11/12/2024

GADE7322

Part 3

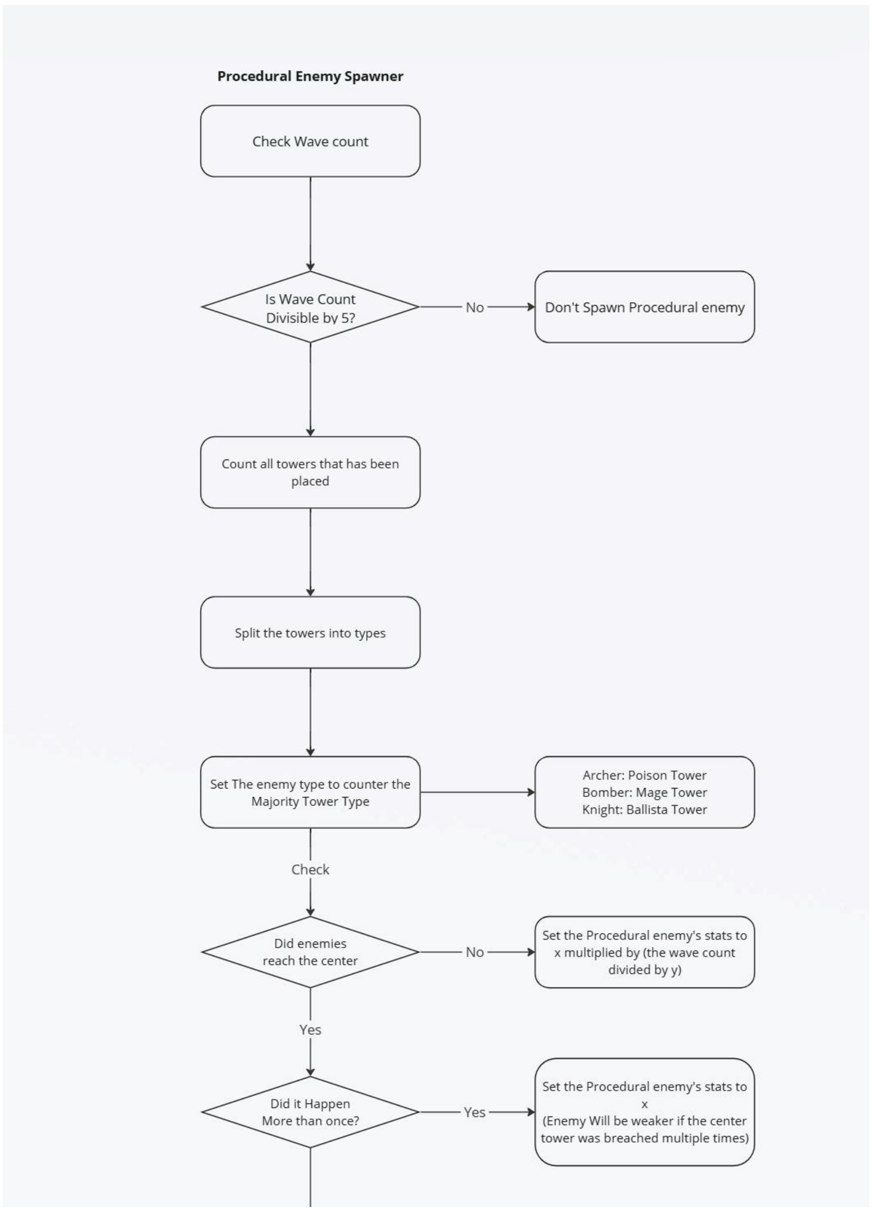
Pasqaulo Shields, Sybrandt van Niekerk

MELLISSA GROBBLER

INDIVIDUAL PROCEDURAL PLANNING:

PASQAULO SHIELDS

PROCEDURAL ENEMY PLANNING:


Procedural Enemy Spawner
Check Wave count
Is Wave Count Divisible by 5?
No
Don't Spawn Procedural enemy
Count all towers that has been placed
Split the towers into types
Set The enemy type to counter the Majority Tower Type
Archer: Poison Tower
Bomber: Mage Tower
Knight: Ballista Tower
Check
Did enemies reach the center
No
Set the Procedural enemy's stats to x multiplied by (the wave count divided by y)
Yes
Did it Happen More than once?
Yes
Set the Procedural enemy's stats to x
(Enemy Will be weaker if the center tower was breached multiple times)

Determining Enemy Type:

In the game, there will be a procedurally generated enemy (Boss enemy), that will spawn every 5 waves. The way that this works is that the game will first check the number of waves that the player is currently on, and if that number is divisible by 5, it will spawn the procedurally generated enemy. However, before the enemy is spawned, the game will count all of the current towers that has been placed in the game. It will then split those towers into types and see which tower type is in the majority. It will then set the enemy type that will be spawned to counter the towers that are in majority. Archer enemy types will counter poison towers, Bomber Units will counter mage towers, and the knight enemy will counter the Ballista tower.

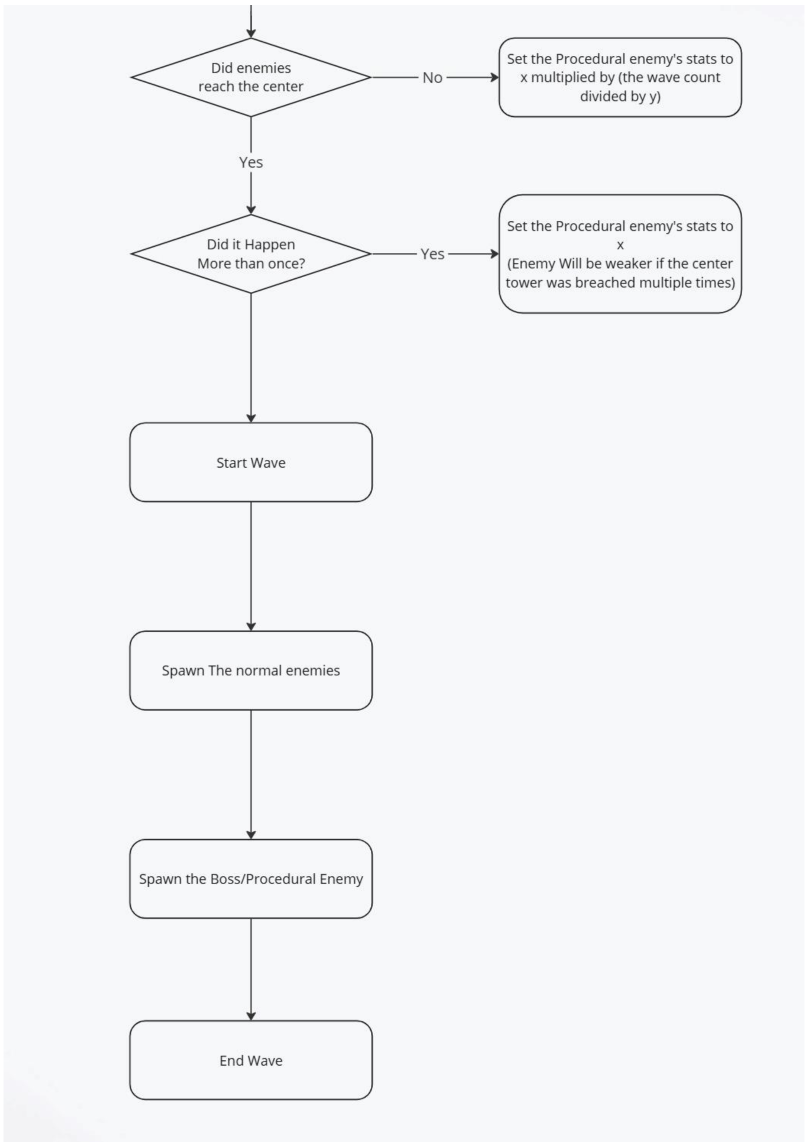
Determining Enemy Stats:

After the enemy type that will be spawned has been determined, it will then perform a check to see if enemies have reached the center tower.

