = 0$
474 $23:9/0 = 0$
475 $24:9/0 = 0$
476 $25:9/0 = 0$
477 $26:9/0 = 0$
478 $27:9/0 = 0$
479 $28:9/0 = 0$
480 $29:9/0 = 0$
481 $30:9/0 = 0$
482 $31:9/0 = 0$
483 $32:9/0 = 0$
484 $33:9/0 = 0$
485 $34:9/0 = 0$
486 $35:9/0 = 0$
487 $36:9/0 = 0$
488 $37:9/0 = 0$
489 $38:9/0 = 0$
490 $39:9/0 = 0$
491 $40:9/0 = 0$

492 41:9/0 = 0
493 42:9/0 = 0
494 43:9/0 = 0
495 44:9/0 = 0
496 45:9/0 = 0
497 46:9/0 = 0
498 47:9/0 = 0
499 48:9/0 = 0
500 0:10/0 = 0
501 1:10/0 = 0
502 2:10/0 = 0
503 3:10/0 = 0
504 4:10/0 = 0
505 5:10/0 = 0
506 6:10/0 = 0
507 7:10/0 = 0
508 8:10/0 = 0
509 9:10/0 = 0
510 10:10/0 = 0
511 11:10/0 = 0
512 12:10/0 = 0
513 13:10/0 = 0
514 14:10/0 = 0
515 15:10/0 = 0
516 16:10/0 = 0
517 17:10/0 = 0
518 18:10/0 = 0
519 19:10/0 = 0
520 20:10/0 = 0
521 21:10/0 = 0
522 22:10/0 = 0
523 23:10/0 = 0
524 24:10/0 = 0

525 25:10/0 = 0
526 26:10/0 = 0
527 27:10/0 = 0
528 28:10/0 = 0
529 29:10/0 = 0
530 30:10/0 = 0
531 31:10/0 = 0
532 32:10/0 = 0
533 33:10/0 = 0
534 34:10/0 = 0
535 35:10/0 = 0
536 36:10/0 = 0
537 37:10/0 = 0
538 38:10/0 = 0
539 39:10/0 = 0
540 40:10/0 = 0
541 41:10/0 = 0
542 42:10/0 = 0
543 43:10/0 = 0
544 44:10/0 = 0
545 45:10/0 = 0
546 46:10/0 = 0
547 47:10/0 = 0
548 48:10/0 = 0
549 0:11/0 = 0
550 1:11/0 = 0
551 2:11/0 = 0
552 3:11/0 = 0
553 4:11/0 = 0
554 5:11/0 = 0
555 6:11/0 = 0
556 7:11/0 = 0
557 8:11/0 = 0

558 9:11/0 = 0
559 10:11/0 = 0
560 11:11/0 = 0
561 12:11/0 = 0
562 13:11/0 = 0
563 14:11/0 = 0
564 15:11/0 = 0
565 16:11/0 = 0
566 17:11/0 = 0
567 18:11/0 = 0
568 19:11/0 = 0
569 20:11/0 = 0
570 21:11/0 = 0
571 22:11/0 = 0
572 23:11/0 = 0
573 24:11/0 = 0
574 25:11/0 = 0
575 26:11/0 = 0
576 27:11/0 = 0
577 28:11/0 = 0
578 29:11/0 = 0
579 30:11/0 = 0
580 31:11/0 = 0
581 32:11/0 = 0
582 33:11/0 = 0
583 34:11/0 = 0
584 35:11/0 = 0
585 36:11/0 = 0
586 37:11/0 = 0
587 38:11/0 = 0
588 39:11/0 = 0
589 40:11/0 = 0
590 41:11/0 = 0

591 42:11/0 = 0
592 43:11/0 = 0
593 44:11/0 = 0
594 45:11/0 = 0
595 46:11/0 = 0
596 47:11/0 = 0
597 48:11/0 = 0
598 0:12/0 = 0
599 1:12/0 = 0
600 2:12/0 = 0
601 3:12/0 = 0
602 4:12/0 = 0
603 5:12/0 = 0
604 6:12/0 = 0
605 7:12/0 = 0
606 8:12/0 = 0
607 9:12/0 = 0
608 10:12/0 = 0
609 11:12/0 = 0
610 12:12/0 = 0
611 13:12/0 = 0
612 14:12/0 = 0
613 15:12/0 = 0
614 16:12/0 = 0
615 17:12/0 = 0
616 18:12/0 = 0
617 19:12/0 = 0
618 20:12/0 = 0
619 21:12/0 = 0
620 22:12/0 = 0
621 23:12/0 = 0
622 24:12/0 = 0
623 25:12/0 = 0

624 26:12/0 = 0
625 27:12/0 = 0
626 28:12/0 = 0
627 29:12/0 = 0
628 30:12/0 = 0
629 31:12/0 = 0
630 32:12/0 = 0
631 33:12/0 = 0
632 34:12/0 = 0
633 35:12/0 = 0
634 36:12/0 = 0
635 37:12/0 = 0
636 38:12/0 = 0
637 39:12/0 = 0
638 40:12/0 = 0
639 41:12/0 = 0
640 42:12/0 = 0
641 43:12/0 = 0
642 44:12/0 = 0
643 45:12/0 = 0
644 46:12/0 = 0
645 47:12/0 = 0
646 48:12/0 = 0
647 0:13/0 = 0
648 1:13/0 = 0
649 2:13/0 = 0
650 3:13/0 = 0
651 4:13/0 = 0
652 5:13/0 = 0
653 6:13/0 = 0
654 7:13/0 = 0
655 8:13/0 = 0
656 9:13/0 = 0

657 10:13/0 = 0
658 11:13/0 = 0
659 12:13/0 = 0
660 13:13/0 = 0
661 14:13/0 = 0
662 15:13/0 = 0
663 16:13/0 = 0
664 17:13/0 = 0
665 18:13/0 = 0
666 19:13/0 = 0
667 20:13/0 = 0
668 21:13/0 = 0
669 22:13/0 = 0
670 23:13/0 = 0
671 24:13/0 = 0
672 25:13/0 = 0
673 26:13/0 = 0
674 27:13/0 = 0
675 28:13/0 = 0
676 29:13/0 = 0
677 30:13/0 = 0
678 31:13/0 = 0
679 32:13/0 = 0
680 33:13/0 = 0
681 34:13/0 = 0
682 35:13/0 = 0
683 36:13/0 = 0
684 37:13/0 = 0
685 38:13/0 = 0
686 39:13/0 = 0
687 40:13/0 = 0
688 41:13/0 = 0
689 42:13/0 = 0

690 43:13/0 = 0
691 44:13/0 = 0
692 45:13/0 = 0
693 46:13/0 = 0
694 47:13/0 = 0
695 48:13/0 = 0
696 0:14/0 = 0
697 1:14/0 = 0
698 2:14/0 = 0
699 3:14/0 = 0
700 4:14/0 = 0
701 5:14/0 = 0
702 6:14/0 = 0
703 7:14/0 = 0
704 8:14/0 = 0
705 9:14/0 = 0
706 10:14/0 = 0
707 11:14/0 = 0
708 12:14/0 = 0
709 13:14/0 = 0
710 14:14/0 = 0
711 15:14/0 = 0
712 16:14/0 = 0
713 17:14/0 = 0
714 18:14/0 = 0
715 19:14/0 = 0
716 20:14/0 = 0
717 21:14/0 = 0
718 22:14/0 = 0
719 23:14/0 = 0
720 24:14/0 = 0
721 25:14/0 = 0
722 26:14/0 = 0

723	$27:14/0 = 0$
724	$28:14/0 = 0$
725	$29:14/0 = 0$
726	$30:14/0 = 0$
727	$31:14/0 = 0$
728	$32:14/0 = 0$
729	$33:14/0 = 0$
730	$34:14/0 = 0$
731	$35:14/0 = 0$
732	$36:14/0 = 0$
733	$37:14/0 = 0$
734	$38:14/0 = 0$
735	$39:14/0 = 0$
736	$40:14/0 = 0$
737	$41:14/0 = 0$
738	$42:14/0 = 0$
739	$43:14/0 = 0$
740	$44:14/0 = 0$
741	$45:14/0 = 0$
742	$46:14/0 = 0$
743	$47:14/0 = 0$
744	$48:14/0 = 0$
745	$0:15/0 = 0$
746	$1:15/0 = 0$
747	$2:15/0 = 0$
748	$3:15/0 = 0$
749	$4:15/0 = 0$
750	$5:15/0 = 0$
751	$6:15/0 = 0$
752	$7:15/0 = 0$
753	$8:15/0 = 0$
754	$9:15/0 = 0$
755	$10:15/0 = 0$

756 11:15/0 = 0
757 12:15/0 = 0
758 13:15/0 = 0
759 14:15/0 = 0
760 15:15/0 = 0
761 16:15/0 = 0
762 17:15/0 = 0
763 18:15/0 = 0
764 19:15/0 = 0
765 20:15/0 = 0
766 21:15/0 = 0
767 22:15/0 = 0
768 23:15/0 = 0
769 24:15/0 = 0
770 25:15/0 = 0
771 26:15/0 = 0
772 27:15/0 = 0
773 28:15/0 = 0
774 29:15/0 = 0
775 30:15/0 = 0
776 31:15/0 = 0
777 32:15/0 = 0
778 33:15/0 = 0
779 34:15/0 = 0
780 35:15/0 = 0
781 36:15/0 = 0
782 37:15/0 = 0
783 38:15/0 = 0
784 39:15/0 = 0
785 40:15/0 = 0
786 41:15/0 = 0
787 42:15/0 = 0
788 43:15/0 = 0

789	$44:15/0 = 0$
790	$45:15/0 = 0$
791	$46:15/0 = 0$
792	$47:15/0 = 0$
793	$48:15/0 = 0$
794	$0:16/0 = 0$
795	$1:16/0 = 0$
796	$2:16/0 = 0$
797	$3:16/0 = 0$
798	$4:16/0 = 0$
799	$5:16/0 = 0$
800	$6:16/0 = 0$
801	$7:16/0 = 0$
802	$8:16/0 = 0$
803	$9:16/0 = 0$
804	$10:16/0 = 0$
805	$11:16/0 = 0$
806	$12:16/0 = 0$
807	$13:16/0 = 0$
808	$14:16/0 = 0$
809	$15:16/0 = 0$
810	$16:16/0 = 0$
811	$17:16/0 = 0$
812	$18:16/0 = 0$
813	$19:16/0 = 0$
814	$20:16/0 = 0$
815	$21:16/0 = 0$
816	$22:16/0 = 0$
817	$23:16/0 = 0$
818	$24:16/0 = 0$
819	$25:16/0 = 0$
820	$26:16/0 = 0$
821	$27:16/0 = 0$

822 28:16/0 = 0
823 29:16/0 = 0
824 30:16/0 = 0
825 31:16/0 = 0
826 32:16/0 = 0
827 33:16/0 = 0
828 34:16/0 = 0
829 35:16/0 = 0
830 36:16/0 = 0
831 37:16/0 = 0
832 38:16/0 = 0
833 39:16/0 = 0
834 40:16/0 = 0
835 41:16/0 = 0
836 42:16/0 = 0
837 43:16/0 = 0
838 44:16/0 = 0
839 45:16/0 = 0
840 46:16/0 = 0
841 47:16/0 = 0
842 48:16/0 = 0
843 0:17/0 = 0
844 1:17/0 = 0
845 2:17/0 = 0
846 3:17/0 = 0
847 4:17/0 = 0
848 5:17/0 = 0
849 6:17/0 = 0
850 7:17/0 = 0
851 8:17/0 = 0
852 9:17/0 = 0
853 10:17/0 = 0
854 11:17/0 = 0

