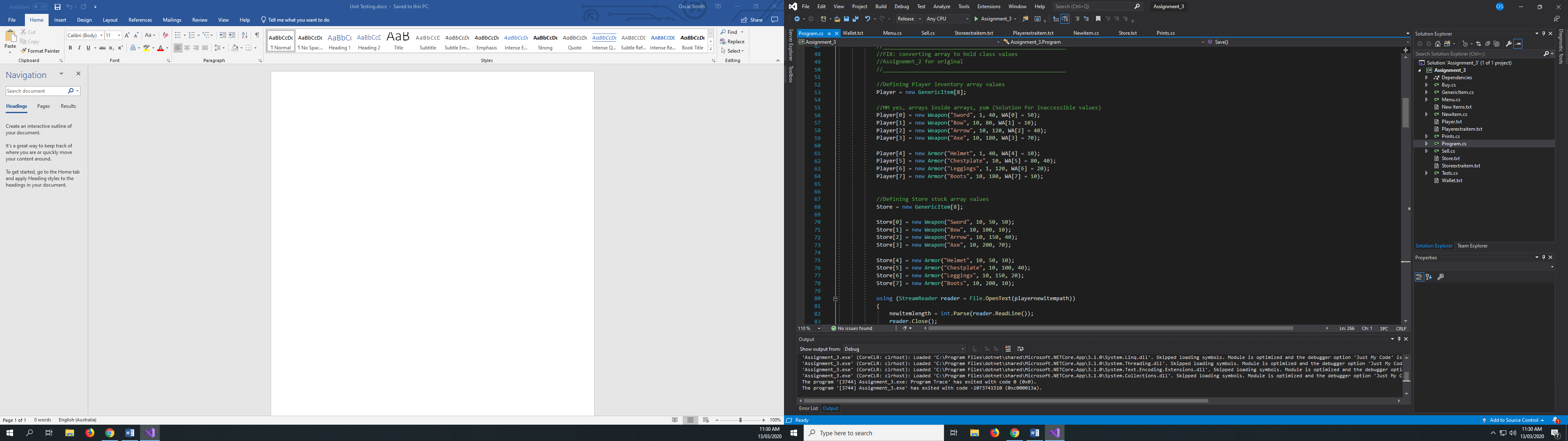
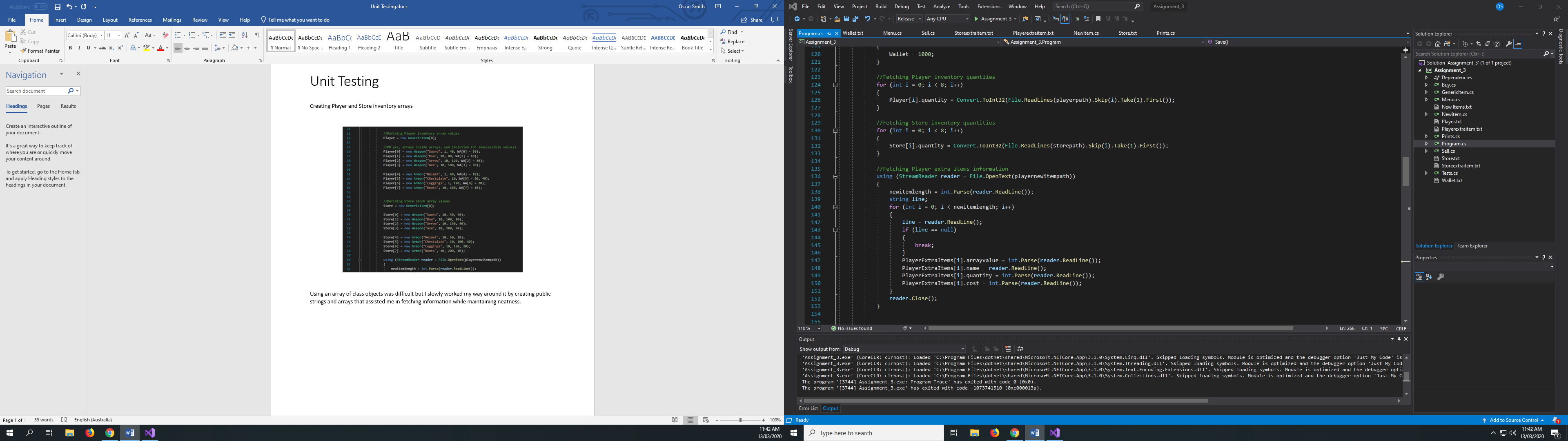
Unit Testing

