Evaluation

For my final year at secondary school, I made a chess program. There are lots of similarities between chess and battleships (2d arrays, turns that only pass on user input), so this gave me a chance to work on the weaknesses I had last year.

The biggest improvement I wanted to make was using object oriented principles more effectively. The chess program became this huge monster with thousands of lines of code, and I figured the cause of this was probably down to not utilising the OO concepts I had learned correctly. So, for this project I tried to approach the problem with a more OO view, and the code this year is far more understandable.

I also had one class doing too many things last year, so I wanted to balance the responsibilities across the classes better this year. This was fixed by the candidate classes – determining the classes to be used from the requirements meant each class only took on its own responsibility, which led to a much more even amount of work being done by each class.

Despite not having time to implement multiplayer, the way I wrote the program should make it easy to do. I already have two boards implemented, ones just not in use. So, from what I can tell I would just need to alter the methods to read from/write to board[0] or board[1]. The player turn could be kept track of using a boolean value, so if the methods referenced board[playerturn], then I would just need to toggle the playerturn at the start of each turn.

Another part of the code I am pleased with is the use of String arrays when processing the input. This is a much tidier way of comparing many values, which can be a problem when using coordinate systems.

Also, the use of a buffer when placing random ships means that if the placement becomes invalid half way through, the process can just be restarted without having to clean up the board. Then, the whole ship can be placed at once if it’s valid.