855 12:17/0 = 0
856 13:17/0 = 0
857 14:17/0 = 0
858 15:17/0 = 0
859 16:17/0 = 0
860 17:17/0 = 0
861 18:17/0 = 0
862 19:17/0 = 0
863 20:17/0 = 0
864 21:17/0 = 0
865 22:17/0 = 0
866 23:17/0 = 0
867 24:17/0 = 0
868 25:17/0 = 0
869 26:17/0 = 0
870 27:17/0 = 0
871 28:17/0 = 0
872 29:17/0 = 0
873 30:17/0 = 0
874 31:17/0 = 0
875 32:17/0 = 0
876 33:17/0 = 0
877 34:17/0 = 0
878 35:17/0 = 0
879 36:17/0 = 0
880 37:17/0 = 0
881 38:17/0 = 0
882 39:17/0 = 0
883 40:17/0 = 0
884 41:17/0 = 0
885 42:17/0 = 0
886 43:17/0 = 0
887 44:17/0 = 0

888 45:17/0 = 0
889 46:17/0 = 0
890 47:17/0 = 0
891 48:17/0 = 0
892 0:18/0 = 0
893 1:18/0 = 0
894 2:18/0 = 0
895 3:18/0 = 0
896 4:18/0 = 0
897 5:18/0 = 0
898 6:18/0 = 0
899 7:18/0 = 0
900 8:18/0 = 0
901 9:18/0 = 0
902 10:18/0 = 0
903 11:18/0 = 0
904 12:18/0 = 0
905 13:18/0 = 0
906 14:18/0 = 0
907 15:18/0 = 0
908 16:18/0 = 0
909 17:18/0 = 0
910 18:18/0 = 0
911 19:18/0 = 0
912 20:18/0 = 0
913 21:18/0 = 0
914 22:18/0 = 0
915 23:18/0 = 0
916 24:18/0 = 0
917 25:18/0 = 0
918 26:18/0 = 0
919 27:18/0 = 0
920 28:18/0 = 0

921 29:18/0 = 0
922 30:18/0 = 0
923 31:18/0 = 0
924 32:18/0 = 0
925 33:18/0 = 0
926 34:18/0 = 0
927 35:18/0 = 0
928 36:18/0 = 0
929 37:18/0 = 0
930 38:18/0 = 0
931 39:18/0 = 0
932 40:18/0 = 0
933 41:18/0 = 0
934 42:18/0 = 0
935 43:18/0 = 0
936 44:18/0 = 0
937 45:18/0 = 0
938 46:18/0 = 0
939 47:18/0 = 0
940 48:18/0 = 0
941 0:19/0 = 0
942 1:19/0 = 0
943 2:19/0 = 0
944 3:19/0 = 0
945 4:19/0 = 0
946 5:19/0 = 0
947 6:19/0 = 0
948 7:19/0 = 0
949 8:19/0 = 0
950 9:19/0 = 0
951 10:19/0 = 0
952 11:19/0 = 0
953 12:19/0 = 0

954 13:19/0 = 0
955 14:19/0 = 0
956 15:19/0 = 0
957 16:19/0 = 0
958 17:19/0 = 0
959 18:19/0 = 0
960 19:19/0 = 0
961 20:19/0 = 0
962 21:19/0 = 0
963 22:19/0 = 0
964 23:19/0 = 0
965 24:19/0 = 0
966 25:19/0 = 0
967 26:19/0 = 0
968 27:19/0 = 0
969 28:19/0 = 0
970 29:19/0 = 0
971 30:19/0 = 0
972 31:19/0 = 0
973 32:19/0 = 0
974 33:19/0 = 0
975 34:19/0 = 0
976 35:19/0 = 0
977 36:19/0 = 0
978 37:19/0 = 0
979 38:19/0 = 0
980 39:19/0 = 0
981 40:19/0 = 0
982 41:19/0 = 0
983 42:19/0 = 0
984 43:19/0 = 0
985 44:19/0 = 0
986 45:19/0 = 0

987	$46:19/0 = 0$
988	$47:19/0 = 0$
989	$48:19/0 = 0$
990	$0:20/0 = 0$
991	$1:20/0 = 0$
992	$2:20/0 = 0$
993	$3:20/0 = 0$
994	$4:20/0 = 0$
995	$5:20/0 = 0$
996	$6:20/0 = 0$
997	$7:20/0 = 0$
998	$8:20/0 = 0$
999	$9:20/0 = 0$
1000	$10:20/0 = 0$
1001	$11:20/0 = 0$
1002	$12:20/0 = 0$
1003	$13:20/0 = 0$
1004	$14:20/0 = 0$
1005	$15:20/0 = 0$
1006	$16:20/0 = 0$
1007	$17:20/0 = 0$
1008	$18:20/0 = 0$
1009	$19:20/0 = 0$
1010	$20:20/0 = 0$
1011	$21:20/0 = 0$
1012	$22:20/0 = 0$
1013	$23:20/0 = 0$
1014	$24:20/0 = 0$
1015	$25:20/0 = 0$
1016	$26:20/0 = 0$
1017	$27:20/0 = 0$
1018	$28:20/0 = 0$
1019	$29:20/0 = 0$

1020	$30:20/0 = 0$
1021	$31:20/0 = 0$
1022	$32:20/0 = 0$
1023	$33:20/0 = 0$
1024	$34:20/0 = 0$
1025	$35:20/0 = 0$
1026	$36:20/0 = 0$
1027	$37:20/0 = 0$
1028	$38:20/0 = 0$
1029	$39:20/0 = 0$
1030	$40:20/0 = 0$
1031	$41:20/0 = 0$
1032	$42:20/0 = 0$
1033	$43:20/0 = 0$
1034	$44:20/0 = 0$
1035	$45:20/0 = 0$
1036	$46:20/0 = 0$
1037	$47:20/0 = 0$
1038	$48:20/0 = 0$
1039	$0:21/0 = 0$
1040	$1:21/0 = 0$
1041	$2:21/0 = 0$
1042	$3:21/0 = 0$
1043	$4:21/0 = 0$
1044	$5:21/0 = 0$
1045	$6:21/0 = 0$
1046	$7:21/0 = 0$
1047	$8:21/0 = 0$
1048	$9:21/0 = 0$
1049	$10:21/0 = 0$
1050	$11:21/0 = 0$
1051	$12:21/0 = 0$
1052	$13:21/0 = 0$

1053	$14:21/0 = 0$
1054	$15:21/0 = 0$
1055	$16:21/0 = 0$
1056	$17:21/0 = 0$
1057	$18:21/0 = 0$
1058	$19:21/0 = 0$
1059	$20:21/0 = 0$
1060	$21:21/0 = 0$
1061	$22:21/0 = 0$
1062	$23:21/0 = 0$
1063	$24:21/0 = 0$
1064	$25:21/0 = 0$
1065	$26:21/0 = 0$
1066	$27:21/0 = 0$
1067	$28:21/0 = 0$
1068	$29:21/0 = 0$
1069	$30:21/0 = 0$
1070	$31:21/0 = 0$
1071	$32:21/0 = 0$
1072	$33:21/0 = 0$
1073	$34:21/0 = 0$
1074	$35:21/0 = 0$
1075	$36:21/0 = 0$
1076	$37:21/0 = 0$
1077	$38:21/0 = 0$
1078	$39:21/0 = 0$
1079	$40:21/0 = 0$
1080	$41:21/0 = 0$
1081	$42:21/0 = 0$
1082	$43:21/0 = 0$
1083	$44:21/0 = 0$
1084	$45:21/0 = 0$
1085	$46:21/0 = 0$

```

1086 47:21/0 = 0
1087 48:21/0 = 0
1088
1089 [sub_resource type="TileSet" id="TileSet_8pb5m"]
1090 sources/0 = SubResource("TileSetAtlasSource_j4usm")
1091
1092 [node name="TileMap" type="TileMap"]
1093 tile_set = SubResource("TileSet_8pb5m")
1094 format = 2
1095 script = ExtResource("2_iyhvf")
1096
1097 [node name="AcceptDialog" parent="." instance=ExtResource("3_bk3rg"
    )]
1098
1099 [node name="WinDialog" parent="." instance=ExtResource("4_4hdc7")]

```

C.4.5 tile_map.gd

```

1  extends TileMap
2
3  const buildings: Array[Vector2i] = [
4      Vector2i(0, 19),
5      Vector2i(1, 19),
6      Vector2i(2, 19),
7      Vector2i(3, 19),
8      Vector2i(4, 19),
9      Vector2i(5, 19),
10     Vector2i(6, 19),
11     Vector2i(7, 19),
12     Vector2i(8, 20),
13     Vector2i(0, 20),
14     Vector2i(1, 20),

```

```

15     Vector2i(2, 20),
16     Vector2i(3, 20),
17     Vector2i(4, 20),
18     Vector2i(5, 20),
19     Vector2i(6, 20),
20     Vector2i(7, 20),
21     Vector2i(8, 20),
22     Vector2i(0, 21),
23     Vector2i(1, 21),
24     Vector2i(2, 21),
25     Vector2i(3, 21),
26     Vector2i(4, 21),
27     Vector2i(5, 21),
28     Vector2i(6, 21),
29     Vector2i(7, 21),
30     Vector2i(8, 21)
31 ]
32 const trees: Array[Vector2i] = [
33     Vector2i(0,1),
34     Vector2i(1,1),
35     Vector2i(2,1),
36     Vector2i(3,1),
37     Vector2i(4,1),
38     Vector2i(5,1),
39     Vector2i(6,1),
40     Vector2i(7,1),
41     Vector2i(0,2),
42     Vector2i(1,2),
43     Vector2i(2,2),
44     Vector2i(3,2),
45     Vector2i(4,2)
46 ]
47 const PLAYER_SPRITE: Vector2i = Vector2i(24, 7)

```



```

48  var player_placement_cell: Vector2i
49  const rings: Array[Vector2i] = [
50      Vector2i(43, 6),
51      Vector2i(44, 6),
52      Vector2i(45, 6),
53      Vector2i(46, 6)
54  ]
55  var ring_placement_cell: Vector2i
56
57  var x_tile_range: int = ProjectSettings.get_setting("display/window
    /size/viewport_width") / tile_set.tile_size.x
58  var y_tile_range: int = ProjectSettings.get_setting("display/window
    /size/viewport_height") / tile_set.tile_size.y
59
60  var cell_points: Array[Vector2]
61  ## The probability that a building gets painted at a cell in lieu
    of a tree. The higher this probability, the more likely a
    building tile gets painted instead of a tree tile.
62  @export_range(0.0, 1.0) var paint_building_probability: float =
    0.125
63  ## The radius value used to measure distances between points for
    the algorithm. The longer the radius, the further apart points
    are during the algorithm's processing, and the further apart
    painted cells are in the game.
64  @export_range(0.5, 2.5) var point_radius: float = 1.0
65  ## The size of the region in which the algorithm is performed. Set
    to the "default" tile map size (72, 40) in the script, shown as
    (0, 0) in the Godot editor. Can be changed to use a smaller
    region for the algorithm itself, of course resulting in less
    cells covered within the boundaries set for this game, though
    the algorithm will perform faster due to less cells being
    checked.
66  @export var region_size: Vector2 = Vector2(x_tile_range,

```

```

        y_tile_range)
67  ## The maximum number of times a cell is checked before it is
    ignored. A cell can be accepted and painted on before this
    maximum number is reached. The higher this value, the more
    times a cell is checked, therefore the higher the algorithm's
    processing time.
68  @export_range(1, 50, 1) var rejection_samples: int = 8
69
70  # Called when the node enters the scene tree for the first time.
71  func _ready():
72      randomize()
73      var start_time: float = Time.get_ticks_msec()
74      cell_points = generate_points(point_radius, region_size,
        rejection_samples)
75      paint_points()
76      place_player()
77      place_ring()
78      var new_time: float = Time.get_ticks_msec() - start_time
79      print("Time taken: " + str(new_time) + "ms")
80      $AcceptDialog.dialog_text = "You're a hollow Golem who seeks the
        ultimate treasure; a ring that's got something on top of it
        . It's somewhere in this large village and barely visible to
        your naked eyes, which took us " + str(new_time) + "
        milliseconds to generate (" + str(new_time / 1000.0) + "
        seconds), but you'll stop at nothing to get what you want.
        You can chow down every tree and fauna that stands in your
        way of the ring, but your Achilles heel is any bricks and
        mortar, which WILL make you stop at your tracks. Since it's
        easy to get lost in here, we'll tell you that you're in
        position " + str(player_placement_cell) + " in this big
        village of size " + str(Vector2i(x_tile_range, y_tile_range)
        ) + ". However, it is YOUR job to find the ring, so are you
        ready to attain the treasure that is rightfully yours?!"

