**Game Specialism 1 – Practical Portfolio Assessment**

**Programming Specialism**

Joshua Martin

One Drive Link Containing Unity Packages for Briefs:

[Game Specialism 1 - Practical Portfolio Assessment - Joshua Martin](https://stulsbuac-my.sharepoint.com/:f:/g/personal/s4219849_lsbu_ac_uk/EkjRI8cgLVlHs2X9eb7ZxmEB-KlKs2Gk9d2XqZzPAAhHjQ?e=VM9MOk)

**Brief 1: Rolling Road**

The first brief I chose to do was the Rolling Road. I already had an idea of how to tackle this one going into it and for the most part I had stuck to that initial plan when creating my code. Initially the only things present in the scene, aside from the camera and lighting, are; the player, a trigger placed a fair distance behind them and an empty game object with a game manager script attached to it.

Upon running in game mode to sections of road are spawned, one below the player and one at the end of the current section of road, these road move towards and eventually pass the player to create the illusion of movement, each road has a trigger at the start which will spawn another section of road at the end of the current one when the player passes through it. The trigger behind the player will destroy any sections of road that enter it once they are out of the camera’s view.

The scale of the road sections and the speed at which they move are set as public variables within the game manager script so that they can be adjusted within the Unity editor. This initially caused some issues particularly with longer sections of road as the longer sections would reach the destroy trigger earlier and would be removed from the scene before the player reached the next section of road thus deleting the road from under the player’s feet. I resolved this issue by adjusting the game manager script to also move the destroy trigger further backwards proportionate to how the road has been scaled.

Of the three briefs that I completed for this assessment, this is the one that went the smoothest as I was already familiar with most of the techniques I used and it was simply a matter of applying them. However, I did encounter some issues. Road being spawning in is still completely visible by the camera, I had attempted to fix this but was unable to do so before I hade to move on to the next brief.

**Brief 2: Score Ten Pin Bowling**

My choice for the intermediate brief was the Ten Pin Bowling scoring system. I chose this brief primarily for the binary aspect as while I have prior knowledge of binary and how it works I have never previously used it within C# programming before and I thought this brief would be a good opportunity to learn some new skills. Before tackling this brief I had to do some research into learning now binary works within C# but through doing so I learned more about techniques such as bitwise operators, bit shifting and bit count for me to use in my code.

An issue I encountered with this brief was with how ten pin bowling is scored. Going into this one I was not super familiar with the game’s scoring system and while learning how it is scored was easy enough through some research figuring out how to implement it through my code was where the difficulty lied. In order to accomplish this my code include many integer and Boolean variables that are being constantly referenced and changed by the code in order to produce the desired result such as checking and calculating the scores for strikes and spares.

Of the three briefs this is the one that took me the longest as there is a lot of code within my scripts, while the code functions. I believe there is a more efficient way to accomplish this task than the way I did so. I could’ve also made a lot better use of function to reduce the amount of lines of code within my script as there are plenty of repeated parts of code especially when determining the score for the second ball in a frame.

My code is also not without issues that I was unable to resolve in time. One major issue is that the code does not properly check what pins were already knocked down by the first ball thus making them out of play resulting in returning scores where the total number of pins knocked down within a frame exceeds ten pins. Another more minor issue is with how I create the score, the scores are created by generating a random number and then converting it to binary, however as the random number generates any possibly binary number with eleven bit this results in the code returning a lot of fouls and making strikes extremely rare.

**Brief 3: Edge Detection Shader**

The final brief I had decided to do was the Edge Detection Shader. Like before with brief 2, this one also required some learning and research as I had not previously worked with shaders within Unity before and, unlike binary, I did not have any prior knowledge of how they work. Once again I considered this brief a good to chance to learn some knew skills and while I was able to learn more about shaders this also had a downside and I had to spent a lot of time learning and did not spend as much time actually working towards completing the brief resulting in poor time management for this last brief.

This time around I created the project using the Universal 3D core as opposed to the standard 3D core hoping it would help with creating the shader as I would be able to make more use of universal render pipeline. When actually creating the shader I struggled to fully get a comprehensive grasp on shader code due to having to learn it from scratch and instead used a shader graph to alleviate this issue, this however still required me to learn how shader graphs worked and resulted in some more research in how to get the desired affect through a shader graph. I had also attempted to view code generated through the shader graph to see if I could learn anything from it and recreate the shader through code however the code for the shader showed multiple errors despite it working fine as a shader graph. I was unsure why this was the case so I opted to leave it alone instead so I didn’t accidently break anything.

Despite my issues with creating the shader, I eventually managed to achieve the desired effect with adjustable parameters for both the colour of the outline and it’s thickness however it is not perfect. However, I do believe this brief is the most helpful for the future out of the ones I had chosen as I can transfer what I learned about shaders to future projects.