If they didn't reach the center tower, the procedural enemy's stats will be set to x times their original stats, and this number will be further multiplied by looking at the number of waves and dividing that number by variable y. This means that the enemy will be stronger if the enemies didn't reach the center.

If the enemies in the wave did reach the center, meaning that the player doesn't have a strong enough defence, the procedural enemy's stats will be set to x times their original stats. This number wont be multiplied further like in the previous part, as it is to ensure that the player is able to kill the procedurally generated enemy.

This will ensure that the enemy provide a consistent challenge throughout the game and doesn't become too difficult/impossible to kill. The enemy will therefore

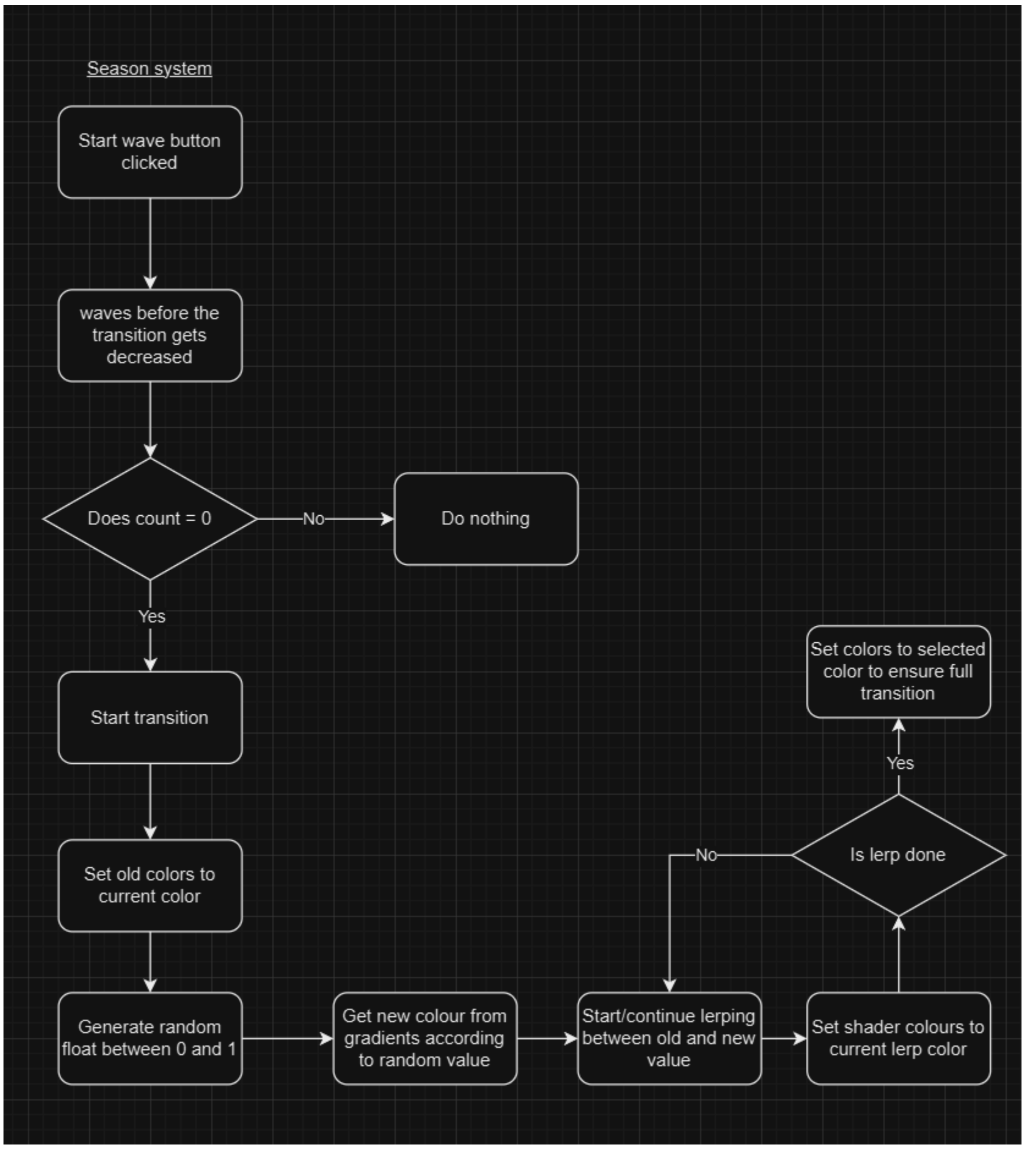

Did enemies reach the center
No
Set the Procedural enemy's stats to x multiplied by (the wave count divided by y)
Yes
Did it Happen More than once?
Yes
Set the Procedural enemy's stats to x (Enemy Will be weaker if the center tower was breached multiple times)
Start Wave
Spawn The normal enemies
Spawn the Boss/Procedural Enemy
End Wave

Spawning the enemy:

After the enemy type and its stats have been determined, it will then start the wave. It will spawn the normal enemies as per usual, but then the procedural/boss enemy will be spawned after all the other enemies have been spawned. It will then end the wave. The purpose of this enemy is to provide the player with more challenging enemies that are able to pose a considerable threat to their defence.