```

```

81     $AcceptDialog.visible = true
82     $AcceptDialog.confirmed.connect(_on_AcceptDialog_closed)
83     $AcceptDialog.canceled.connect(_on_AcceptDialog_closed)
84     $WinDialog.confirmed.connect(_on_WinDialog_confirmed)
85     $WinDialog.canceled.connect(_on_WinDialog_canceled)
86     get_tree().paused = true
87
88     func _on_WinDialog_confirmed() -> void:
89         get_tree().reload_current_scene()
90
91     func _on_WinDialog_canceled() -> void:
92         get_tree().quit()
93
94     func _on_AcceptDialog_closed() -> void:
95         $AcceptDialog.visible = false
96         get_tree().paused = false
97
98     func paint_points() -> void:
99         for point in cell_points:
100             var cell_point: Vector2i = Vector2i(roundi(point.x), roundi(
                point.y))
101             if randf() < paint_building_probability:
102                 set_cell(0, cell_point, 0, buildings.pick_random())
103             else:
104                 set_cell(0, cell_point, 0, trees.pick_random())
105
106     func _get_random_placement_cell() -> Vector2i:
107         return Vector2i(randi() % x_tile_range, randi() % y_tile_range)
108
109     func place_player() -> void:
110         player_placement_cell = _get_random_placement_cell()
111         while get_used_cells(0).has(player_placement_cell):
112             player_placement_cell = _get_random_placement_cell()

```

```

113     set_cell(0, player_placement_cell, 0, PLAYER_SPRITE)
114
115     func place_ring() -> void:
116         ring_placement_cell = _get_random_placement_cell()
117         while get_used_cells(0).has(ring_placement_cell):
118             ring_placement_cell = _get_random_placement_cell()
119         set_cell(0, ring_placement_cell, 0, rings.pick_random())
120
121     func _is_not_out_of_bounds(cell: Vector2i) -> bool:
122         return cell.x >= 0 and cell.x < x_tile_range and cell.y >= 0 and
            cell.y < y_tile_range
123
124     func _physics_process(_delta) -> void:
125         var previous_cell: Vector2i = player_placement_cell
126         var direction: Vector2i = Vector2i.ZERO
127         if Input.is_action_pressed("ui_up"): direction = Vector2i.UP
128         elif Input.is_action_pressed("ui_down"): direction = Vector2i.
            DOWN
129         elif Input.is_action_pressed("ui_left"): direction = Vector2i.
            LEFT
130         elif Input.is_action_pressed("ui_right"): direction = Vector2i.
            RIGHT
131         var new_placement_cell: Vector2i = player_placement_cell +
            direction
132         if (not get_used_cells(0).has(new_placement_cell) or trees.has(
            get_cell_atlas_coords(0, new_placement_cell)) or
            new_placement_cell == ring_placement_cell) and
            _is_not_out_of_bounds(new_placement_cell):
133             player_placement_cell = new_placement_cell
134             set_cell(0, previous_cell, 0) # deletes contents of previous
                cell (atlas_coords = Vector2i(-1, -1))
135             set_cell(0, player_placement_cell, 0, PLAYER_SPRITE)
136             if player_placement_cell == ring_placement_cell:

```

```

137         $WinDialog.visible = true
138         get_tree().paused = true
139
140     # ALGORITHM BEGINS HERE
141
142     func generate_points(radius: float, sample_region_size: Vector2,
        number_of_samples_before_rejection: int = 30) -> Array[Vector2
        ]:
143         var cell_size: float = radius / sqrt(2)
144         var grid: Array[Array] = []
145         var points: Array[Vector2] = []
146         var spawn_points: Array[Vector2] = []
147         var grid_x_axis_size: int = ceili(sample_region_size.x/cell_size
            )
148         var grid_y_axis_size: int = ceili(sample_region_size.y/cell_size
            )
149
150         for i in range(grid_x_axis_size):
151             grid.append([])
152             for j in range(grid_y_axis_size):
153                 grid[i].append(0)
154
155         spawn_points.append(sample_region_size/2)
156
157         while spawn_points.size() > 0:
158             var spawn_index: int = randi_range(0, spawn_points.size() -
                1)
159             var spawn_centre: Vector2 = spawn_points[spawn_index]
160             var candidate_accepted: bool = false
161
162             for i in range(number_of_samples_before_rejection):
163                 var angle: float = randf() * TAU # TAU = PI * 2
164                 var direction: Vector2 = Vector2(sin(angle), cos(angle))

```

```

165         var candidate: Vector2 = spawn_centre + direction *
            randf_range(radius, 2 * radius)
166         if is_valid(candidate, sample_region_size, cell_size,
            radius, points, grid, grid_x_axis_size,
            grid_y_axis_size):
167             points.append(candidate)
168             spawn_points.append(candidate)
169             grid[int(candidate.x/cell_size)][int(candidate.y/
                cell_size)] = len(points)
170             candidate_accepted = true
171             break
172
173         if not candidate_accepted:
174             spawn_points.remove_at(spawn_index)
175
176     return points
177
178 func is_valid(candidate: Vector2, sample_region_size: Vector2,
    cell_size: float, radius: float, points: Array[Vector2], grid:
    Array[Array], grid_x_axis_size: int, grid_y_axis_size: int) ->
    bool:
179     if candidate.x >= 0 and candidate.x < sample_region_size.x and
        candidate.y >= 0 and candidate.y < sample_region_size.y:
180         var cell_x: int = roundi(candidate.x / cell_size)
181         var cell_y: int = roundi(candidate.y / cell_size)
182         var search_start_x: int = max(0, cell_x - 2)
183         var search_end_x: int = min(cell_x + 2, grid_x_axis_size - 1)
184         var search_start_y: int = max(0, cell_y - 2)
185         var search_end_y: int = min(cell_y + 2, grid_y_axis_size - 1)
186         for x in range(search_start_x, search_end_x):
187             for y in range(search_start_y, search_end_y):
188                 var point_index: int = grid[x][y] - 1
189                 if point_index != -1:

```

```

190         var distance: float = (candidate - points[
                point_index]).length_squared()
191         if distance < radius:
192             return false
193         return true
194     return false

```

C.4.6 accept_dialog.tscn

```

1  [gd_scene format=3 uid="uid://cau5jgogdnf53"]
2
3  [node name="AcceptDialog" type="AcceptDialog"]
4  title = "Tree-Munching Time!"
5  position = Vector2i(326, 100)
6  size = Vector2i(500, 421)
7  mouse_passthrough = true
8  ok_button_text = "Bring it on!"
9  dialog_text = "You're a hollow Golem who seeks the ultimate
    treasure; a ring that's got something on top of it. It's
    somewhere in this large village and barely visible to your
    naked eyes, but you'll stop at nothing to get what you want.
    You can chow down every tree and fauna that stands in your way
    of the ring, but your Achilles heel is any bricks and mortar,
    which will make you stop at your tracks. Are you ready to
    attain your treasure?w Golem in a black-and-white world, in
    search for your most desired treasure. It's a ring with
    something on top of it. And you'll stop at nothing to get what
    you want. You can chow down every tree and fauna that stands in
    your way of the ring, but your Achilles heel is any bricks and
    mortar, which will make you stop at your tracks. Are you ready
    to attain the treasure that is rightfully yours?!"
10 dialog_autowrap = true

```

C.4.7 win_dialog.tscn

```
1  [gd_scene format=3 uid="uid://b5q8ovcigrvyr"]
2
3  [node name="WinDialog" type="ConfirmationDialog"]
4  title = "You Found the Treasure!"
5  position = Vector2i(326, 100)
6  size = Vector2i(500, 421)
7  mouse_passthrough = true
8  ok_button_text = "Get Me a New Village"
9  dialog_text = "You found your treasure! Well done, you!"
10
11  Would you like to travel to a new village in the hopes of finding
    another ring? Or would you like to take your treasure home now?
    "
12  dialog_autowrap = true
13  cancel_button_text = "Get Me Out of Here"
```

C.4.8 icon.svg.import

```
1  [remap]
2
3  importer="texture"
4  type="CompressedTexture2D"
5  uid="uid://uotfe6soknht"
6  path="res://.godot/imported/icon.svg-218
    a8f2b3041327d8a5756f3a245f83b.ctex"
7  metadata={
8  "vram_texture": false
9  }
10
11  [deps]
```



```

12
13   source_file="res://icon.svg"
14   dest_files=["res://.godot/imported/icon.svg-218
           a8f2b3041327d8a5756f3a245f83b.ctex"]
15
16   [params]
17
18   compress/mode=0
19   compress/high_quality=false
20   compress/lossy_quality=0.7
21   compress/hdr_compression=1
22   compress/normal_map=0
23   compress/channel_pack=0
24   mipmaps/generate=false
25   mipmaps/limit=-1
26   roughness/mode=0
27   roughness/src_normal=""
28   process/fix_alpha_border=true
29   process/premult_alpha=false
30   process/normal_map_invert_y=false
31   process/hdr_as_srgb=false
32   process/hdr_clamp_exposure=false
33   process/size_limit=0
34   detect_3d/compress_to=1
35   svg/scale=1.0
36   editor/scale_with_editor_scale=false
37   editor/convert_colors_with_editor_theme=false

```

C.4.9 monochrome_packed.png.import

```

1   [remap]
2

```

```

3  importer="texture"
4  type="CompressedTexture2D"
5  uid="uid://c3bpsm4r8t504"
6  path="res://.godot/imported/monochrome_packed.png-6
    b9bd1c64dd50f72acd3afd14d1ac34f.ctex"
7  metadata={
8  "vram_texture": false
9  }
10
11  [deps]
12
13  source_file="res://monochrome_packed.png"
14  dest_files=["res://.godot/imported/monochrome_packed.png-6
    b9bd1c64dd50f72acd3afd14d1ac34f.ctex"]
15
16  [params]
17
18  compress/mode=0
19  compress/high_quality=false
20  compress/lossy_quality=0.7
21  compress/hdr_compression=1
22  compress/normal_map=0
23  compress/channel_pack=0
24  mipmaps/generate=false
25  mipmaps/limit=-1
26  roughness/mode=0
27  roughness/src_normal=""
28  process/fix_alpha_border=true
29  process/premult_alpha=false
30  process/normal_map_invert_y=false
31  process/hdr_as_srgb=false
32  process/hdr_clamp_exposure=false
33  process/size_limit=0