Creating Player and Store inventory arrays

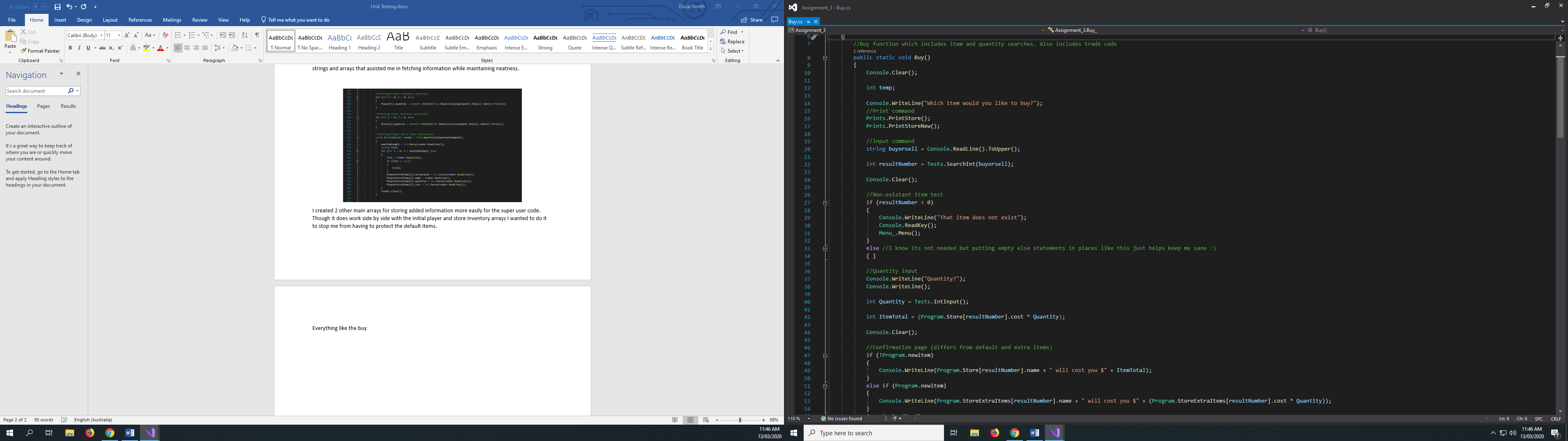


Using an array of class objects was difficult but I slowly worked my way around it by creating public strings and arrays that assisted me in fetching information while maintaining neatness.



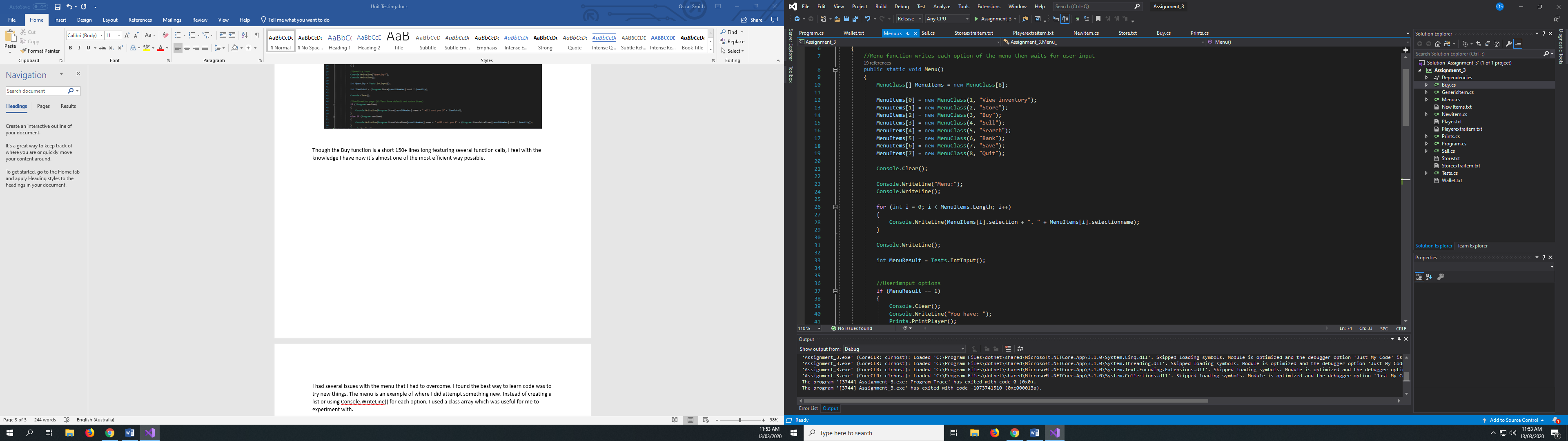
I created 2 other main arrays for storing added information more easily for the super user code. Though it does work side by side with the initial player and store inventory arrays I wanted to do it to stop me from having to protect the default items.