SYBRANDT VAN NIEKERK

PROCEDURAL SEASON SYSTEM


Season system
Start wave button clicked
waves before the transition gets decreased
Does count = 0
No
Do nothing
Yes
Start transition
Set old colors to current color
Generate random float between 0 and 1
Get new colour from gradients according to random value
Start/continue lerping between old and new value
Set shader colours to current lerp color
Is lerp done
No
Yes
Set colors to selected color to ensure full transition

Determining when to transition:

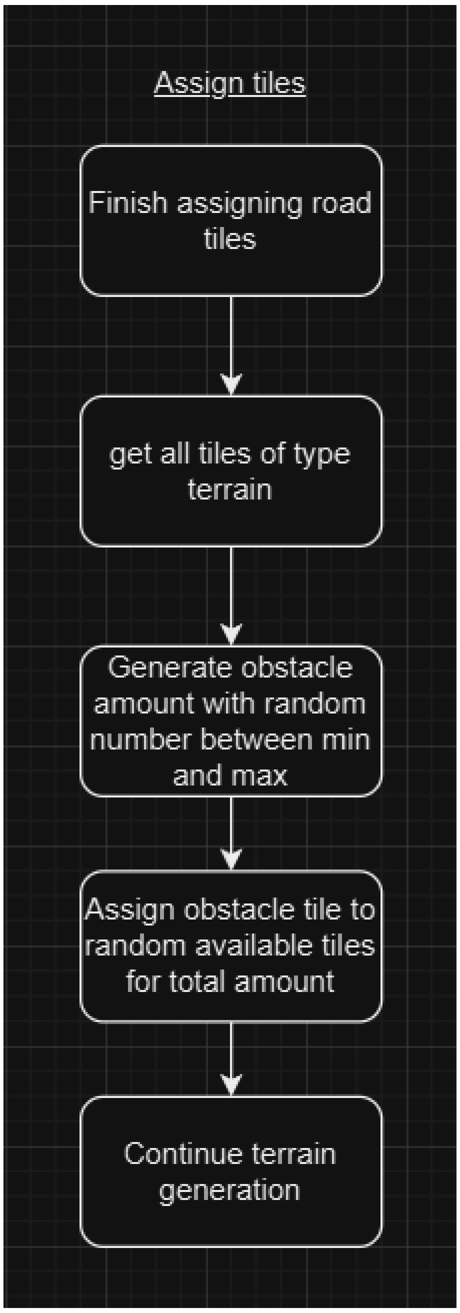
The system will have an adjustable int which it will use to determine how many waves the player has completed. Every time the player starts a new wave, this value will be decreased by 1 and once it is 0 the transition sequence will be triggered, and the value will be reset.

Transitioning:

Once the transition sequence is triggered the system will keep track of what the current/old colour of the shaders were. It will then generate a random number between 0 and 1 which it will use to get the new colour to transition to from the season gradient. After this has been gotten it will start to lerp from the current colour to the new randomly selected colour over whatever time you specified.

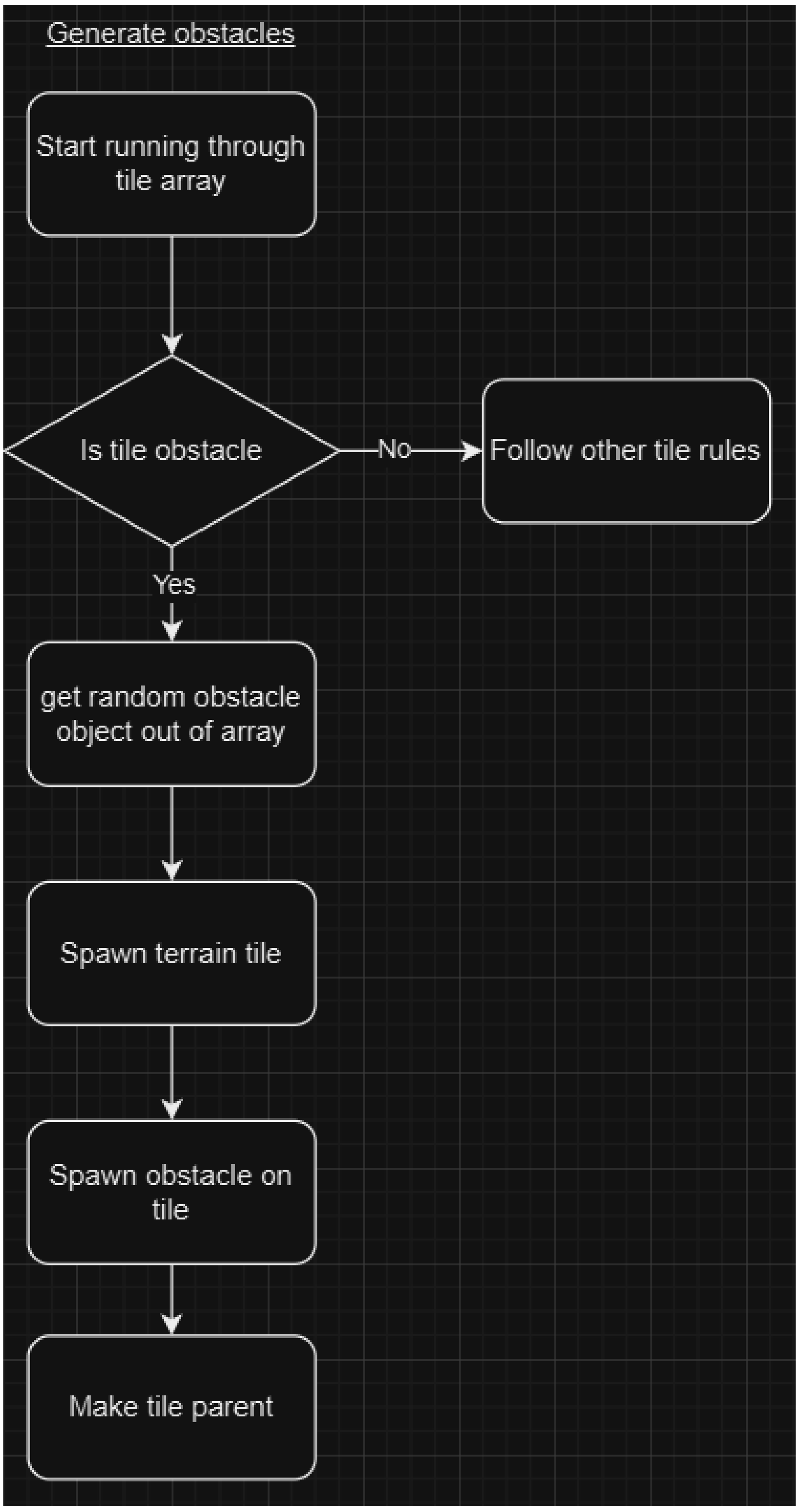
The system will also keep track of what season it is in and depending on this value it will select which gradients to use for each shader. The gradients are divided up into summer, autumn, winter and spring which each uses complimentary colours of the other shaders gradients, to ensure the colour pallet will always look appealing.

PROCEDURAL OBSTACLE SPAWNER


Assign tiles
Finish assigning road tiles
get all tiles of type terrain
Generate obstacle amount with random number between min and max
Assign obstacle tile to random available tiles for total amount
Continue terrain generation

Assigning obstacle tiles:

After the terrain generator has assigned its paths and spawners it will start assigning obstacle tile types. It will take all terrain tile types and add them in a list. After that a random obstacle count will be generated using assigned min and max values. The system will then randomly assign the generated number of obstacles to tiles in this list. Once this process is done it will continue generating the map as usual.



Generating obstacles:

When the system starts generating actual objects it will start running through all assigned tiles. If the tile is of type obstacle it will generate a default empty terrain tile for its base. When this step is done for all tiles, it will run through the list of obstacle tiles to get their positions. It will then randomly select objects from an array and spawn them on these tiles until all tiles have received their obstacles. When this is done the terrain generator will start generating trees which will naturally go around obstacles in their way making the environment look natural.