```

```
34 detect_3d/compress_to=1
```

C.4.10 LICENSE

```
1 This work is licensed under the Creative Commons Attribution-
  ShareAlike 4.0 International License. To view a copy of this
  license, visit http://creativecommons.org/licenses/by-sa/4.0/
  or send a letter to Creative Commons, PO Box 1866, Mountain
  View, CA 94042, USA.
```

C.5 GD4NoiseRPG

These are the source code listings for the Perlin/Simplex Noise implementation of the 2D tile-map RPG scenario in the Godot game engine. The link to the publicly available source code repository is here: <https://github.com/Zishan-Rahman/GD4NoiseRPG>

C.5.1 .gitattributes

```
1 # Normalize EOL for all files that Git considers text files.
2 * text=auto eol=lf
```

C.5.2 .gitignore

```
1 # Godot 4+ specific ignores
2 .godot/
```

C.5.3 project.godot

```
1 ; Engine configuration file.
```

```

2   ; It's best edited using the editor UI and not directly,
3   ; since the parameters that go here are not all obvious.
4   ;
5   ; Format:
6   ;   [section] ; section goes between []
7   ;   param=value ; assign values to parameters
8
9   config_version=5
10
11  [application]
12
13  config/name="Noise Demo"
14  run/main_scene="res://tile_map.tscn"
15  config/features=PackedStringArray("4.0", "Forward Plus")
16  config/icon="res://icon.svg"
17
18  [display]
19
20  window/size/viewport_height=640
21
22  [rendering]
23
24  environment/defaults/default_clear_color=Color(0, 0, 0, 1)

```

C.5.4 tile_map.tscn

```

1  [gd_scene load_steps=7 format=3 uid="uid://d4jdcavluwx6s"]
2
3  [ext_resource type="Texture2D" uid="uid://m662wwd4prmn" path="res
      ://monochrome_packed.png" id="1_ld7xx"]
4  [ext_resource type="Script" path="res://tile_map.gd" id="2_o1dn1"]
5  [ext_resource type="PackedScene" uid="uid://cau5jgogdnf53" path="

```

```

    res://accept_dialog.tscn" id="3_e0ur6"]
6  [ext_resource type="PackedScene" uid="uid://b5q8ovcigrvyr" path="
    res://win_dialog.tscn" id="4_ecfaa"]
7
8  [sub_resource type="TileSetAtlasSource" id="
    TileSetAtlasSource_1e80b"]
9  texture = ExtResource("1_ld7xx")
10  0:0/0 = 0
11  1:0/0 = 0
12  2:0/0 = 0
13  3:0/0 = 0
14  4:0/0 = 0
15  5:0/0 = 0
16  6:0/0 = 0
17  7:0/0 = 0
18  8:0/0 = 0
19  9:0/0 = 0
20  10:0/0 = 0
21  11:0/0 = 0
22  12:0/0 = 0
23  13:0/0 = 0
24  14:0/0 = 0
25  15:0/0 = 0
26  16:0/0 = 0
27  17:0/0 = 0
28  18:0/0 = 0
29  19:0/0 = 0
30  20:0/0 = 0
31  21:0/0 = 0
32  22:0/0 = 0
33  23:0/0 = 0
34  24:0/0 = 0
35  25:0/0 = 0

```

36 26:0/0 = 0
37 27:0/0 = 0
38 28:0/0 = 0
39 29:0/0 = 0
40 30:0/0 = 0
41 31:0/0 = 0
42 32:0/0 = 0
43 33:0/0 = 0
44 34:0/0 = 0
45 35:0/0 = 0
46 36:0/0 = 0
47 37:0/0 = 0
48 38:0/0 = 0
49 39:0/0 = 0
50 40:0/0 = 0
51 41:0/0 = 0
52 42:0/0 = 0
53 43:0/0 = 0
54 44:0/0 = 0
55 45:0/0 = 0
56 46:0/0 = 0
57 47:0/0 = 0
58 48:0/0 = 0
59 0:1/0 = 0
60 1:1/0 = 0
61 2:1/0 = 0
62 3:1/0 = 0
63 4:1/0 = 0
64 5:1/0 = 0
65 6:1/0 = 0
66 7:1/0 = 0
67 8:1/0 = 0
68 9:1/0 = 0

69 10:1/0 = 0
70 11:1/0 = 0
71 12:1/0 = 0
72 13:1/0 = 0
73 14:1/0 = 0
74 15:1/0 = 0
75 16:1/0 = 0
76 17:1/0 = 0
77 18:1/0 = 0
78 19:1/0 = 0
79 20:1/0 = 0
80 21:1/0 = 0
81 22:1/0 = 0
82 23:1/0 = 0
83 24:1/0 = 0
84 25:1/0 = 0
85 26:1/0 = 0
86 27:1/0 = 0
87 28:1/0 = 0
88 29:1/0 = 0
89 30:1/0 = 0
90 31:1/0 = 0
91 32:1/0 = 0
92 33:1/0 = 0
93 34:1/0 = 0
94 35:1/0 = 0
95 36:1/0 = 0
96 37:1/0 = 0
97 38:1/0 = 0
98 39:1/0 = 0
99 40:1/0 = 0
100 41:1/0 = 0
101 42:1/0 = 0

102	$43:1/0 = 0$
103	$44:1/0 = 0$
104	$45:1/0 = 0$
105	$46:1/0 = 0$
106	$47:1/0 = 0$
107	$48:1/0 = 0$
108	$0:2/0 = 0$
109	$1:2/0 = 0$
110	$2:2/0 = 0$
111	$3:2/0 = 0$
112	$4:2/0 = 0$
113	$5:2/0 = 0$
114	$6:2/0 = 0$
115	$7:2/0 = 0$
116	$8:2/0 = 0$
117	$9:2/0 = 0$
118	$10:2/0 = 0$
119	$11:2/0 = 0$
120	$12:2/0 = 0$
121	$13:2/0 = 0$
122	$14:2/0 = 0$
123	$15:2/0 = 0$
124	$16:2/0 = 0$
125	$17:2/0 = 0$
126	$18:2/0 = 0$
127	$19:2/0 = 0$
128	$20:2/0 = 0$
129	$21:2/0 = 0$
130	$22:2/0 = 0$
131	$23:2/0 = 0$
132	$24:2/0 = 0$
133	$25:2/0 = 0$
134	$26:2/0 = 0$

135 27:2/0 = 0
136 28:2/0 = 0
137 29:2/0 = 0
138 30:2/0 = 0
139 31:2/0 = 0
140 32:2/0 = 0
141 33:2/0 = 0
142 34:2/0 = 0
143 35:2/0 = 0
144 36:2/0 = 0
145 37:2/0 = 0
146 38:2/0 = 0
147 39:2/0 = 0
148 40:2/0 = 0
149 41:2/0 = 0
150 42:2/0 = 0
151 43:2/0 = 0
152 44:2/0 = 0
153 45:2/0 = 0
154 46:2/0 = 0
155 47:2/0 = 0
156 48:2/0 = 0
157 0:3/0 = 0
158 1:3/0 = 0
159 2:3/0 = 0
160 3:3/0 = 0
161 4:3/0 = 0
162 5:3/0 = 0
163 6:3/0 = 0
164 7:3/0 = 0
165 8:3/0 = 0
166 9:3/0 = 0
167 10:3/0 = 0

168	$11:3/0 = 0$
169	$12:3/0 = 0$
170	$13:3/0 = 0$
171	$14:3/0 = 0$
172	$15:3/0 = 0$
173	$16:3/0 = 0$
174	$17:3/0 = 0$
175	$18:3/0 = 0$
176	$19:3/0 = 0$
177	$20:3/0 = 0$
178	$21:3/0 = 0$
179	$22:3/0 = 0$
180	$23:3/0 = 0$
181	$24:3/0 = 0$
182	$25:3/0 = 0$
183	$26:3/0 = 0$
184	$27:3/0 = 0$
185	$28:3/0 = 0$
186	$29:3/0 = 0$
187	$30:3/0 = 0$
188	$31:3/0 = 0$
189	$32:3/0 = 0$
190	$33:3/0 = 0$
191	$34:3/0 = 0$
192	$35:3/0 = 0$
193	$36:3/0 = 0$
194	$37:3/0 = 0$
195	$38:3/0 = 0$
196	$39:3/0 = 0$
197	$40:3/0 = 0$
198	$41:3/0 = 0$
199	$42:3/0 = 0$
200	$43:3/0 = 0$

201	$44:3/0 = 0$
202	$45:3/0 = 0$
203	$46:3/0 = 0$
204	$47:3/0 = 0$
205	$48:3/0 = 0$
206	$0:4/0 = 0$
207	$1:4/0 = 0$
208	$2:4/0 = 0$
209	$3:4/0 = 0$
210	$4:4/0 = 0$
211	$5:4/0 = 0$
212	$6:4/0 = 0$
213	$7:4/0 = 0$
214	$8:4/0 = 0$
215	$9:4/0 = 0$
216	$10:4/0 = 0$
217	$11:4/0 = 0$
218	$12:4/0 = 0$
219	$13:4/0 = 0$
220	$14:4/0 = 0$
221	$15:4/0 = 0$
222	$16:4/0 = 0$
223	$17:4/0 = 0$
224	$18:4/0 = 0$
225	$19:4/0 = 0$
226	$20:4/0 = 0$
227	$21:4/0 = 0$
228	$22:4/0 = 0$
229	$23:4/0 = 0$
230	$24:4/0 = 0$
231	$25:4/0 = 0$
232	$26:4/0 = 0$
233	$27:4/0 = 0$

234	$28:4/0 = 0$
235	$29:4/0 = 0$
236	$30:4/0 = 0$
237	$31:4/0 = 0$
238	$32:4/0 = 0$
239	$33:4/0 = 0$
240	$34:4/0 = 0$
241	$35:4/0 = 0$
242	$36:4/0 = 0$
243	$37:4/0 = 0$
244	$38:4/0 = 0$
245	$39:4/0 = 0$
246	$40:4/0 = 0$
247	$41:4/0 = 0$
248	$42:4/0 = 0$
249	$43:4/0 = 0$
250	$44:4/0 = 0$
251	$45:4/0 = 0$
252	$46:4/0 = 0$
253	$47:4/0 = 0$
254	$48:4/0 = 0$
255	$0:5/0 = 0$
256	$1:5/0 = 0$
257	$2:5/0 = 0$
258	$3:5/0 = 0$
259	$4:5/0 = 0$
260	$5:5/0 = 0$
261	$6:5/0 = 0$
262	$7:5/0 = 0$
263	$8:5/0 = 0$
264	$9:5/0 = 0$
265	$10:5/0 = 0$
266	$11:5/0 = 0$

267	$12:5/0 = 0$
268	$13:5/0 = 0$
269	$14:5/0 = 0$
270	$15:5/0 = 0$
271	$16:5/0 = 0$
272	$17:5/0 = 0$
273	$18:5/0 = 0$
274	$19:5/0 = 0$
275	$20:5/0 = 0$
276	$21:5/0 = 0$
277	$22:5/0 = 0$
278	$23:5/0 = 0$
279	$24:5/0 = 0$
280	$25:5/0 = 0$
281	$26:5/0 = 0$
282	$27:5/0 = 0$
283	$28:5/0 = 0$
284	$29:5/0 = 0$
285	$30:5/0 = 0$
286	$31:5/0 = 0$
287	$32:5/0 = 0$
288	$33:5/0 = 0$
289	$34:5/0 = 0$
290	$35:5/0 = 0$
291	$36:5/0 = 0$
292	$37:5/0 = 0$
293	$38:5/0 = 0$
294	$39:5/0 = 0$
295	$40:5/0 = 0$
296	$41:5/0 = 0$
297	$42:5/0 = 0$
298	$43:5/0 = 0$
299	$44:5/0 = 0$