Everything like the buy function are somewhat simple though take up a lot of space. I found that the use of functions to outsource lines and decrease on repetition became useful. Also, then using functions universally made adding additional code easier. As I tried to make everything in the application as break proof as possible there are many if else and for loops in the sequence.

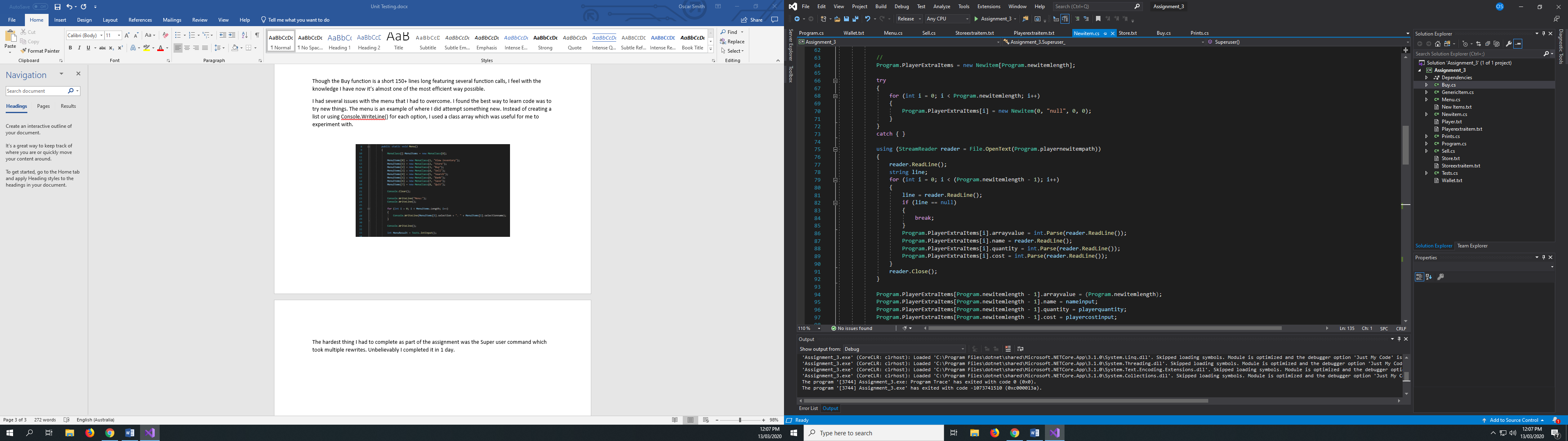


Though the Buy function is a short 150+ lines long featuring several function calls, I feel with the knowledge I have now it’s almost one of the most efficient way possible.

I had several issues with the menu that I had to overcome. I found the best way to learn code was to try new things. The menu is an example of where I did attempt something new. Instead of creating a list or using Console.WriteLine() for each option, I used a class array which was useful for me to experiment with.



The hardest thing I had to complete as part of the assignment was the Super user command which took multiple rewrites. Unbelievably I completed it in 1 day.

s

The Super user command includes inputs outlining the Name of the item, the cost for both buying and selling and the player and store quantities of the item. Each time an item was added the application would also automatically save the information to a text document which also avoided some major errors in the item creation. Items can also be later deleted