CHOSEN SYSTEMS:

We decided to go with Sybrandt's systems as these systems would make our terrain less static and more natural. By implementing obstacles, we could add an additional layer of challenge as there might be less space for placing towers. These obstacles also made it look like the terrain was an old battlefield which has been active for a long time.

The reason why we also implemented the seasonal system was to give the user as sense of movement in time. This system also strengthened the idea of an old battlefield as the player would go through multiple seasons, thus years pass in the game. By doing this we could immerse our player better by making the game world feel more natural and non-static with differently coloured passing seasons.

GADE PART 2:

ENEMIES:

2 Additional Enemy types:

• **Archer**	**:poison**	**(ML health, L attack, L speed, H range)**
• **Suicide bomber**	**:ballista**	**(L health, H attack, H speed, L Range)**
• **Knight**	**: mage**	**(M health, M attack, M speed, L range)**

Have health bars

Attack and destroy towers

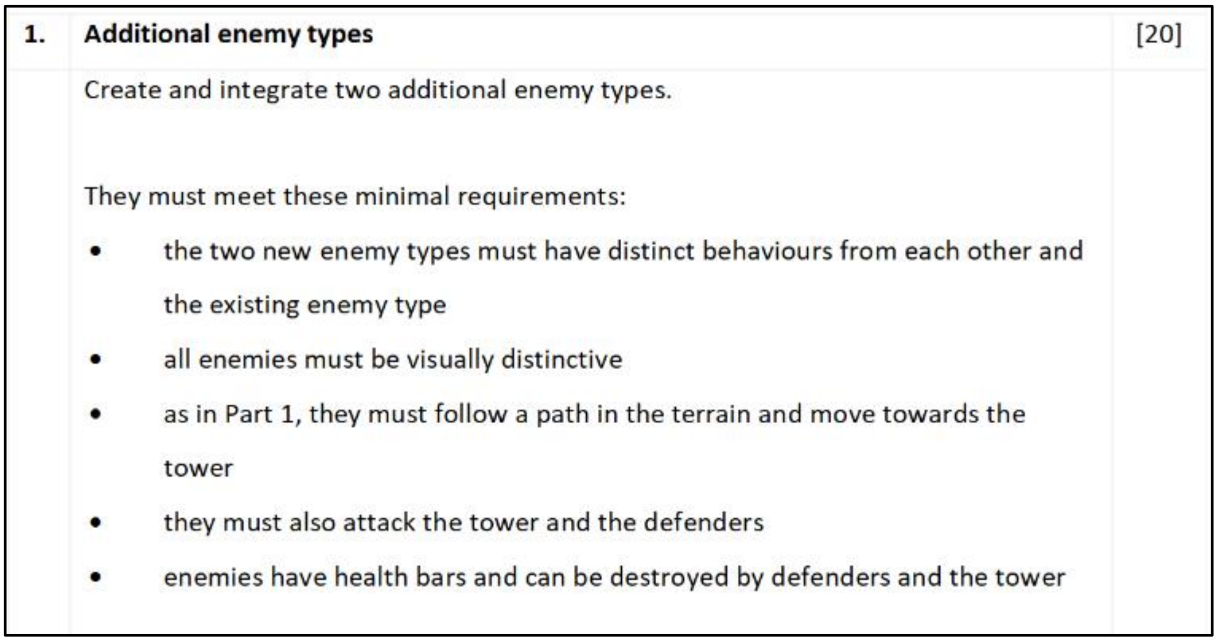
Follow same path

Tower Types:

Poison: knights, suicide bombers

Mage: archers, suicide

Balista: Knights, archers



1.	**Additional enemy types**	[20]
	Create and integrate two additional enemy types. They must meet these minimal requirements: • the two new enemy types must have distinct behaviours from each other and the existing enemy type • all enemies must be visually distinctive • as in Part 1, they must follow a path in the terrain and move towards the tower • they must also attack the tower and the defenders • enemies have health bars and can be destroyed by defenders and the tower	

DEFENDERS:

2 Additional Defender types:

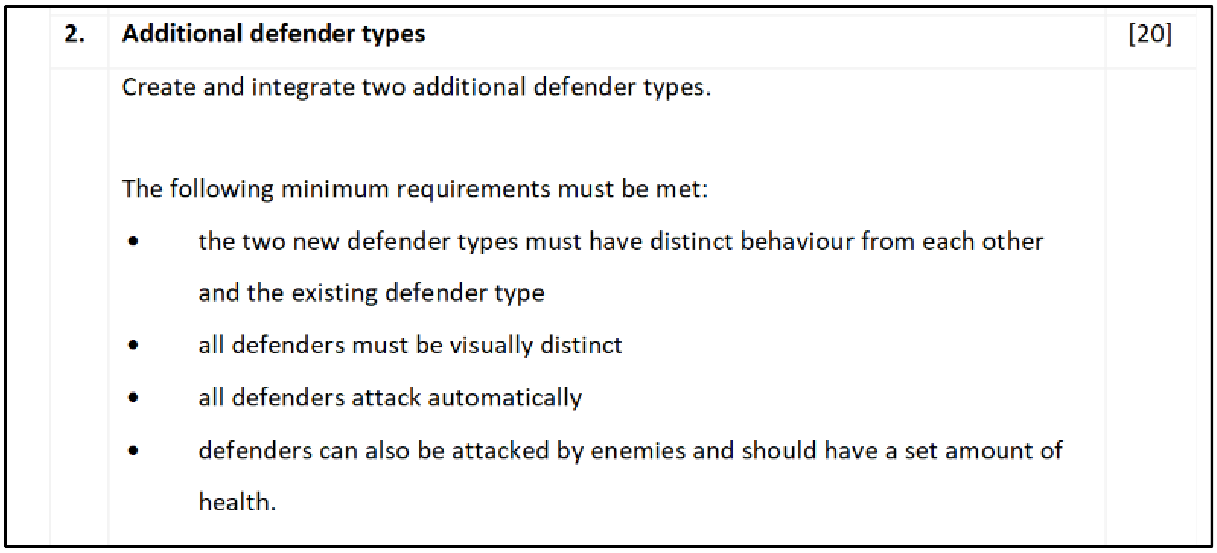
- **Mage** or cannon (Projectile that hits multiple enemies) **(L health, MH Attack, L attack speed, H attack range)**
- **Poison** tower (damage over time) **(MH health, L attack, MH attack Speed, ML Attack Range)**
- **Ballista** **(M health, M attack, Medium Attack Speed, M attack Range)**

