Section 5—Problem Solving Capability [20 Marks]

13. Write approximately 500 words, describe the major game development problems you have overcome in your 2D sprite project. Please describe your solutions to these problems.

I think the main problem in the sprite project for me was learning to work in a team environment where every member of the team had different skill levels, different specialisations, and understanding how to integrate with the different sets of code written by each member.

I found understanding the architecture of the game that my other team member designed was extremely difficult at first, not only because it was a large project with many levels of relation and abstraction but also because there were a few unknown factors like how exactly to work with the tile data and other such variables that would contribute to the core architecture.

To give more detail, the main areas that I found were tricky were learning how best to input the sprites and their respective animations into the current framework, how to develop a projectile that would fire from the main character upon attempting to damage an enemy, how best to implement the spritesheet in a way that would allow for easy coding and integration / use in the project. Furthermore, using the Visual Studio Git plugin was something we believed would make development and version control easier, but was in a lot of ways more trouble than it was worth, as it had a lot of occasions where it failed to properly merge code, corrupted commits, or just generally broke the code as a result of conflicting versions. Upon being recommended to use TortoiseGit instead, a lot of these issues became a lot more straightforward to deal with and each team member had a lot less problems integrating commits together.

On the whole, really all that my main issue required doing was just spending long hours not actually coding, but just spending that time trying best to read through the code and familiarise myself with what was actually going on. It felt to me like it was not actually being productive because nothing tangible was actually being done, but in retrospect it was really the only reason I was able to contribute anything at all; after all, if you can’t understand the framework you’re working in, there’s really no way to use it for any purpose. It also took asking my teammates to try and explain as much of it as possible, but I underestimated how in the dark my teammates were regarding the project themselves, so while it did help understanding the basic relationships between classes, it didn’t help particularly in terms of understanding how the actual code functioned to do what it did. That was something that still needed to be understood on an individual level; something I needed to figure out for myself.

In the end I familiarised myself with enough to be able to contribute comparable levels of work as my teammates, and also by familiarising myself enough, I was able to work on more than just my main responsibility (being sprites) and focus on the game as a whole and just try and add features or fix issues as I saw them arise. It was a valuable experience in the sense of learning how to work in a team code-wise, how to go about a team-based project’s workflow process, and how to more systematically familiarise myself with new code before developing code for it, and I feel I gained a lot of experience as a software developer in doing so. While the project itself was in some ways missing some things that would perhaps make it a more polished game, something I’ve realised in the process of making games is that there’s always something that can be improved; and if you kept trying to fix or improve every time you found something to fix or improve, the game would quite literally never be released.