Rolling Road - Diary

- Created a 3D Unity Project
- Constructed a straight road and right turn road with 3D unity objects, parenting the road parts with empty game objects placed in the middle where the road starts, adding a child empty game object 'RoadEnd' at the middle of where the road ends for anchoring and connecting the roads. Then I duplicate the right turn road and flip it to form a left turn road piece, finally turning each of the road segments into prefabs
- Added a child empty game object to the centre of the turns on the turning road prefabs turn behaviour
- Added a child empty game object to the main camera positioned in front of it for the runner position and placed a capsule object on top of it with a collider and Rigid body set to kinematic for it to be detected by trigger colliders
- Tilted the camera 30° down vertically looking towards the road
- Placed an empty game object 'RoadCtrl' at the base of the runner to handle road movement and rotation
- Created a new script 'RoadCtrl' attaching this to the 'RoadCtrl' game object for handling the road behaviour

Straight Road Generation

- Declared a public GameObject variable 'road' for attaching the straight road prefab in the inspector, a Vector3 'nextSpawn' to hold the position for where a new road should spawn, a public integer 'startSpawnAmount' for adjusting how many roads to spawn when the generation starts defaulted at 8, a public float 'moveSpeed' for altering the speed of the road movement, and a GameObject variable 'newRd' to reference the latest spawned road game object
- Add a public function 'SpawnRoad()' to spawn the road sections
- At the start of 'SpawnRoad()' a road prefab is instantiated at the position the 'nextSpawn' Vector3, referencing the instantiated prefab in 'newRd', then using 'newRd' reference to parent the new road section to the 'RoadCtrl' game object.
- Next the 'nextSpawn' Vector3 is updated with the 'newRd' game object's 'RoadEnd' position using the update method to keep this up to date with the road's movement
- Created a new function 'SpawnMass()' to spawn a mass of roads, starting with a for loop with 'i' starting at 0 looping to the int value of 'startSpawnAmount', calling 'SpawnRd() each loop. Finally calling this function from the start method
- Testing the roads spawn in a row at the start as expected, but the road needs to spawn back behind the runner more as its currently at the edge
- Declared a variable 'startRoadZOffset' with a default value of -2 in the 'RoadCtrl' script, adding its value to the z axis of the 'nextSpawn' Vector3 variable to begin the road spawn behind the runner
- Testing the runner is no longer on the starting edge of the road
- I added a box collider trigger around each of road prefabs to detect when the runner enters or leaves the road section
- Created a new script 'SpawnNext' to attach to the road prefabs, started by declaring a public 'destroyTime' float variable, and a 'rC' RoadCtrl variable to reference the 'RoadCtrl' script and finding the game object of its type to apply to 'rC' in the start method, then using an 'OnTriggerExit' method to call the 'SpawnRd()' function in the 'RoadCtrl()' script to spawn a new road ahead, finally a destroy method destroys the current road game object with a delay set in the 'destroyTime' float
- Back in the 'RoadCtrl' script I put in foreach loop inside of the update method, looping through its child roads transform's using their Translate methods to move in the z axis by the value of 'moveSpeed' converted to a negative making the road move backwards towards the camera times Time.deltaTime to keep the speed stable with performance, setting the translate relative to world space making the roads move in the same direction at any rotation

- Testing roads move towards the runner and spawn a new road section every time the player leaves a section as expected