Must be visually distinct

Attack automatically

Must have different behaviour

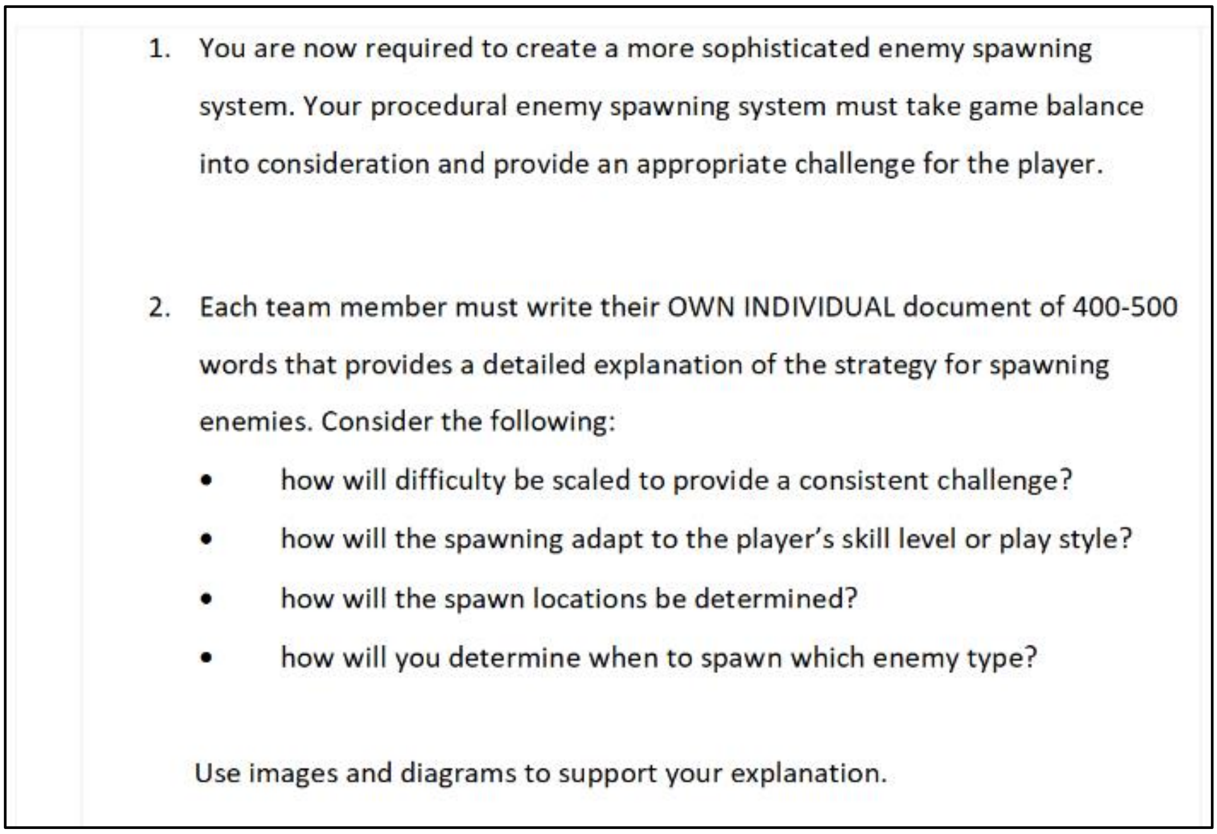
Should have set health and can be destroyed



2.	**Additional defender types**	[20]
	Create and integrate two additional defender types. The following minimum requirements must be met: • the two new defender types must have distinct behaviour from each other and the existing defender type • all defenders must be visually distinct • all defenders attack automatically • defenders can also be attacked by enemies and should have a set amount of health.	

PLANNING PROCEDURAL ENEMY WAVES:

Individual:

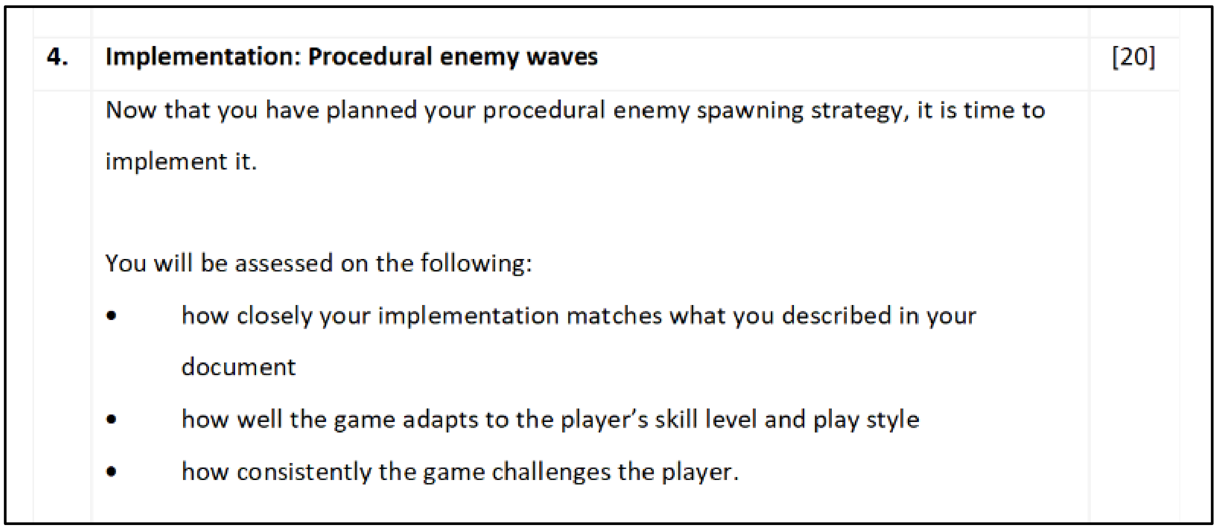


1. You are now required to create a more sophisticated enemy spawning system. Your procedural enemy spawning system must take game balance into consideration and provide an appropriate challenge for the player.

2. Each team member must write their OWN INDIVIDUAL document of 400-500 words that provides a detailed explanation of the strategy for spawning enemies. Consider the following:
   - how will difficulty be scaled to provide a consistent challenge?
   - how will the spawning adapt to the player's skill level or play style?
   - how will the spawn locations be determined?
   - how will you determine when to spawn which enemy type?

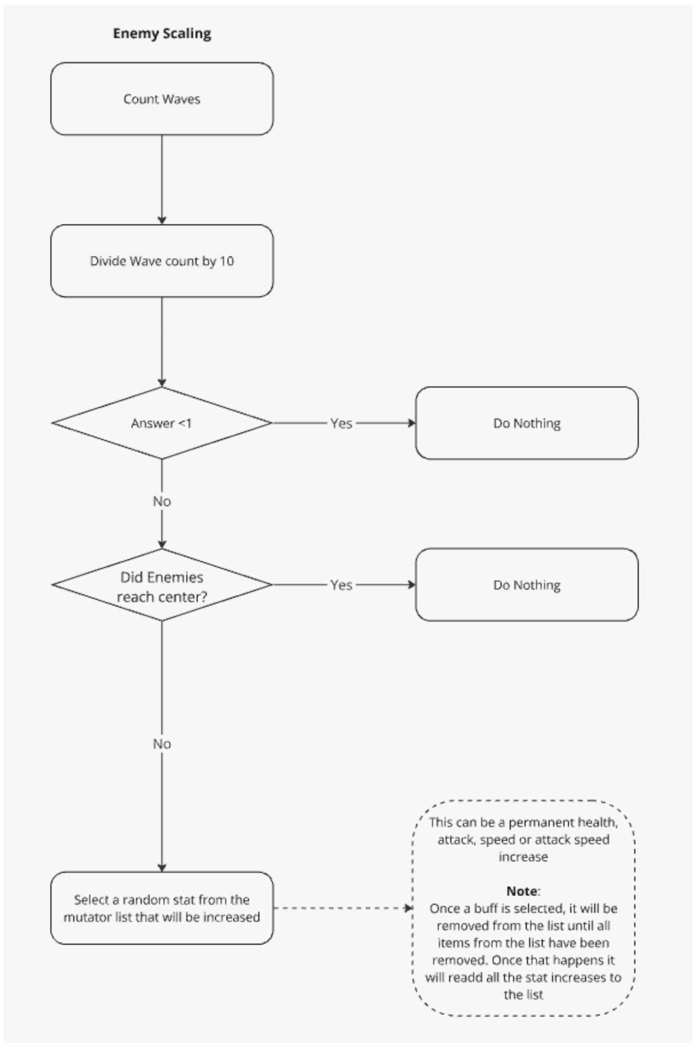
Use images and diagrams to support your explanation.

