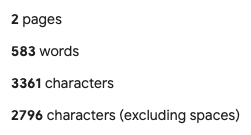
**Team Members:**

Mohammed Tokaria - 180008589

Max Kelly - 180004073

Ramsay Sewall – 170012517

**Words count (Inc. headings)**

****

## **Our Approach**

During our first meeting we initially decided to split the work, one of us would work on the design whilst the others work on the code. But after the first couple of C# files we realised that our code is completely different from each other and it was causing issues will compiling it together. Then after we all decided to work on everything collaboratively as it makes it easier to compile and keep a track of what needs to be done and what we have achieved so far. Since a couple of our group members use MacOS, we were unable to use Visual Studio and so were forced to work in VS Code, while the Windows user compiled our code in Visual Studio as well as designing the Windows Forms.

## **Problems faced and how we overcame**

Whilst working on this project, we came across many difficulties as this was our first project in C#. Below are the problems we faced, with the respective solution that we came up across.

1. **Checking for a valid move**

A large portion of the development of the project, was consumed with figuring out how we would turn a square grid of Windows Forms buttons, into a grid of Checkers. After the grid was set up with the initial positions, we had to focus on the validation for each move. When the current player would select one of their pieces, the valid moves for the respective piece would be highlighted, allowing the player to go forward with their choice of move.

1. **Jumps**

One of the main features of Checkers is to “take” an opponent’s piece. When the player’s checker is diagonally opposite the opponent’s, and there is a blank position on the other side, the player would be able to “Jump” over the opponent’s piece, removing it from the game and increasing the player score by 1. This feature required extra implementation on the “standard” move as we needed to check if the place was forwards and if the piece is ‘king’ it will also allow, to move backwards.

Whilst making the algorithm we also faced issues with the jumps where, the player was able to take a checker, but was also removing non-adjacent pieces of the opponent from the board at the same time. After doing a bit of research and debugging the code, we were able to overcome this issue.

1. **Visual Studio / VS Code - Live share issues**

Whilst working on the project we encountered a corrupt Visual Studio update that caused issues with the collaborative working process. This delayed some of the implementations and relied on us working harder the following day after the new update was released.

1. **Artificial Intelligence**

After completely accomplishing all the issues we had whilst working on this project, we decided to work on the AI so that our game is available to play in single player mode. But even after having several tries, we were unable to get the AI to work.

## **Reviewing**

As this was our first coursework in C#, we worked really well as a group and accomplished a solid game. We worked especially hard to ensure that players would not encounter any unhandled errors or bugs. All the members contributed in coding and designing the form even though some members were unable to work in Visual Studio due to different Operating Systems.

## **References**

<https://stackoverflow.com/questions/1669318/override-standard-close-x-button-in-a-windows-form>