Game 7

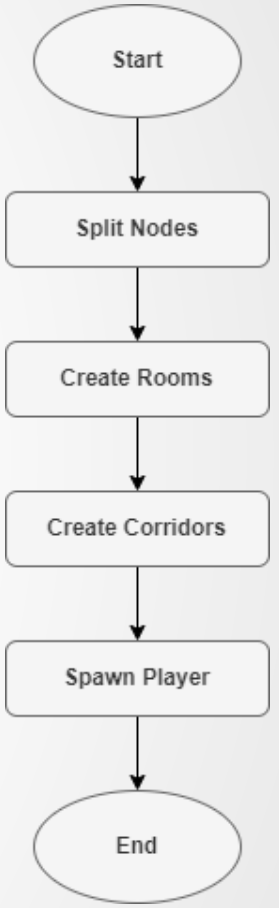
Procedural Generated Dungeon

Procedural Generated Dungeon ~ Introduction

- Procedural Generation is a very useful technique of generating content on the fly.
- It allows developers to be more creative with their games, and to give their projects a sense of re-playability.
- Content is generated by following a set of pre-defined mathematical algorithms and rules.
- The most popular known game to make use of this technique is Minecraft, which generates an essentially infinite world that is unique to each seed, meaning it gives players the opportunity to explore new terrain without any limitations.

Procedural Generated Dungeon ~ AI Explanation (1)

- In this dungeon game, the procedural generation technique is used to create a layout of a traversable dungeon for the player.
- This is done through the use of an algorithm called Binary Space Partitioning (BSP).
- BSP works by splitting a map into several different nodes such that each one is approximately the size of a single dungeon room.
- Each node is then filled up with a room and, through the use of loops and Raycasts, is connected to one another via corridors.



Procedural Generated Dungeon ~ AI Explanation (2)

- Firstly, the root node is created. The size is set to the size of the map, and the position to (0, 0, 0).
- This node is then passed into the Split() function, where it has a 50/50 chance of either being split horizontally or vertically.
- Regardless of which manner it is split, two new children nodes are created and are given either a new height or width.
- Their positions are adjusted to their respective parts of the map.
- The orientation for the next split is then alternated. This is done before each split to avoid an unnatural layout.

Procedural Generated Dungeon ~ AI Explanation (3)

- However, before the children nodes are split again, each one is placed into another function called CheckDimensions() which ensures that they are still big enough to continue being split further.
- The check condition is done by comparing their heights and widths to a pre-set variable with denotes the maximum height of a room.
- If both values are too small, the node is labelled as a leaf and is passed into the function CreateRoom().
- If only one of the values is too small, the node continues to be split but only in a certain orientation.

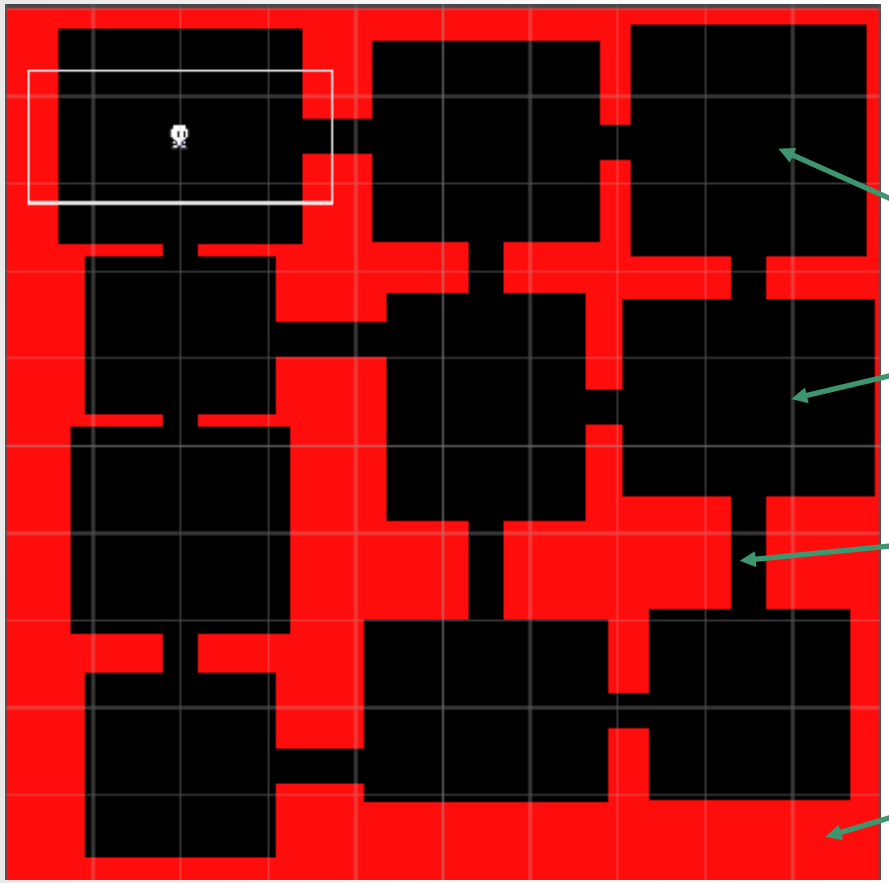
Procedural Generated Dungeon ~ AI Explanation (4)