Road Turn Generation

- In the 'RoadCtrl' script I turn the 'road' GameObject variable into an array so I can reference multiple road prefabs
- Declared a public int variable 'rdIndex' to hold the index for the 'road' array
- Used a random range in the update method to choose a random int between 0 and the length of the 'road' array to pick a random road prefab setting this to 'rdIndex'
- Used the 'rdIndex' int to index the 'road' array used in Instantiate method inside the 'SpawnRd()' function
- Back on unity in the 'RoadCtrl' inspector properties I set the straight prefab at element 0 of the 'road' array, the right turn at element 1, and the left turn at element 2
- Testing the random road sections spawn but after a turn the new roads spawn at the same rotation, not at the new road rotation relating to the turn
- I declare a new Vector3 variable 'rotation' in the 'RoadCtrl' script to hold the correct rotation for new roads
- I add an argument to the 'SpawnRd()' function taking an int 'index' to hold the randomised 'road' array index from the time it was called, replacing 'rdIndex' with the 'index' argument throughout the function code, and using the 'rdIndex' value as its argument wherever the function is called (inside 'SpawnNext')
- Added two if statements in the 'SpawnRd()' function after instantiation checking if the 'index' is 1 or 2 relating to road turn prefabs, adding 90° to the 'rotation' Vector3 y axis if it is a right turn, or subtracting 90° if it is a left turn, then in the instantiate method spawning the roads I replace the identity rotation with a Euler rotation using the 'rotation' value
- Testing roads now spawn at their correct rotation after a turn, but the road does not turn once the runner reaches a turn section and roads sometimes spawn overlapping each other after turning towards itself
- I created a public function 'Turn' with a float argument 'yAngle' used for the desired angle, inside the 'RoadCtrl' script, to control when the whole road should rotate once the runner reaches a turn. First the 'RoadCtrl' game object is rotated on the y with the angle provided by the 'yAngle' argument, and then the 'rotation' Vector3 y axis has the 'yAngle' value added to it to update the roads correct spawn rotation
- On the 'Turn' game objects of the road turn prefabs I added box colliders as triggers along the middle of the turns
- Created a 'Turn' script to attach to the 'Turn' game objects, first declaring a RoadCtrl variable 'rC' to reference the 'RoadCtrl' script finding the game object of its type to set in the start method, and a public float 'turnAngle' to specify the turn rotation. With the OnTriggerStay method I used an if to check if the runners z position was equal or greater than the 'Turn' z position to make sure the runner is in line with the middle of the road, if true calling the 'Turn' function inside 'RoadCtrl' with the 'turnAngle' value as the function argument to pass through the rotation
- Testing the turn code repeats itself resulting with the road constantly spinning around during the turn
- Added a bool 'done' with a default value of false to the 'Turn' script for preventing the OnTriggerStay method from repeating, adding an argument to the if also checking is 'done' is false, setting 'done' to true inside the if to prevent repetition
- Testing the road now rotates at turns but sometimes the runner is a little out of line with its previous horizontal road position
- Added an empty game object 'OverlapCheck' to the road prefabs placed slightly in front of the 'RoadEnd' game object and at the middle height of the road barriers
- Created a new script 'OverlapCheck' to attach to the 'OverlapCheck' game objects, starting with a public string function 'TurnCorrect()' that sends a Raycast left and right checking for other road prefabs in the way, outputting as a string the possible turn direction 'left', 'right' or 'forward'

- Next in the 'SpawnRd()' function of the 'RoadCtrl' script I use an if to check the current index is not 0 (0 is a straight road), if true using the 'TurnCorrect()' function found in the latest road's 'OverlapCheck' script to find an unobstructed turn path and setting the 'index' argument to the correct turn, otherwise spawning a straight road if the 'index' is 0
- Testing the roads no longer overlap, but the road turns are very frequent at times
- In 'RoadCtrl' script I declare 2 new variables, one public float 'turnProbabilityThreshold' with a default value of 5 and inspector range modifier between 0 an 10f to control the turn frequency, and a float 'turnProbability' to hold a randomly picked float used for calculating the frequency, using a random range to set its value in the update method
- Then in the 'SpawnRd' function adding to the if checking the index is not 0 (meaning a turn section has been selected) also checking if the randomly picked float 'turnProbability' is bigger than the 'turnProbabilityThreshold' to control the frequency of which a turn section spawns
- In the 'RoadCtrl' inspector properties I set the 'turnProbabilityThreshold' to 6.5f
- Testing the road turns spawn less frequently as expected

Road Generation based on road visibility

- Created a script 'RoadEndVisible' to attach to the road prefabs 'RoadEnd' game objects for detecting if the road ends are visible to the camera, if so, spawning a new road section. First I declared a public bool 'rEVisible' with a default value of false, then used a 'OnBecameVisible' method to set 'rEVisible' to true, and 'OnBecameInvisible' setting it to false
- In 'RoadCtrl' I used an if in the update method to detect if the latest road in 'newRd' is visible using its 'rEVisible' bool found in 'RoadEndVisible', if true calling 'SpawnRd()' using the randomised int 'rdIndex'
- Next, I removed the 'SpawnRd()' call from the 'SpawnNext' script to prevent too many roads spawning
- Testing some of the roads have gaps between the previous section likely from roads now spawning rapidly and too many roads spawn causing performance issues
- Updated the 'nextSpawn' position with the latest road in 'newRd' inside the if checking whether the latest road end is visible to spawn a road, preventing gaps in the road
- Testing there are no longer any gaps between the road sections, but the roads camera visibility check still get triggered even outside of the camera view
- Discovered the scene camera also triggers the visibility check
- Declared a public int 'maxRoadSpawn' variable with a default value of 30 at the top of 'RoadCtrl'
- Inside the 'SpawnRd()' function I wrap its code with an if checking the 'RoadCtrl' transform child count is less than the value of 'maxRoadSpawn'
- Testing the road spawning is now restricted which improves performance as expected
- Renamed the 'SpawnMass()' function to 'SpawnMassStraight()'
- Switched the camera to being a child of the runner
- Renamed the 'SpawnNext' script to 'RoadSection' as its only used to destroy the road parts when the runner has exited its trigger
- Set the 'maxRoadSpawn' inspector property of the 'RoadCtrl' prefab to 75
- Updated the 'nextSpawn' position to the 'RoadCtrl' game object position in the start function of the 'RoadCtrl' script to prevent it from always spawning at 0,0,0
- Created prefabs out of the 'Runner' and 'RoadCtrl' game objects
- Moved the 'RoadCtrl' game object as a child of the 'Runner' positioned at the base of it capsule to keep the road pivot centred and below the runner