SURNAME: Moodley

FIRST NAME: Tenisha

STUDENT NO.: 2105982

COURSE CODE: WSOA3003A

ESSAY TOPIC: Micro Project 1: Data Design of a Turn Based Combat Game

The Intent

The intent behind creating this prototype was to create, engage with and design the beginning core system of a turn based combat game. This will be done in order to observe how data design in turn based combat games require users to interact with the system and in return, how the system then reacts to a user's interaction.

Furthermore, this prototype will be using a technique known as a state machine, a system that allows the prototype to move between the player and other system states, such as moving from the player turn state to the enemy turn state once the player has concluded their turn. This is essential to creating the allusion of turn play within the system. While the system itself stays the same, the 'state' of the system can change, and this manipulation of data and communication is the main objective of this prototype.

The Process

User Interface (UI): This will be used to allow the players to interact with the system according to the system's data design, such as with Text box's and numbers via life bars and UI buttons.

Environment: will entitle the essential aspects or game bits of the system, such as a player and an NPC, who will therefore need battle stations of some sorts to represent their position in the game.

Game play: The alternation between game states are what is used to distinguish the system's progression. This games states would consist of; the start, player 1 turn, NPC turn, won, and lost. Turn based systems have combat, and combat are turn based. In order to create this turn based combat system, the system needs to move between data recognised as the states of the system, 1 state at a time.

Logic: these are the core system scripts we will need to create, a unit script for the player and NPC's information, a battle system script for the game states and other battle information, and a battle Head's Up Display (HUD) script for updating the unit UI.

Reflection

After establishing my intent towards this prototyped system, which was to create a turn based combat system which explores the possibilities of data design, I started the implementation of the system design process. I started by focusing on creating the UI and feedback designs to communicate the system to the players and afterwards working on the core systems or scripts of the system to accompany the UI. However, I realised that by doing this I over scoped the idea of my system and was left with numerous scripts that could have been designed more data efficiently, meaning that I could have used arrays, loops and PlayerPrefs instead of coroutines and static variables as these types of hard coding leads to more lines of script and conditioning that is data inefficient.

The NPC is wired to understand when it needs healing or when it should lower the players accuracy. Although, the NPC's system is not extremely advanced or particularly inventive, I still believe I ran out of time trying to finish this project with the scope and lack of time as is, and therefore more advanced data designed code was avoided.

Lastly, although the defence statistic has been implemented into the system, the defence does not actually work. For the time being it simply shows what data manipulates it but concerning the ability of a defence statistic, the defence does not actually weaken attack damage. This issue was realised late into play testing and the lack of a solution is due to insufficient time management used during the design of this prototype. A quick fix to this issue however, was to sightly increase the user's attack damage statistic whenever the opponent's defence rose.

A table showing what the interacted data, in the prototyped system, can effect, and how or why that data is being influenced

What the interactive data can effect:	How, or why, the data is being influenced:
What state the system is in	The system state would need to change in order to encourage interaction between the player and the NPC, using a state machine
How the system communicates state change to the player	It's important to relay feedback to the player about state changes because the player needs to stay aware of how to strategically react to the NPC's actions, and the player won't be able to if they can't figure out what state the system is in. This was accomplished by disabling the players buttons during the enemy's turn
What happens when player wins/loses	Level increase, new moves unlocked for that specific level, the player becomes more powerful but so does the enemy
If the player/enemy is attacked or if the player/enemy attacks	If the enemy's accuracy statistic is equal to or greater than the player's evasiveness/speed statistic
If attacks have missed	When accuracy is less than evasiveness/speed was the previous way of dodging, however a another system was added onto this where the dodging is randomised throughout gameplay as well to introduce players to a level of game flow that comes with randomisation

What the interactive data can effect:	How, or why, the data is being influenced:
What happens when statistics are maxed out negatively or positively, such as that the player's life points can no longer be healed or if the enemy's accuracy has reached 0 percent accurate	Messages will appear to communicate such situations with the player, and the statistics won't proceed further than the minimum or maximum values. It's important to communicate this with the player because if the player doesn't get communication on this they could keep wasting turns instead of winning the system