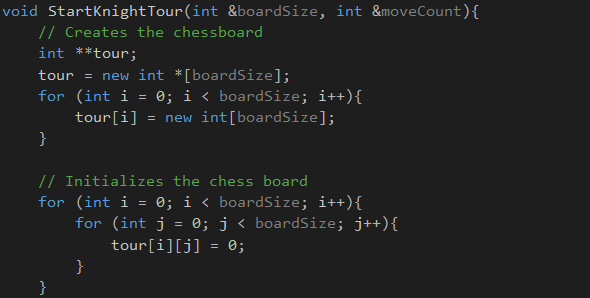
**Knight’s tour report**

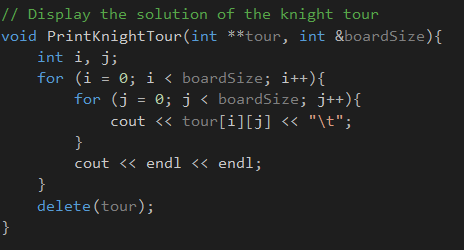
Name: Siem van der Kluft  
Studentnumber: 500709315  
Class: IVGT2

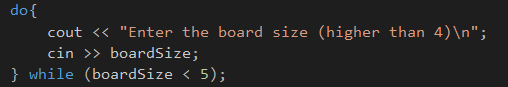
**Solution**

The solution I found for the knight’s tour is a brute force solution. The program uses a recursive function to calculate a tour for the knight. The recursive method has couple of parameters. These parameters contain the 2-dimensional array that contain the data of the knight tour, the board size, the moves the knight can make, the current position of the knight and the move count of the knight. Every time the knight moves the program checks if it can go there and if it can it adds 1 to the move count. This way it can be displayed in the terminal from 0 to the amount of steps taken. If the knight can’t move to that location, it selects another move it can make. This is done until there are no options left. This is my solution for the knight’s tour.

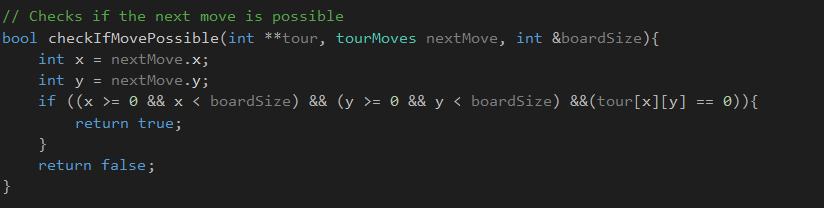
**Requirements**

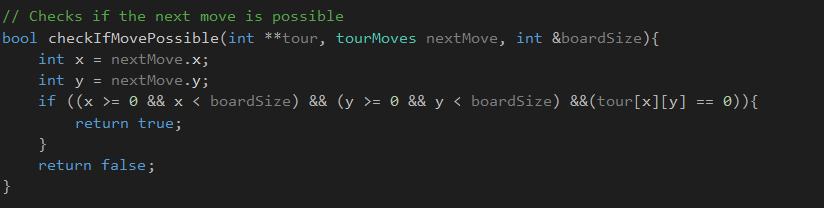
**The program makes use of pointers and references:**  


**There is a visual representation:**The knight’s tour has a visual representation. Printing is done by looking up all the values of the tour array which contains the data from the knight’s tour.  


**The board is at least 5x5 in size:**The user can set the board size, but it checks if the size the user choose bigger is than 4. If that isn’t the case the program asks it again and the user needs to fill in a new value.

**The knight has to move according to its movement rules:**I set all the possible moves the knight could make and the program checks if the knight doesn’t move outside of the board. So if the knight moves outside of the board the move is not valid.  

**The knight can only visit each square once:**The program checks if the spot where the knight wants to move has been visited already. This is done by checking if the value within the tour array equals zero. If the value equals zero than it isn’t visited, else it is visited and the program uses another spot and does this same step again.

**Exemplar**

**Board size can be set by the user (NxN size):**The user can enter the boardSize value, the program checks if it is a correct value. If it isn’t a correct value the user needs to enter the value again. After that the program creates the array. This is done by creating array of pointers to arrays. This way the 2 dimensional array can be created dynamically.