300	$45:5/0 = 0$
301	$46:5/0 = 0$
302	$47:5/0 = 0$
303	$48:5/0 = 0$
304	$0:6/0 = 0$
305	$1:6/0 = 0$
306	$2:6/0 = 0$
307	$3:6/0 = 0$
308	$4:6/0 = 0$
309	$5:6/0 = 0$
310	$6:6/0 = 0$
311	$7:6/0 = 0$
312	$8:6/0 = 0$
313	$9:6/0 = 0$
314	$10:6/0 = 0$
315	$11:6/0 = 0$
316	$12:6/0 = 0$
317	$13:6/0 = 0$
318	$14:6/0 = 0$
319	$15:6/0 = 0$
320	$16:6/0 = 0$
321	$17:6/0 = 0$
322	$18:6/0 = 0$
323	$19:6/0 = 0$
324	$20:6/0 = 0$
325	$21:6/0 = 0$
326	$22:6/0 = 0$
327	$23:6/0 = 0$
328	$24:6/0 = 0$
329	$25:6/0 = 0$
330	$26:6/0 = 0$
331	$27:6/0 = 0$
332	$28:6/0 = 0$

333 29:6/0 = 0
334 30:6/0 = 0
335 31:6/0 = 0
336 32:6/0 = 0
337 33:6/0 = 0
338 34:6/0 = 0
339 35:6/0 = 0
340 36:6/0 = 0
341 37:6/0 = 0
342 38:6/0 = 0
343 39:6/0 = 0
344 40:6/0 = 0
345 41:6/0 = 0
346 42:6/0 = 0
347 43:6/0 = 0
348 44:6/0 = 0
349 45:6/0 = 0
350 46:6/0 = 0
351 47:6/0 = 0
352 48:6/0 = 0
353 0:7/0 = 0
354 1:7/0 = 0
355 2:7/0 = 0
356 3:7/0 = 0
357 4:7/0 = 0
358 5:7/0 = 0
359 6:7/0 = 0
360 7:7/0 = 0
361 8:7/0 = 0
362 9:7/0 = 0
363 10:7/0 = 0
364 11:7/0 = 0
365 12:7/0 = 0

366 13:7/0 = 0
367 14:7/0 = 0
368 15:7/0 = 0
369 16:7/0 = 0
370 17:7/0 = 0
371 18:7/0 = 0
372 19:7/0 = 0
373 20:7/0 = 0
374 21:7/0 = 0
375 22:7/0 = 0
376 23:7/0 = 0
377 24:7/0 = 0
378 25:7/0 = 0
379 26:7/0 = 0
380 27:7/0 = 0
381 28:7/0 = 0
382 29:7/0 = 0
383 30:7/0 = 0
384 31:7/0 = 0
385 32:7/0 = 0
386 33:7/0 = 0
387 34:7/0 = 0
388 35:7/0 = 0
389 36:7/0 = 0
390 37:7/0 = 0
391 38:7/0 = 0
392 39:7/0 = 0
393 40:7/0 = 0
394 41:7/0 = 0
395 42:7/0 = 0
396 43:7/0 = 0
397 44:7/0 = 0
398 45:7/0 = 0

399	$46:7/0 = 0$
400	$47:7/0 = 0$
401	$48:7/0 = 0$
402	$0:8/0 = 0$
403	$1:8/0 = 0$
404	$2:8/0 = 0$
405	$3:8/0 = 0$
406	$4:8/0 = 0$
407	$5:8/0 = 0$
408	$6:8/0 = 0$
409	$7:8/0 = 0$
410	$8:8/0 = 0$
411	$9:8/0 = 0$
412	$10:8/0 = 0$
413	$11:8/0 = 0$
414	$12:8/0 = 0$
415	$13:8/0 = 0$
416	$14:8/0 = 0$
417	$15:8/0 = 0$
418	$16:8/0 = 0$
419	$17:8/0 = 0$
420	$18:8/0 = 0$
421	$19:8/0 = 0$
422	$20:8/0 = 0$
423	$21:8/0 = 0$
424	$22:8/0 = 0$
425	$23:8/0 = 0$
426	$24:8/0 = 0$
427	$25:8/0 = 0$
428	$26:8/0 = 0$
429	$27:8/0 = 0$
430	$28:8/0 = 0$
431	$29:8/0 = 0$

432 30:8/0 = 0
433 31:8/0 = 0
434 32:8/0 = 0
435 33:8/0 = 0
436 34:8/0 = 0
437 35:8/0 = 0
438 36:8/0 = 0
439 37:8/0 = 0
440 38:8/0 = 0
441 39:8/0 = 0
442 40:8/0 = 0
443 41:8/0 = 0
444 42:8/0 = 0
445 43:8/0 = 0
446 44:8/0 = 0
447 45:8/0 = 0
448 46:8/0 = 0
449 47:8/0 = 0
450 48:8/0 = 0
451 0:9/0 = 0
452 1:9/0 = 0
453 2:9/0 = 0
454 3:9/0 = 0
455 4:9/0 = 0
456 5:9/0 = 0
457 6:9/0 = 0
458 7:9/0 = 0
459 8:9/0 = 0
460 9:9/0 = 0
461 10:9/0 = 0
462 11:9/0 = 0
463 12:9/0 = 0
464 13:9/0 = 0

465 14:9/0 = 0
466 15:9/0 = 0
467 16:9/0 = 0
468 17:9/0 = 0
469 18:9/0 = 0
470 19:9/0 = 0
471 20:9/0 = 0
472 21:9/0 = 0
473 22:9/0 = 0
474 23:9/0 = 0
475 24:9/0 = 0
476 25:9/0 = 0
477 26:9/0 = 0
478 27:9/0 = 0
479 28:9/0 = 0
480 29:9/0 = 0
481 30:9/0 = 0
482 31:9/0 = 0
483 32:9/0 = 0
484 33:9/0 = 0
485 34:9/0 = 0
486 35:9/0 = 0
487 36:9/0 = 0
488 37:9/0 = 0
489 38:9/0 = 0
490 39:9/0 = 0
491 40:9/0 = 0
492 41:9/0 = 0
493 42:9/0 = 0
494 43:9/0 = 0
495 44:9/0 = 0
496 45:9/0 = 0
497 46:9/0 = 0

498 47:9/0 = 0
499 48:9/0 = 0
500 0:10/0 = 0
501 1:10/0 = 0
502 2:10/0 = 0
503 3:10/0 = 0
504 4:10/0 = 0
505 5:10/0 = 0
506 6:10/0 = 0
507 7:10/0 = 0
508 8:10/0 = 0
509 9:10/0 = 0
510 10:10/0 = 0
511 11:10/0 = 0
512 12:10/0 = 0
513 13:10/0 = 0
514 14:10/0 = 0
515 15:10/0 = 0
516 16:10/0 = 0
517 17:10/0 = 0
518 18:10/0 = 0
519 19:10/0 = 0
520 20:10/0 = 0
521 21:10/0 = 0
522 22:10/0 = 0
523 23:10/0 = 0
524 24:10/0 = 0
525 25:10/0 = 0
526 26:10/0 = 0
527 27:10/0 = 0
528 28:10/0 = 0
529 29:10/0 = 0
530 30:10/0 = 0

531 31:10/0 = 0
532 32:10/0 = 0
533 33:10/0 = 0
534 34:10/0 = 0
535 35:10/0 = 0
536 36:10/0 = 0
537 37:10/0 = 0
538 38:10/0 = 0
539 39:10/0 = 0
540 40:10/0 = 0
541 41:10/0 = 0
542 42:10/0 = 0
543 43:10/0 = 0
544 44:10/0 = 0
545 45:10/0 = 0
546 46:10/0 = 0
547 47:10/0 = 0
548 48:10/0 = 0
549 0:11/0 = 0
550 1:11/0 = 0
551 2:11/0 = 0
552 3:11/0 = 0
553 4:11/0 = 0
554 5:11/0 = 0
555 6:11/0 = 0
556 7:11/0 = 0
557 8:11/0 = 0
558 9:11/0 = 0
559 10:11/0 = 0
560 11:11/0 = 0
561 12:11/0 = 0
562 13:11/0 = 0
563 14:11/0 = 0

564 15:11/0 = 0
565 16:11/0 = 0
566 17:11/0 = 0
567 18:11/0 = 0
568 19:11/0 = 0
569 20:11/0 = 0
570 21:11/0 = 0
571 22:11/0 = 0
572 23:11/0 = 0
573 24:11/0 = 0
574 25:11/0 = 0
575 26:11/0 = 0
576 27:11/0 = 0
577 28:11/0 = 0
578 29:11/0 = 0
579 30:11/0 = 0
580 31:11/0 = 0
581 32:11/0 = 0
582 33:11/0 = 0
583 34:11/0 = 0
584 35:11/0 = 0
585 36:11/0 = 0
586 37:11/0 = 0
587 38:11/0 = 0
588 39:11/0 = 0
589 40:11/0 = 0
590 41:11/0 = 0
591 42:11/0 = 0
592 43:11/0 = 0
593 44:11/0 = 0
594 45:11/0 = 0
595 46:11/0 = 0
596 47:11/0 = 0

597 48:11/0 = 0
598 0:12/0 = 0
599 1:12/0 = 0
600 2:12/0 = 0
601 3:12/0 = 0
602 4:12/0 = 0
603 5:12/0 = 0
604 6:12/0 = 0
605 7:12/0 = 0
606 8:12/0 = 0
607 9:12/0 = 0
608 10:12/0 = 0
609 11:12/0 = 0
610 12:12/0 = 0
611 13:12/0 = 0
612 14:12/0 = 0
613 15:12/0 = 0
614 16:12/0 = 0
615 17:12/0 = 0
616 18:12/0 = 0
617 19:12/0 = 0
618 20:12/0 = 0
619 21:12/0 = 0
620 22:12/0 = 0
621 23:12/0 = 0
622 24:12/0 = 0
623 25:12/0 = 0
624 26:12/0 = 0
625 27:12/0 = 0
626 28:12/0 = 0
627 29:12/0 = 0
628 30:12/0 = 0
629 31:12/0 = 0

630 32:12/0 = 0
631 33:12/0 = 0
632 34:12/0 = 0
633 35:12/0 = 0
634 36:12/0 = 0
635 37:12/0 = 0
636 38:12/0 = 0
637 39:12/0 = 0
638 40:12/0 = 0
639 41:12/0 = 0
640 42:12/0 = 0
641 43:12/0 = 0
642 44:12/0 = 0
643 45:12/0 = 0
644 46:12/0 = 0
645 47:12/0 = 0
646 48:12/0 = 0
647 0:13/0 = 0
648 1:13/0 = 0
649 2:13/0 = 0
650 3:13/0 = 0
651 4:13/0 = 0
652 5:13/0 = 0
653 6:13/0 = 0
654 7:13/0 = 0
655 8:13/0 = 0
656 9:13/0 = 0
657 10:13/0 = 0
658 11:13/0 = 0
659 12:13/0 = 0
660 13:13/0 = 0
661 14:13/0 = 0
662 15:13/0 = 0