- The CreateRoom() function simply picks a randomized height and width between specific parameters for the room to be created.
- The room Game Object is instantiated at the centre of the node, and rescaled to the new dimensions.
- When all the nodes have been split and all the rooms have been created, the corridors start being generated.
- This is done by iterating through each room and using Raycasts to check each direction for a neighbouring room.
- If the neighbour is still missing a corridor in its direction, then the CreateCorridors() function is called.

Procedural Generated Dungeon ~ AI Explanation (5)

- The CreateCorridors() function simply finds the midpoint between both rooms and then instantiates the corridor Game Object.
- The height and width of the corridor are set according to the orientation of the rooms and the distance between them.
- Once all the corridors have been generated, the player is finally spawned into the first room and is free to explore the dungeon.

Procedural Generated Dungeon ~ Mini-Game Implementation (1)



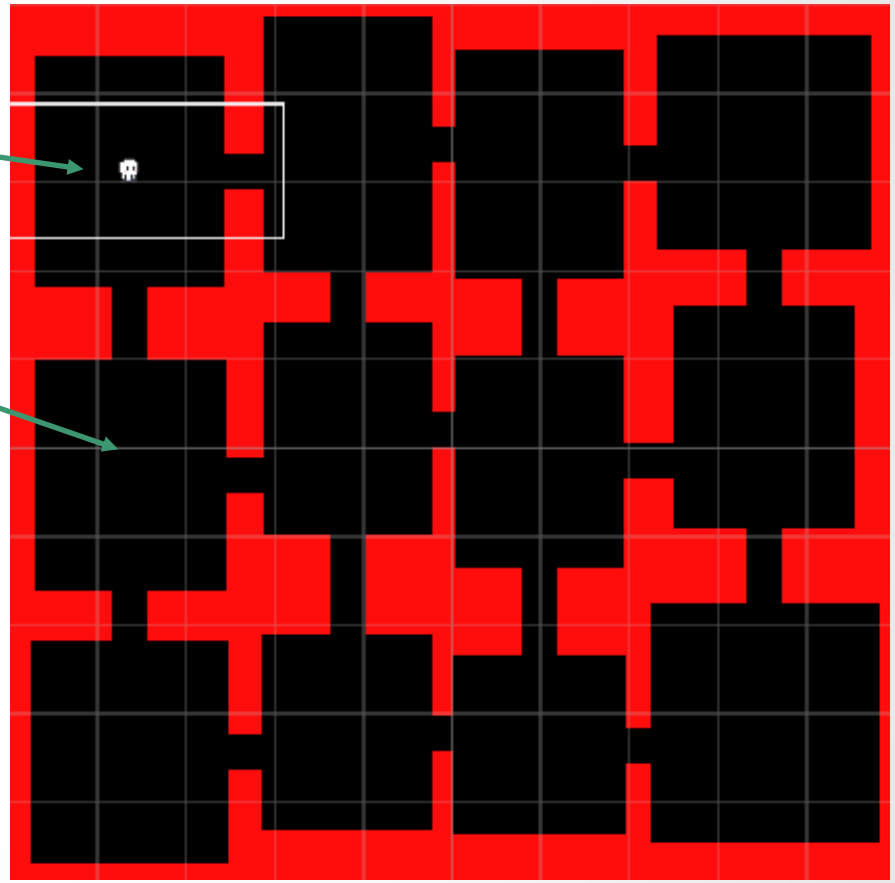
Playable Area:

