SOFTWARE ENGINEERING – Assignment 2 (Nabeel Mirza & Simon Garland)

- To begin, we used GitHub to contact each other and then both worked on the one file instead of committing constantly.
- We then began to look at the design requirements and started to make a plan to divide up the initial work load. Nabeel decided to do the first couple of requirements all to with players while Simon decided to do the slots part.
- In the main of the programme you will initially see the players part, where it prompts the user to enter a number of players between 1-6, while if a number not between 1-6 is entered, in a “while” loop it prompts the user again to enter the number of players correctly between 1-6. Then the programme asks the user to enter the names for each player. This is done using a while loop moving through the amount of players assigning the name entered to each one. This is followed by prompting the user with another question to choose which type of character they desire for each player, similarly done as the previous while loop. All of this information is then stored in an array at the beginning of the programme outside of the main.
- The programme then moves onto the next part of the project where the players capabilities are set using the rand function. For this the time library was used and the line “srand (time(NULL));” had to be included. The way the player’s capabilities were assigned was using a for loop moving through the number of players and then randomising the players e.g. Luck between the limits for the specific player.
- The programme then prints out each player and current capabilities.
- The programme then like players, asks the user to enter the number of slots wanted for the game between 1-20. However, in this case if the number of slots is greater than 20 or smaller than the number of players playing the user is told to re-enter the number of slots correctly.
- The slots are then all assigned a number between 1-3 randomly and then a type is assigned to the slot depending on its corresponding number e.g. 2 = Hill. To do this the strcpy function was used. It wasn’t fully necessary to use it but it made it a lot easier because, if we used the numbers 1,2 and 3 we would have to remember which each number means constantly.
- After this a new array was assigned to the max size of the slots so that only one player can be assigned to the slot. Then all the players were assigned randomly to the slots.
- Moving onto the next part of the code, we decided that it would be too hard to do bit by bit and trying to implement each-others code so instead we sat down together and worked on it at the same time which proved to be a lot easier and more efficient.
- This part of the code begins with a while loop. Inside this while loop the user is initialling asked which action they would like to carry out.
- The next part of the code is the moving and attacking section where the user is prompted with a choice to either move to the previous or next slot, or else to attack.
- For the different choices the user could make we implemented different if conditions. In the same way we were able to make conditions for non-existing previous or next slots and when another player was close to be given the choice to

attack. After a player is attacked or attacks the different cases of changing life points happened.

- Finally at the end of the programme once the users action was carried out a printf statement prints out all the players, their types and their according lifepoints.