663 16:13/0 = 0
664 17:13/0 = 0
665 18:13/0 = 0
666 19:13/0 = 0
667 20:13/0 = 0
668 21:13/0 = 0
669 22:13/0 = 0
670 23:13/0 = 0
671 24:13/0 = 0
672 25:13/0 = 0
673 26:13/0 = 0
674 27:13/0 = 0
675 28:13/0 = 0
676 29:13/0 = 0
677 30:13/0 = 0
678 31:13/0 = 0
679 32:13/0 = 0
680 33:13/0 = 0
681 34:13/0 = 0
682 35:13/0 = 0
683 36:13/0 = 0
684 37:13/0 = 0
685 38:13/0 = 0
686 39:13/0 = 0
687 40:13/0 = 0
688 41:13/0 = 0
689 42:13/0 = 0
690 43:13/0 = 0
691 44:13/0 = 0
692 45:13/0 = 0
693 46:13/0 = 0
694 47:13/0 = 0
695 48:13/0 = 0

696	$0:14/0 = 0$
697	$1:14/0 = 0$
698	$2:14/0 = 0$
699	$3:14/0 = 0$
700	$4:14/0 = 0$
701	$5:14/0 = 0$
702	$6:14/0 = 0$
703	$7:14/0 = 0$
704	$8:14/0 = 0$
705	$9:14/0 = 0$
706	$10:14/0 = 0$
707	$11:14/0 = 0$
708	$12:14/0 = 0$
709	$13:14/0 = 0$
710	$14:14/0 = 0$
711	$15:14/0 = 0$
712	$16:14/0 = 0$
713	$17:14/0 = 0$
714	$18:14/0 = 0$
715	$19:14/0 = 0$
716	$20:14/0 = 0$
717	$21:14/0 = 0$
718	$22:14/0 = 0$
719	$23:14/0 = 0$
720	$24:14/0 = 0$
721	$25:14/0 = 0$
722	$26:14/0 = 0$
723	$27:14/0 = 0$
724	$28:14/0 = 0$
725	$29:14/0 = 0$
726	$30:14/0 = 0$
727	$31:14/0 = 0$
728	$32:14/0 = 0$

729 33:14/0 = 0
730 34:14/0 = 0
731 35:14/0 = 0
732 36:14/0 = 0
733 37:14/0 = 0
734 38:14/0 = 0
735 39:14/0 = 0
736 40:14/0 = 0
737 41:14/0 = 0
738 42:14/0 = 0
739 43:14/0 = 0
740 44:14/0 = 0
741 45:14/0 = 0
742 46:14/0 = 0
743 47:14/0 = 0
744 48:14/0 = 0
745 0:15/0 = 0
746 1:15/0 = 0
747 2:15/0 = 0
748 3:15/0 = 0
749 4:15/0 = 0
750 5:15/0 = 0
751 6:15/0 = 0
752 7:15/0 = 0
753 8:15/0 = 0
754 9:15/0 = 0
755 10:15/0 = 0
756 11:15/0 = 0
757 12:15/0 = 0
758 13:15/0 = 0
759 14:15/0 = 0
760 15:15/0 = 0
761 16:15/0 = 0

762	$17:15/0 = 0$
763	$18:15/0 = 0$
764	$19:15/0 = 0$
765	$20:15/0 = 0$
766	$21:15/0 = 0$
767	$22:15/0 = 0$
768	$23:15/0 = 0$
769	$24:15/0 = 0$
770	$25:15/0 = 0$
771	$26:15/0 = 0$
772	$27:15/0 = 0$
773	$28:15/0 = 0$
774	$29:15/0 = 0$
775	$30:15/0 = 0$
776	$31:15/0 = 0$
777	$32:15/0 = 0$
778	$33:15/0 = 0$
779	$34:15/0 = 0$
780	$35:15/0 = 0$
781	$36:15/0 = 0$
782	$37:15/0 = 0$
783	$38:15/0 = 0$
784	$39:15/0 = 0$
785	$40:15/0 = 0$
786	$41:15/0 = 0$
787	$42:15/0 = 0$
788	$43:15/0 = 0$
789	$44:15/0 = 0$
790	$45:15/0 = 0$
791	$46:15/0 = 0$
792	$47:15/0 = 0$
793	$48:15/0 = 0$
794	$0:16/0 = 0$

795	$1:16/0 = 0$
796	$2:16/0 = 0$
797	$3:16/0 = 0$
798	$4:16/0 = 0$
799	$5:16/0 = 0$
800	$6:16/0 = 0$
801	$7:16/0 = 0$
802	$8:16/0 = 0$
803	$9:16/0 = 0$
804	$10:16/0 = 0$
805	$11:16/0 = 0$
806	$12:16/0 = 0$
807	$13:16/0 = 0$
808	$14:16/0 = 0$
809	$15:16/0 = 0$
810	$16:16/0 = 0$
811	$17:16/0 = 0$
812	$18:16/0 = 0$
813	$19:16/0 = 0$
814	$20:16/0 = 0$
815	$21:16/0 = 0$
816	$22:16/0 = 0$
817	$23:16/0 = 0$
818	$24:16/0 = 0$
819	$25:16/0 = 0$
820	$26:16/0 = 0$
821	$27:16/0 = 0$
822	$28:16/0 = 0$
823	$29:16/0 = 0$
824	$30:16/0 = 0$
825	$31:16/0 = 0$
826	$32:16/0 = 0$
827	$33:16/0 = 0$

828	$34:16/0 = 0$
829	$35:16/0 = 0$
830	$36:16/0 = 0$
831	$37:16/0 = 0$
832	$38:16/0 = 0$
833	$39:16/0 = 0$
834	$40:16/0 = 0$
835	$41:16/0 = 0$
836	$42:16/0 = 0$
837	$43:16/0 = 0$
838	$44:16/0 = 0$
839	$45:16/0 = 0$
840	$46:16/0 = 0$
841	$47:16/0 = 0$
842	$48:16/0 = 0$
843	$0:17/0 = 0$
844	$1:17/0 = 0$
845	$2:17/0 = 0$
846	$3:17/0 = 0$
847	$4:17/0 = 0$
848	$5:17/0 = 0$
849	$6:17/0 = 0$
850	$7:17/0 = 0$
851	$8:17/0 = 0$
852	$9:17/0 = 0$
853	$10:17/0 = 0$
854	$11:17/0 = 0$
855	$12:17/0 = 0$
856	$13:17/0 = 0$
857	$14:17/0 = 0$
858	$15:17/0 = 0$
859	$16:17/0 = 0$
860	$17:17/0 = 0$

861 18:17/0 = 0
862 19:17/0 = 0
863 20:17/0 = 0
864 21:17/0 = 0
865 22:17/0 = 0
866 23:17/0 = 0
867 24:17/0 = 0
868 25:17/0 = 0
869 26:17/0 = 0
870 27:17/0 = 0
871 28:17/0 = 0
872 29:17/0 = 0
873 30:17/0 = 0
874 31:17/0 = 0
875 32:17/0 = 0
876 33:17/0 = 0
877 34:17/0 = 0
878 35:17/0 = 0
879 36:17/0 = 0
880 37:17/0 = 0
881 38:17/0 = 0
882 39:17/0 = 0
883 40:17/0 = 0
884 41:17/0 = 0
885 42:17/0 = 0
886 43:17/0 = 0
887 44:17/0 = 0
888 45:17/0 = 0
889 46:17/0 = 0
890 47:17/0 = 0
891 48:17/0 = 0
892 0:18/0 = 0
893 1:18/0 = 0

894 2:18/0 = 0
895 3:18/0 = 0
896 4:18/0 = 0
897 5:18/0 = 0
898 6:18/0 = 0
899 7:18/0 = 0
900 8:18/0 = 0
901 9:18/0 = 0
902 10:18/0 = 0
903 11:18/0 = 0
904 12:18/0 = 0
905 13:18/0 = 0
906 14:18/0 = 0
907 15:18/0 = 0
908 16:18/0 = 0
909 17:18/0 = 0
910 18:18/0 = 0
911 19:18/0 = 0
912 20:18/0 = 0
913 21:18/0 = 0
914 22:18/0 = 0
915 23:18/0 = 0
916 24:18/0 = 0
917 25:18/0 = 0
918 26:18/0 = 0
919 27:18/0 = 0
920 28:18/0 = 0
921 29:18/0 = 0
922 30:18/0 = 0
923 31:18/0 = 0
924 32:18/0 = 0
925 33:18/0 = 0
926 34:18/0 = 0

927 35:18/0 = 0
928 36:18/0 = 0
929 37:18/0 = 0
930 38:18/0 = 0
931 39:18/0 = 0
932 40:18/0 = 0
933 41:18/0 = 0
934 42:18/0 = 0
935 43:18/0 = 0
936 44:18/0 = 0
937 45:18/0 = 0
938 46:18/0 = 0
939 47:18/0 = 0
940 48:18/0 = 0
941 0:19/0 = 0
942 1:19/0 = 0
943 2:19/0 = 0
944 3:19/0 = 0
945 4:19/0 = 0
946 5:19/0 = 0
947 6:19/0 = 0
948 7:19/0 = 0
949 8:19/0 = 0
950 9:19/0 = 0
951 10:19/0 = 0
952 11:19/0 = 0
953 12:19/0 = 0
954 13:19/0 = 0
955 14:19/0 = 0
956 15:19/0 = 0
957 16:19/0 = 0
958 17:19/0 = 0
959 18:19/0 = 0

960 19:19/0 = 0
961 20:19/0 = 0
962 21:19/0 = 0
963 22:19/0 = 0
964 23:19/0 = 0
965 24:19/0 = 0
966 25:19/0 = 0
967 26:19/0 = 0
968 27:19/0 = 0
969 28:19/0 = 0
970 29:19/0 = 0
971 30:19/0 = 0
972 31:19/0 = 0
973 32:19/0 = 0
974 33:19/0 = 0
975 34:19/0 = 0
976 35:19/0 = 0
977 36:19/0 = 0
978 37:19/0 = 0
979 38:19/0 = 0
980 39:19/0 = 0
981 40:19/0 = 0
982 41:19/0 = 0
983 42:19/0 = 0
984 43:19/0 = 0
985 44:19/0 = 0
986 45:19/0 = 0
987 46:19/0 = 0
988 47:19/0 = 0
989 48:19/0 = 0
990 0:20/0 = 0
991 1:20/0 = 0
992 2:20/0 = 0

993	$3:20/0 = 0$
994	$4:20/0 = 0$
995	$5:20/0 = 0$
996	$6:20/0 = 0$
997	$7:20/0 = 0$
998	$8:20/0 = 0$
999	$9:20/0 = 0$
1000	$10:20/0 = 0$
1001	$11:20/0 = 0$
1002	$12:20/0 = 0$
1003	$13:20/0 = 0$
1004	$14:20/0 = 0$
1005	$15:20/0 = 0$
1006	$16:20/0 = 0$
1007	$17:20/0 = 0$
1008	$18:20/0 = 0$
1009	$19:20/0 = 0$
1010	$20:20/0 = 0$
1011	$21:20/0 = 0$
1012	$22:20/0 = 0$
1013	$23:20/0 = 0$
1014	$24:20/0 = 0$
1015	$25:20/0 = 0$
1016	$26:20/0 = 0$
1017	$27:20/0 = 0$
1018	$28:20/0 = 0$
1019	$29:20/0 = 0$
1020	$30:20/0 = 0$
1021	$31:20/0 = 0$
1022	$32:20/0 = 0$
1023	$33:20/0 = 0$
1024	$34:20/0 = 0$
1025	$35:20/0 = 0$