Player

Rooms

Corridors

Map



Procedural Generated Dungeon ~ Mini-Game Implementation (2)

- The dimensions for the rooms may be set through the parameters MinRoomWidth, MinRoomHeight, MaxRoomWidth, and MaxRoomHeight.
- The dimensions for the map may also be set through the parameters MinX, MinY, MaxX, and MaxY.
- The sprites for the rooms, corridors, player and map can also be set.

Procedural Generated Dungeon ~ Mini-Game Implementation (3)

- **How to Set Up Package:**
  - After importing package, navigate to the top right of Unity Editor and do the following:
    - **Layers > Edit Layers.**
    - **Click the Slider button at the top right of the Inspector, in between the question mark and the three dots.**
    - **Select Preset.**
  - Now navigate to the top left of Unity Editor and do the following:
    - **Edit > Project Settings > Physics 2D > General Settings**
    - **Make sure the option 'Queries Start on Colliders' is unticked.**
    - **Click Layer Collision 2D and untick all the last 3 rows.**

Procedural Generated Dungeon ~ Exercise (1)

- **Now Its your turn to Code! – Let's implement a Procedural Generated Dungeon ☺**

1. Find the script BSP and navigate to the Split() function.
2. You must check if the node being split is a leaf, and whether the split will be horizontal or vertical. The code will remain the same for each split but the dimensions being adjust will be alternated.
3. Create two child nodes.
4. Depending on the orientation of the split, find a random value for the new node's height or width in the range between the maximum room dimensions, and the difference between the node's dimensions and the maximum dimensions.
5. Assign one child the new dimensions, and the sibling the difference between the original node's dimensions and the new ones.

Procedural Generated Dungeon ~ Exercise (2)

6. Check if each child is big enough for another split using the CheckDimensions() function.
    - In this function, you should compare the height and width of the node to the maximum dimensions of a room. (Tip: use 1.5x the maximum dimensions. This is to ensure that no less than 1 room at full size can fit inside the node).
7. Alternate the next split to be the inverse orientation of the current split.
8. If a child can continue being split, pass it to the Split() function, otherwise pass it to the CreateRoom() function.

Procedural Generated Dungeon ~ Conclusion

- Procedural Generation is a very useful technique which all game developers should be familiar with.
- It adds a certain unpredictability to one's game which allows for more re-playability and more unique experiences.
- It also saves the developer the hassle of having to model each map by their own hand.

Procedural Generated Dungeon ~ References

[1] – Prof. A. Dingli, ICS2211: "Level5AutomatedContentGeneration" [Online]. Available: https://www.um.edu.mt/vle/pluginfile.php/1122580/mod_resource/content/1/Level5AutomatedContentGeneration.pdf [Accessed: 17-May-2023]

[2] – G. Uribe, Dungeon Generation using Binary Space Trees, 2019 [Online]. Available: https://medium.com/@guribemontero/dungeon-generation-using-binary-space-trees-47d4a668e2d0 [Accessed: 17-May-2023]

[3] – your daily cup of tea™, Dungeon generation using BSP trees, 2013 [Online]. Available: https://eskerda.com/bsp-dungeon-generation/ [Accessed: 17-May-2023]

[4] – Goldmetal, "Unity Asset Store: Simple 2D Platformer Assets Pack" 2021 [Online]. Available: https://assetstore.unity.com/packages/2d/characters/simple-2d-platformer-assets-pack-188518 [Accessed: 17-May-2023]