IMPLEMENTING PROCEDURAL ENEMY WAVES:



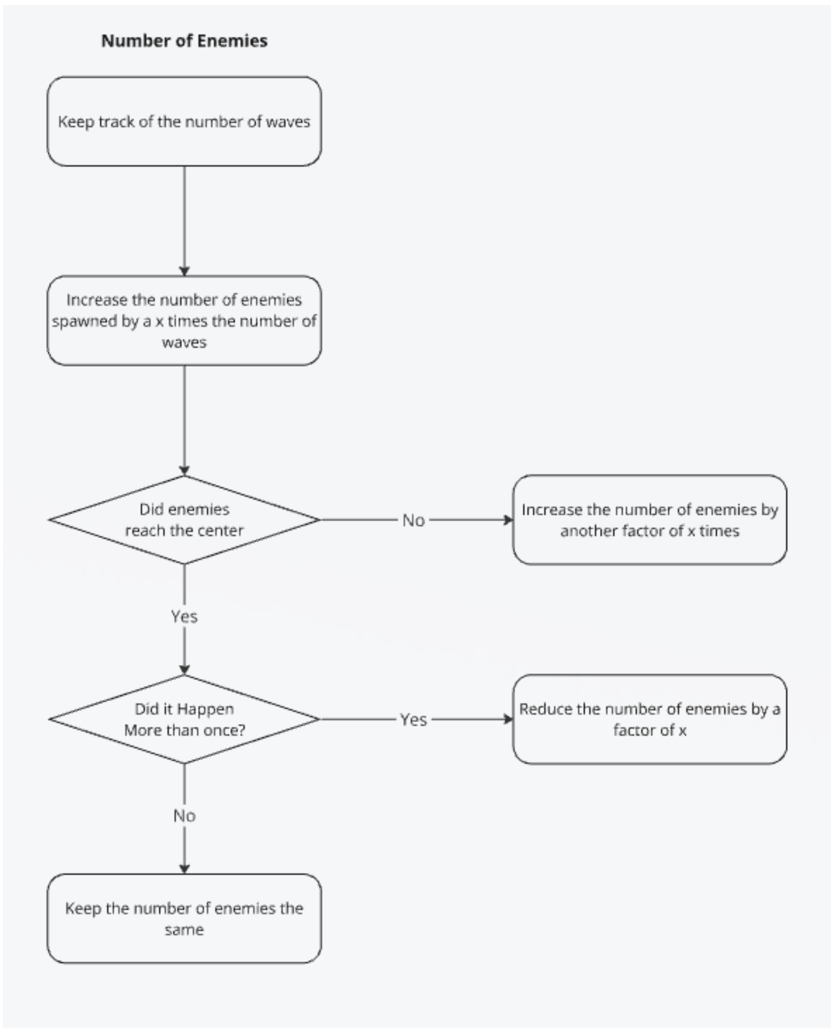
4.	**Implementation: Procedural enemy waves**	[20]
	Now that you have planned your procedural enemy spawning strategy, it is time to implement it. You will be assessed on the following: • how closely your implementation matches what you described in your document • how well the game adapts to the player's skill level and play style • how consistently the game challenges the player.	

HOW THE DIFFICULTY WILL BE SCALED TO PROVIDE A CONSISTENT CHALLENGE:


Enemy Scaling
Count Waves
Divide Wave count by 10
Answer <1
Yes
Do Nothing
No
Did Enemies reach center?
Yes
Do Nothing
No
Select a random stat from the mutator list that will be increased
This can be a permanent health, attack, speed or attack speed increase
Note: Once a buff is selected, it will be removed from the list until all items from the list have been removed. Once that happens it will readd all the stat increases to the list

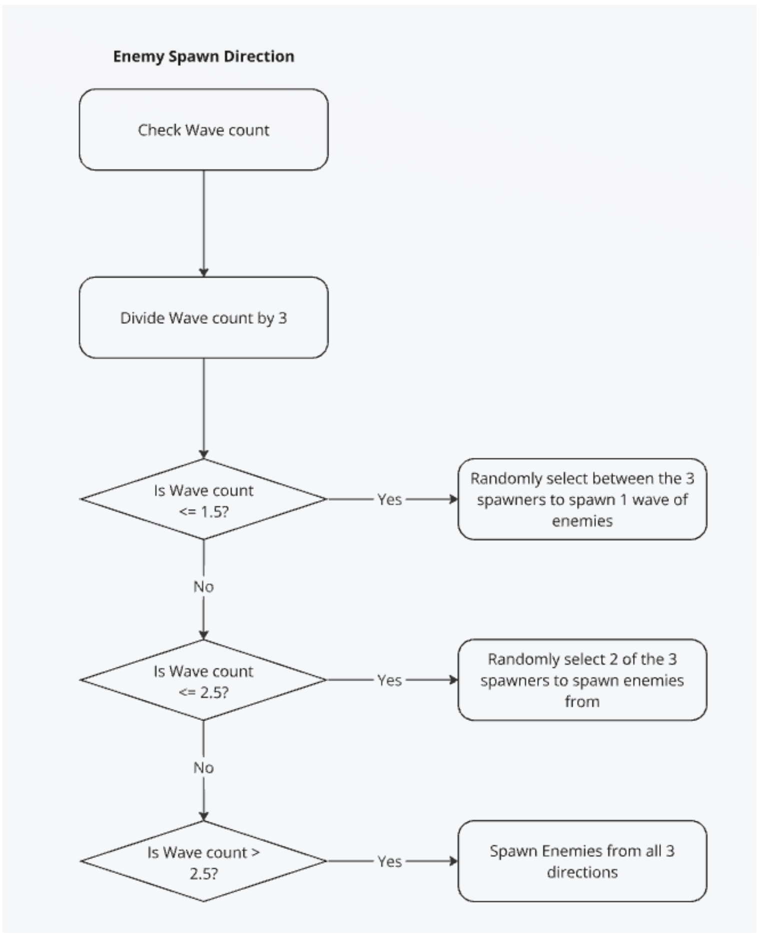
In the game, the enemy scaling will be handled dynamically. It will initially start by counting the number of waves in order to see how far the player has gotten into the game. It will then divide that number by 10. The reason for this is so that the enemies only start scaling after round 10 once all 3 spawners are used to spawn enemies (Discussed in spawn locations). If the answer is larger than 1, meaning that the player has reached wave 10, it will first look at if enemies reached the centre tower. If they didn't reach the centre tower, a random stat of theirs will be increased from a list of available stat increases. Once a stat is increased, it will be removed from the list until all the items in the list are removed. Once this happens, it will repopulate the list so that the stat increases can stack.