1026	$36:20/0 = 0$
1027	$37:20/0 = 0$
1028	$38:20/0 = 0$
1029	$39:20/0 = 0$
1030	$40:20/0 = 0$
1031	$41:20/0 = 0$
1032	$42:20/0 = 0$
1033	$43:20/0 = 0$
1034	$44:20/0 = 0$
1035	$45:20/0 = 0$
1036	$46:20/0 = 0$
1037	$47:20/0 = 0$
1038	$48:20/0 = 0$
1039	$0:21/0 = 0$
1040	$1:21/0 = 0$
1041	$2:21/0 = 0$
1042	$3:21/0 = 0$
1043	$4:21/0 = 0$
1044	$5:21/0 = 0$
1045	$6:21/0 = 0$
1046	$7:21/0 = 0$
1047	$8:21/0 = 0$
1048	$9:21/0 = 0$
1049	$10:21/0 = 0$
1050	$11:21/0 = 0$
1051	$12:21/0 = 0$
1052	$13:21/0 = 0$
1053	$14:21/0 = 0$
1054	$15:21/0 = 0$
1055	$16:21/0 = 0$
1056	$17:21/0 = 0$
1057	$18:21/0 = 0$
1058	$19:21/0 = 0$

```

1059    20:21/0 = 0
1060    21:21/0 = 0
1061    22:21/0 = 0
1062    23:21/0 = 0
1063    24:21/0 = 0
1064    25:21/0 = 0
1065    26:21/0 = 0
1066    27:21/0 = 0
1067    28:21/0 = 0
1068    29:21/0 = 0
1069    30:21/0 = 0
1070    31:21/0 = 0
1071    32:21/0 = 0
1072    33:21/0 = 0
1073    34:21/0 = 0
1074    35:21/0 = 0
1075    36:21/0 = 0
1076    37:21/0 = 0
1077    38:21/0 = 0
1078    39:21/0 = 0
1079    40:21/0 = 0
1080    41:21/0 = 0
1081    42:21/0 = 0
1082    43:21/0 = 0
1083    44:21/0 = 0
1084    45:21/0 = 0
1085    46:21/0 = 0
1086    47:21/0 = 0
1087    48:21/0 = 0
1088
1089    [sub_resource type="TileSet" id="TileSet_qtrb6"]
1090    sources/0 = SubResource("TileSetAtlasSource_1e80b")
1091

```

```

1092 [node name="TileMap" type="TileMap"]
1093 texture_filter = 1
1094 tile_set = SubResource("TileSet_qtrb6")
1095 format = 2
1096 script = ExtResource("2_o1dn1")
1097
1098 [node name="AcceptDialog" parent="." instance=ExtResource("3_e0ur6"
    )]
1099
1100 [node name="WinDialog" parent="." instance=ExtResource("4_ecfaa")]
1101 title = "You Found the Treasure!"

```

C.5.5 tile_map.gd

```

1 extends TileMap
2
3 const buildings: Array[Vector2i] = [
4     Vector2i(0, 19),
5     Vector2i(1, 19),
6     Vector2i(2, 19),
7     Vector2i(3, 19),
8     Vector2i(4, 19),
9     Vector2i(5, 19),
10    Vector2i(6, 19),
11    Vector2i(7, 19),
12    Vector2i(8, 20),
13    Vector2i(0, 20),
14    Vector2i(1, 20),
15    Vector2i(2, 20),
16    Vector2i(3, 20),
17    Vector2i(4, 20),
18    Vector2i(5, 20),

```

```

19     Vector2i(6, 20),
20     Vector2i(7, 20),
21     Vector2i(8, 20),
22     Vector2i(0, 21),
23     Vector2i(1, 21),
24     Vector2i(2, 21),
25     Vector2i(3, 21),
26     Vector2i(4, 21),
27     Vector2i(5, 21),
28     Vector2i(6, 21),
29     Vector2i(7, 21),
30     Vector2i(8, 21)
31 ]
32 const trees: Array[Vector2i] = [
33     Vector2i(0,1),
34     Vector2i(1,1),
35     Vector2i(2,1),
36     Vector2i(3,1),
37     Vector2i(4,1),
38     Vector2i(5,1),
39     Vector2i(6,1),
40     Vector2i(7,1),
41     Vector2i(0,2),
42     Vector2i(1,2),
43     Vector2i(2,2),
44     Vector2i(3,2),
45     Vector2i(4,2)
46 ]
47 const PLAYER_SPRITE: Vector2i = Vector2i(24, 7)
48 var player_placement_cell: Vector2i
49 const rings: Array[Vector2i] = [
50     Vector2i(43, 6),
51     Vector2i(44, 6),

```

```

52     Vector2i(45, 6),
53     Vector2i(46, 6)
54 ]
55 var ring_placement_cell: Vector2i
56
57 var noise: FastNoiseLite
58 ## Defines the type of noise generation algorithm to use. Equates
    to the noise_type property in FastNoiseLite.
59 @export_enum("Perlin", "Simplex", "Simplex Smooth", "Value", "Value
    Cubic") var noise_type: String = "Simplex Smooth"
60 ## Defines the type of method used to combine octaves of a noise
    image into a fractal. Directly equates to the FractalType enum
    in FastNoiseLite.
61 @export var fractal_type: FastNoiseLite.FractalType
62 ## Defines the function used to calculate the distance between the
    nearest/second-nearest point(s). Directly equates to the
    CellularDistanceFunction enum in FastNoiseLite.
63 @export var cellular_distance_type: FastNoiseLite.
    CellularDistanceFunction
64 ### Defines the number of noise octaves to use in the generated
    image.
65 #@export_range(1, 10, 1) var octaves: int = 5
66 ## Defines the frequency of the generated noise, the higher the
    frequency, the rougher and more granular the noise.
67 @export_range(0.0, 1.0) var noise_frequency: float = 0.894
68
69 ## Defines the upper limit to set for painting a tree tile on a
    specific noise pixel. If the value returned by the get_noise_2d
    method (in FastNoiseLite) is smaller than this, then it gets
    painted.
70 @export_range(-1.0, 1.0) var tree_cap: float = -0.048
71 ## Defines the upper limit to set for painting a building tile on a
    specific noise pixel. If the value returned by the

```



```

    get_noise_2d method (in FastNoiseLite) is smaller than this,
    then it gets painted. If the value of building_cap is smaller
    than tree_cap, then decide whether or not to paint a building
    cell there with building_overtakes_tree.
72 @export_range(-1.0, 1.0) var building_cap: float = -0.252
73 ## Only used when building_cap is smaller than tree_cap. Determines
    the probability that a building tile would be painted in a
    cell where a tree tile was, or could be, also painted. Whether
    or not the cell actually is painted over is decided on
    computation time.
74 @export_range(0.0, 0.5) var building_overtakes_tree: float = 0.12
75 var x_tile_range: int = ProjectSettings.get_setting("display/window
    /size/viewport_width") / tile_set.tile_size.x
76 var y_tile_range: int = ProjectSettings.get_setting("display/window
    /size/viewport_height") / tile_set.tile_size.y
77
78 # Called when the node enters the scene tree for the first time.
79 func _ready() -> void:
80     randomize()
81     var start_time: float = Time.get_ticks_msec()
82     set_noise()
83     paint_tiles()
84     place_player()
85     place_ring()
86     var new_time: float = Time.get_ticks_msec() - start_time
87     print("Time taken: " + str(new_time) + "ms")
88     $AcceptDialog.dialog_text = "You're a hollow Golem who seeks the
        ultimate treasure; a ring that's got something on top of it
        . It's somewhere in this large village and barely visible to
        your naked eyes, which took us " + str(new_time) + "
        milliseconds to generate (" + str(new_time / 1000.0) + "
        seconds), but you'll stop at nothing to get what you want.
        You can chow down every tree and fauna that stands in your

```

```

way of the ring, but your Achilles heel is any bricks and
mortar, which WILL make you stop at your tracks. Since it's
easy to get lost in here, we'll tell you that you're in
position " + str(player_placement_cell) + " in this big
village of size " + str(Vector2i(x_tile_range, y_tile_range)
) + ". However, it is YOUR job to find the ring, so are you
ready to attain the treasure that is rightfully yours?!"

89     $AcceptDialog.visible = true
90     $AcceptDialog.confirmed.connect(_on_AcceptDialog_closed)
91     $AcceptDialog.canceled.connect(_on_AcceptDialog_closed)
92     $WinDialog.confirmed.connect(_on_WinDialog_confirmed)
93     $WinDialog.canceled.connect(_on_WinDialog_canceled)
94     get_tree().paused = true
95
96     func _on_WinDialog_confirmed() -> void:
97         get_tree().reload_current_scene()
98
99     func _on_WinDialog_canceled() -> void:
100         get_tree().quit()
101
102     func _on_AcceptDialog_closed() -> void:
103         $AcceptDialog.visible = false
104         get_tree().paused = false
105
106     func _get_random_placement_cell() -> Vector2i:
107         return Vector2i(randi() % x_tile_range, randi() % y_tile_range)
108
109     func place_player() -> void:
110         player_placement_cell = _get_random_placement_cell()
111         while get_used_cells(0).has(player_placement_cell):
112             player_placement_cell = _get_random_placement_cell()
113         set_cell(0, player_placement_cell, 0, PLAYER_SPRITE)
114

```

```

115 func place_ring() -> void:
116     ring_placement_cell = _get_random_placement_cell()
117     while get_used_cells(0).has(ring_placement_cell):
118         ring_placement_cell = _get_random_placement_cell()
119     set_cell(0, ring_placement_cell, 0, rings.pick_random())
120
121 func _is_not_out_of_bounds(cell: Vector2i) -> bool:
122     return cell.x >= 0 and cell.x < x_tile_range and cell.y >= 0 and
        cell.y < y_tile_range
123
124 func _physics_process(_delta: float) -> void:
125     var previous_cell: Vector2i = player_placement_cell
126     var direction: Vector2i = Vector2i.ZERO
127     if Input.is_action_pressed("ui_up"): direction = Vector2i.UP
128     elif Input.is_action_pressed("ui_down"): direction = Vector2i.
        DOWN
129     elif Input.is_action_pressed("ui_left"): direction = Vector2i.
        LEFT
130     elif Input.is_action_pressed("ui_right"): direction = Vector2i.
        RIGHT
131     var new_placement_cell: Vector2i = player_placement_cell +
        direction
132     if (not get_used_cells(0).has(new_placement_cell) or trees.has(
        get_cell_atlas_coords(0, new_placement_cell)) or
        new_placement_cell == ring_placement_cell) and
        _is_not_out_of_bounds(new_placement_cell):
133         player_placement_cell = new_placement_cell
134         set_cell(0, previous_cell, 0) # deletes contents of previous
            cell (atlas_coords = Vector2i(-1, -1))
135         set_cell(0, player_placement_cell, 0, PLAYER_SPRITE)
136         if player_placement_cell == ring_placement_cell:
137             $WinDialog.visible = true
138             get_tree().paused = true

```

```

139
140 # ALGORITHM BEGINS HERE
141
142 func _get_noise_type() -> int:
143     match noise_type:
144         "Perlin": return 3
145         "Simplex": return 0
146         "Value": return 5
147         "Value Cubic": return 4
148         _: return 1 # Return Simplex Smooth by default
149
150 func set_noise() -> void:
151     noise = FastNoiseLite.new()
152     noise.frequency = noise_frequency
153     noise.noise_type = _get_noise_type() as FastNoiseLite.NoiseType
154     noise.fractal_type = fractal_type
155     noise.cellular_distance_function = cellular_distance_type
156     # noise.fractal_octaves = octaves
157     noise.seed = randi()
158
159 # I took inspiration from a Godot 3.1 tutorial: https://youtu.be/SBDs8hbs43w
160 # However, no code was taken or adapted in any way, shape or form.
161
162 func paint_tiles() -> void:
163     for x in range(x_tile_range):
164         for y in range(y_tile_range):
165             var noise_point: float = noise.get_noise_2d(x * tile_set.
                tile_size.x, y * tile_set.tile_size.y)
166             if noise_point < tree_cap and not get_used_cells(0).has(
                Vector2i(x, y)):
167                 set_cell(0, Vector2i(x, y), 0, trees.pick_random())
168             if ((building_cap <= tree_cap and randf() <

```

```

        building_overtakes_tree) or (building_cap > tree_cap
        and noise_point < building_cap)) and not
        get_used_cells(0).has(Vector2i(x, y)):
169         set_cell(0, Vector2i(x, y), 0, buildings.pick_random())

```

C.5.6 accept_dialog.tscn

```

1  [gd_scene format=3 uid="uid://cau5jgogdnf53"]
2
3  [node name="AcceptDialog" type="AcceptDialog"]
4  title = "Tree-Munching Time!"
5  position = Vector2i(326, 100)
6  size = Vector2i(500, 421)
7  mouse_passthrough = true
8  ok_button_text = "Bring it on!"
9  dialog_text = "You're a hollow Golem who seeks the ultimate
    treasure; a ring that's got something on top of it. It's
    somewhere in this large village and barely visible to your
    naked eyes, but you'll stop at nothing to get what you want.
    You can chow down every tree and fauna that stands in your way
    of the ring, but your Achilles heel is any bricks and mortar,
    which will make you stop at your tracks. Are you ready to
    attain your treasure?w Golem in a black-and-white world, in
    search for your most desired treasure. It's a ring with
    something on top of it. And you'll stop at nothing to get what
    you want. You can chow down every tree and fauna that stands in
    your way of the ring, but your Achilles heel is any bricks and
    mortar, which will make you stop at your tracks. Are you ready
    to attain the treasure that is rightfully yours?!"
10 dialog_autowrap = true

```

C.5.7 win_dialog.tscn

```
1  [gd_scene format=3 uid="uid://b5q8ovcigrvyr"]
2
3  [node name="WinDialog" type="ConfirmationDialog"]
4  title = "Tree-Munching Time!"
5  position = Vector2i(326, 100)
6  size = Vector2i(500, 421)
7  mouse_passthrough = true
8  ok_button_text = "Get Me a New Village"
9  dialog_text = "You found your treasure! Well done, you!"
10
11  Would you like to travel to a new village in the hopes of finding
    another ring? Or would you like to take your treasure home now?
    "
12  dialog_autowrap = true
13  cancel_button_text = "Get Me Out of Here"
```

C.5.8 icon.svg.import

```
1  [remap]
2
3  importer="texture"
4  type="CompressedTexture2D"
5  uid="uid://crgf6ascxsdt0"
6  path="res://.godot/imported/icon.svg-218
    a8f2b3041327d8a5756f3a245f83b.ctex"
7  metadata={
8  "vram_texture": false
9  }
10
11  [deps]
```

```

12
13     source_file="res://icon.svg"
14     dest_files=["res://.godot/imported/icon.svg-218
           a8f2b3041327d8a5756f3a245f83b.ctex"]
15
16     [params]
17
18     compress/mode=0
19     compress/high_quality=false
20     compress/lossy_quality=0.7
21     compress/hdr_compression=1
22     compress/normal_map=0
23     compress/channel_pack=0
24     mipmaps/generate=false
25     mipmaps/limit=-1
26     roughness/mode=0
27     roughness/src_normal=""
28     process/fix_alpha_border=true
29     process/premult_alpha=false
30     process/normal_map_invert_y=false
31     process/hdr_as_srgb=false
32     process/hdr_clamp_exposure=false
33     process/size_limit=0
34     detect_3d/compress_to=1
35     svg/scale=1.0
36     editor/scale_with_editor_scale=false
37     editor/convert_colors_with_editor_theme=false

```

C.5.9 monochrome_packed.png.import

```

1     [remap]
2

```

```

3  importer="texture"
4  type="CompressedTexture2D"
5  uid="uid://m662wwd4prmn"
6  path="res://.godot/imported/monochrome_packed.png-6
    b9bd1c64dd50f72acd3afd14d1ac34f.ctex"
7  metadata={
8  "vram_texture": false
9  }
10
11  [deps]
12
13  source_file="res://monochrome_packed.png"
14  dest_files=["res://.godot/imported/monochrome_packed.png-6
    b9bd1c64dd50f72acd3afd14d1ac34f.ctex"]
15
16  [params]
17
18  compress/mode=0
19  compress/high_quality=false
20  compress/lossy_quality=0.7
21  compress/hdr_compression=1
22  compress/normal_map=0
23  compress/channel_pack=0
24  mipmaps/generate=false
25  mipmaps/limit=-1
26  roughness/mode=0
27  roughness/src_normal=""
28  process/fix_alpha_border=true
29  process/premult_alpha=false
30  process/normal_map_invert_y=false
31  process/hdr_as_srgb=false
32  process/hdr_clamp_exposure=false
33  process/size_limit=0

```



```
34 detect_3d/compress_to=1
```

C.5.10 LICENSE

```
1 This work is licensed under the Creative Commons Attribution-
  ShareAlike 4.0 International License. To view a copy of this
  license, visit http://creativecommons.org/licenses/by-sa/4.0/
  or send a letter to Creative Commons, PO Box 1866, Mountain
  View, CA 94042, USA.
```

C.6 LSystemGrammarDemo

These are the source code listings for the L-System demonstration used to generate the screenshots in Figures 3.1, 3.2, 3.3 and 3.4 (in the report body). The link to the publicly available source code repository is here: <https://github.com/Zishan-Rahman/LSystemGrammarDemo>

C.6.1 .gitattributes

```
1 # Normalize EOL for all files that Git considers text files.
2 * text=auto eol=lf
```

C.6.2 .gitignore

```
1 # Godot 4+ specific ignores
2 .godot/
```

C.6.3 project.godot

```
1 ; Engine configuration file.
```

```

2   ; It's best edited using the editor UI and not directly,
3   ; since the parameters that go here are not all obvious.
4   ;
5   ; Format:
6   ;   [section] ; section goes between []
7   ;   param=value ; assign values to parameters
8
9   config_version=5
10
11  [application]
12
13  config/name="LSystemGrammarDemo"
14  run/main_scene="res://DemoNode.tscn"
15  config/features=PackedStringArray("4.0")
16
17  [display]
18
19  window/stretch/mode="canvas_items"
20  window/stretch/aspect="expand"
21
22  [gui]
23
24  common/drop_mouse_on_gui_input_disabled=true
25
26  [physics]
27
28  common/enable_pause_aware_picking=true

```

C.6.4 DemoNode.tscn

```

1   [gd_scene load_steps=2 format=3 uid="uid://bu380we4od0ln"]
2

```

```

3  [ext_resource type="Script" path="res://DemoNode.gd" id="1"]
4
5  [node name="DemoNode" type="Node"]
6  script = ExtResource("1")
7  choices = "deterministic"
8
9  [node name="Timer" type="Timer" parent="."]
10
11 [node name="TextLabel" type="Label" parent="."]
12 offset_right = 1152.0
13 offset_bottom = 23.0
14 autowrap_mode = 3
15
16 [connection signal="timeout" from="Timer" to="." method="
    _on_Timer_timeout"]

```

C.6.5 DemoNode.gd

```

1  extends Node
2
3  # Basic: https://youtu.be/feNVBEPXAcE?t=77 (L = +)
4  # Choices: http://paulbourke.net/fractals/lsys/
5  # Deterministic: https://ww1.biologie.uni-hamburg.de/b-online/
    e28_3/lsys.html#DOL-system
6
7  ## Allows you to decide which ruleset to use. See the script file
    for the sources of said rulesets.
8  @export_enum("basic", "choices", "deterministic") var choices:
    String = "choices"
9  var axiom: String
10 @onready var string: String
11 @onready var timer = $Timer

```

```

12  @onready var label = $TextLabel
13  @onready var rules: Array[Dictionary]
14
15  func set_values() -> void:
16      match choices:
17          "basic":
18              rules = [
19                  {
20                      "from": "F",
21                      "to": "F+F"
22                  }
23              ]
24              axiom = "F+"
25          "choices":
26              rules = [
27                  {
28                      "from": "F",
29                      "to": "F+--FFFF+F+-FF"
30                  }
31              ]
32              axiom = "F+F+F+F"
33          "deterministic":
34              rules = [
35                  {
36                      "from": "a",
37                      "to": "ab"
38                  },
39                  {
40                      "from": "b",
41                      "to": "a"
42                  }
43              ]
44              axiom = "b"

```

```

45
46 func _ready() -> void:
47     set_values()
48     string = axiom
49     label.size.x = get_viewport().size.x
50     label.text = string
51     print(len(string))
52     timer.start()
53
54 # Thanks to Alexander Gillberg (Codat) for inspiration
55 # https://youtu.be/eY9XkJERiG0
56 # Code adapted with his permission
57 func get_new_replacement(character: String) -> String:
58     for rule in rules:
59         if rule["from"] == character:
60             return rule["to"]
61     return ""
62
63 func _on_Timer_timeout() -> void:
64     # Thanks to Alexander Gillberg (Codat) for inspiration
65     # https://youtu.be/eY9XkJERiG0
66     # Code adapted with his permission
67     var new_string: String = ""
68     for character in string:
69         new_string += get_new_replacement(character)
70     string = new_string
71     label.text = string
72     print(len(string))

```

C.6.6 icon.svg.import

```

1 [remap]

```

```

2
3   importer="texture"
4   type="CompressedTexture2D"
5   uid="uid://cwnnuqmejj04q"
6   path="res://.godot/imported/icon.svg-218
      a8f2b3041327d8a5756f3a245f83b.ctex"
7   metadata={
8     "vram_texture": false
9   }
10
11   [deps]
12
13   source_file="res://icon.svg"
14   dest_files=["res://.godot/imported/icon.svg-218
      a8f2b3041327d8a5756f3a245f83b.ctex"]
15
16   [params]
17
18   compress/mode=0
19   compress/high_quality=false
20   compress/lossy_quality=0.7
21   compress/hdr_compression=1
22   compress/normal_map=0
23   compress/channel_pack=0
24   mipmaps/generate=false
25   mipmaps/limit=-1
26   roughness/mode=0
27   roughness/src_normal=""
28   process/fix_alpha_border=true
29   process/premult_alpha=false
30   process/normal_map_invert_y=false
31   process/hdr_as_srgb=false
32   process/hdr_clamp_exposure=false

```

```
33 process/size_limit=0
34 detect_3d/compress_to=1
35 svg/scale=1.0
36 editor/scale_with_editor_scale=false
37 editor/convert_colors_with_editor_theme=false
```

C.6.7 LICENSE

```
1 This work is licensed under the Creative Commons Attribution-
  ShareAlike 4.0 International License. To view a copy of this
  license, visit http://creativecommons.org/licenses/by-sa/4.0/
  or send a letter to Creative Commons, PO Box 1866, Mountain
  View, CA 94042, USA.
```

C.7 README

```
1 I verify that I am the sole author of the programs contained in
  this folder, except where explicitly stated to the contrary.
2
3 Zishan Rahman
4 21st April 2023
```