HOW THE SPAWNING WILL ADAPT TO THE PLAYER'S SKILL LEVEL OR PLAYSTYLE:


Number of Enemies
Keep track of the number of waves
Increase the number of enemies spawned by a x times the number of waves
Did enemies reach the center
No
Increase the number of enemies by another factor of x times
Yes
Did it Happen More than once?
Yes
Reduce the number of enemies by a factor of x
No
Keep the number of enemies the same

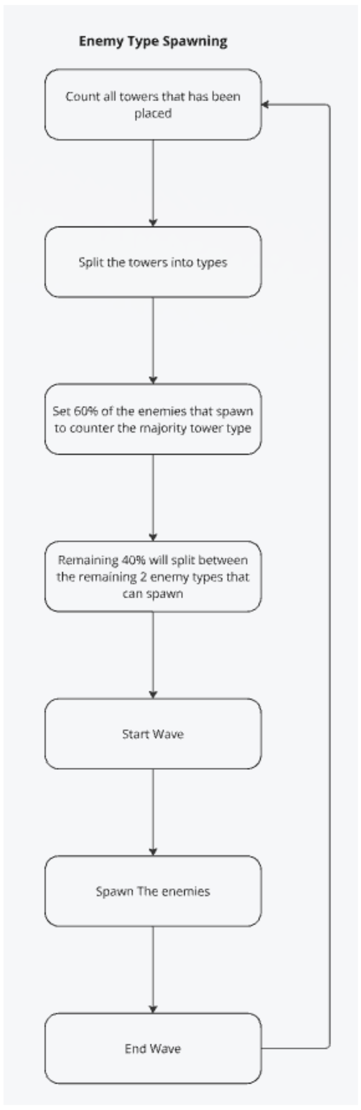
For the enemy spawning, it will keep track of the number of waves and increase how many enemies will be spawned according to the wave that is currently active. That way the enemy number will constantly increase. Then it will look at if the enemies reached the centre tower. If they didn't, the number of enemies will increase by an additional amount. If the enemies reached the centre tower more than once, the number of enemies will be reduced by a certain amount. If the enemies reached the centre tower, but it didn't happen more than once, the number of enemies will be kept the same. This is done so that the difficulty of the game remains relatively constant.

HOW THE SPAWN LOCATIONS WILL BE DETERMINED:


Enemy Spawn Direction
Check Wave count
Divide Wave count by 3
Is Wave count <= 1.5?
Yes
Randomly select between the 3 spawners to spawn 1 wave of enemies
No
Is Wave count <= 2.5?
Yes
Randomly select 2 of the 3 spawners to spawn enemies from
No
Is Wave count > 2.5?
Yes
Spawn Enemies from all 3 directions

The spawn locations will be determined according to the wave count. It will count the amount of waves and divide it by 3. If the answer is less than 1.5 (meaning that it is wave 1-4), it will randomly select a singular spawner between the 3 spawners to spawn one wave of enemies. If the divided number is smaller than 2.5, but greater than 1.5, 2 out of the 3 spawners will be selected to spawn enemies from. Finally, if the divided number is larger than 2.5, meaning that it is wave 8+, all 3 spawners will be used to spawn enemies from.

HOW THE ENEMY TYPES WILL BE DETERMINED:


Enemy Type Spawning
Count all towers that has been placed
Split the towers into types
Set 60% of the enemies that spawn to counter the majority tower type
Remaining 40% will split between the remaining 2 enemy types that can spawn
Start Wave
Spawn The enemies
End Wave

To spawn the different enemy types, it will look at all the towers that have been placed so far. Once that has finished, it will split the list of towers into their respective defender types. Once the majority tower type has been determined, it will set 60% of the enemies that spawn to counter that specific tower type. For example, if the majority tower is the ballista tower, the majority number of enemies that will be spawned will be bombers. These bombers are fast and do a lot of damage, but have little health. However since the ballista tower only focusses singular targets, the bombers will reach the tower before they are all killed. The remaining 40% of enemies will be split between the remaining 2 enemy types that can be spawned. The wave will then start. Once the wave ends, it will repeat the process.

GADE PART 2 INDIVIDUAL PLANNING: SYBRANDT

ENEMY SPAWNING DIRECTIONS:

The following way is how the enemy spawning direction will be decided:

The system will keep track of what wave the player is currently on and divide it by 10, thus increasing the directions enemies come from each 10 waves. Afterwards it will check if this value is greater than the number of spawners on the map. If the value is less, it will then randomly select spawners from the spawner list equal to the amount of the value. These spawners will be selected randomly each time this action happens. If the value is greater than the number of spawners, enemies will come from all directions.

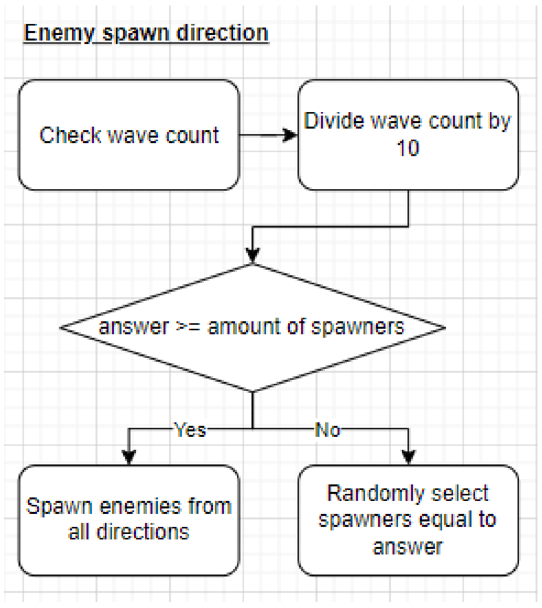

Enemy spawn direction
Check wave count
Divide wave count by 10
answer >= amount of spawners
Yes
No
Spawn enemies from all directions
Randomly select spawners equal to answer

Figure 1: Diagram of enemy spawn direction system

ENEMY MUTATORS:

The system will check what wave count the player is on. If the waves divided by 10 is greater than three, it will add a mutator for every number higher than three, thus adding a mutator every ten waves starting at wave 40.

The mutators will be randomly selected from a list and added to the enemies. Once a mutator has been selected it will be removed from the list and won't be selected again.

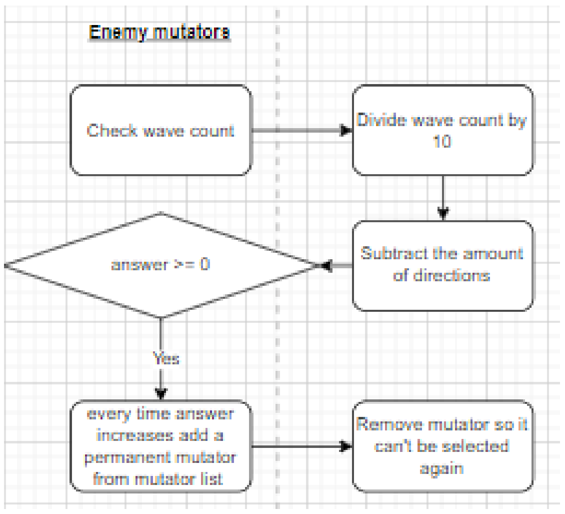

Enemy mutators
Check wave count
Divide wave count by 10
Subtract the amount of directions
answer >= 0
Yes
every time answer increases add a permanent mutator from mutator list
Remove mutator so it can't be selected again

Figure 2:Diagram of enemy mutator system

NUMBER OF ENEMIES TO SPAWN:

The system will keep track to the closest an enemy has gotten to the centre tower. If the enemies did get closer then the previous wave, the number of enemies added to the next wave will be slightly lower than the previous wave. If they did not get closer, it will check if this has happened more than a set number of times. If it did, the number of added enemies will slightly increase in the next wave. If it did not happen x number of times, the added enemy count will stay the same.

Along with this the system will also check if the enemies made it to the centre tower. If they made it more than once the number of enemies will slightly decrease and if not, the enemy count will stay the same.

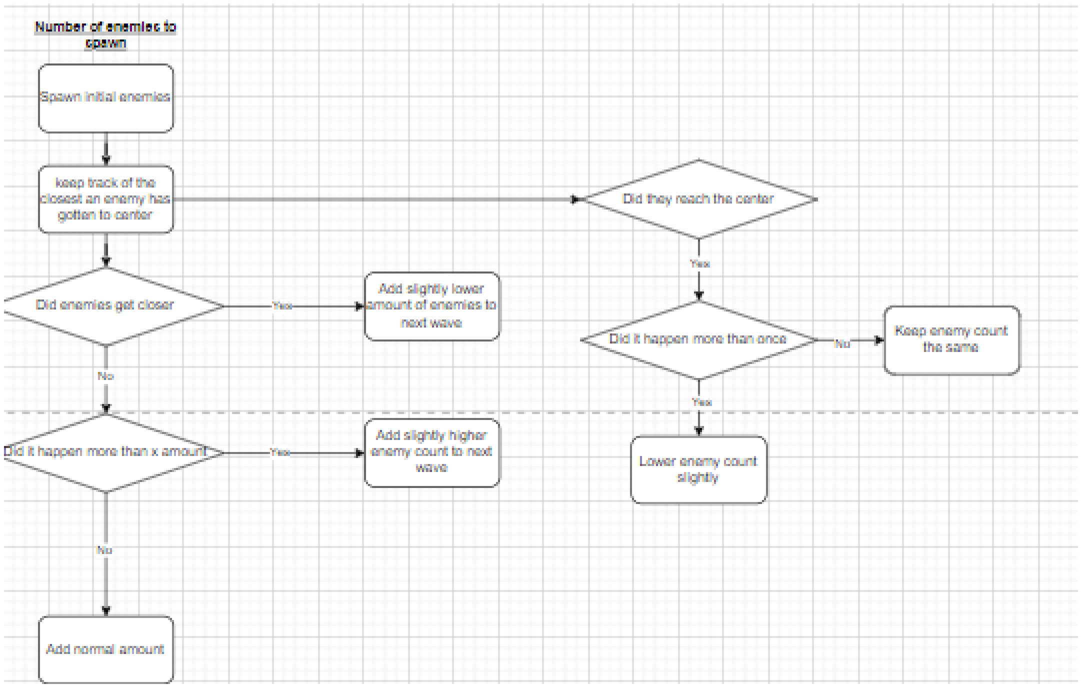

Number of enemies to spawn
Spawn initial enemies
keep track of the closest an enemy has gotten to center
Did they reach the center
Yes
Did enemies get closer
Yes
Add slightly lower amount of enemies to next wave
Did it happen more than once
No
Keep enemy count the same
No
Yes
Did it happen more than x amount
Yes
Add slightly higher enemy count to next wave
Lower enemy count slightly
No
Add normal amount

Figure 3: Diagram of number of enemies to spawn system

ENEMY TYPE SPAWNING:

The system will keep track of all the towers that have been placed and divide them up into their types. It will then see what the majority towers are and select the enemies that combat that tower. These enemies will then be spawned first at the start of the wave to destroy towers and after them, other enemy types will randomly be spawned in.

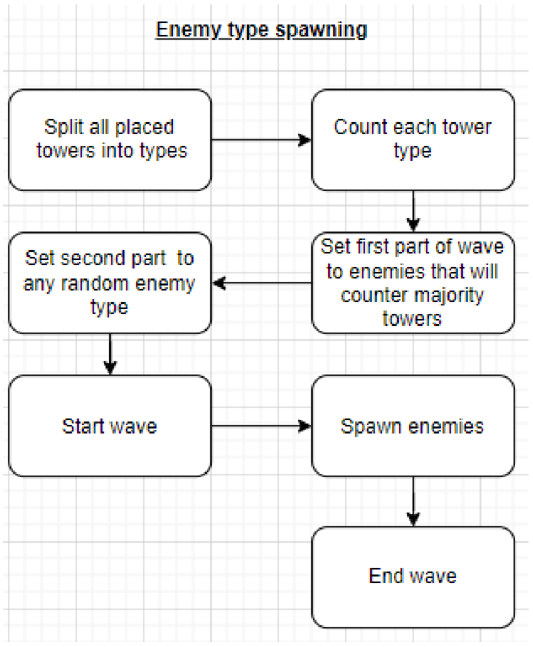

Enemy type spawning
Split all placed towers into types
Count each tower type
Set first part of wave to enemies that will counter majority towers
Set second part to any random enemy type
Start wave
Spawn enemies
End wave

Figure 4: Diagram of enemy type spawning system

FINAL:

SPAWN DIRECTION:

For the enemy spawning, we will mix both of our spawning directions. Every 5 waves will use an additional spawner. For example:

Wave counts:

- 1-5: One random direction each wave
- 6-10: Two random directions each round
- 11-15: All three directions each wave onward

ENEMY SCALING:

Waves 21+, every 5 waves will add an additional enemy mutator. Once the list is empty, it will re-add the list of mutators so that it can stack.

List of mutators:

- Health
- Attack
- Speed
- Attack Speed

NUMBER OF ENEMIES TO SPAWN:

For the number of enemies to spawn we will be using an altered method Sybrandt's spawning method.

The additional enemy count will be lowered by a percentage if the central tower was reached and they will no longer focus on how close they came

ENEMY TYPE SPAWNING:

For the enemy types, we will be using Pasqaulo's method as it uses ratios that can be implemented easily.

This method will start working from wave 16+.