Knights Tour

Methods and Techniques audit

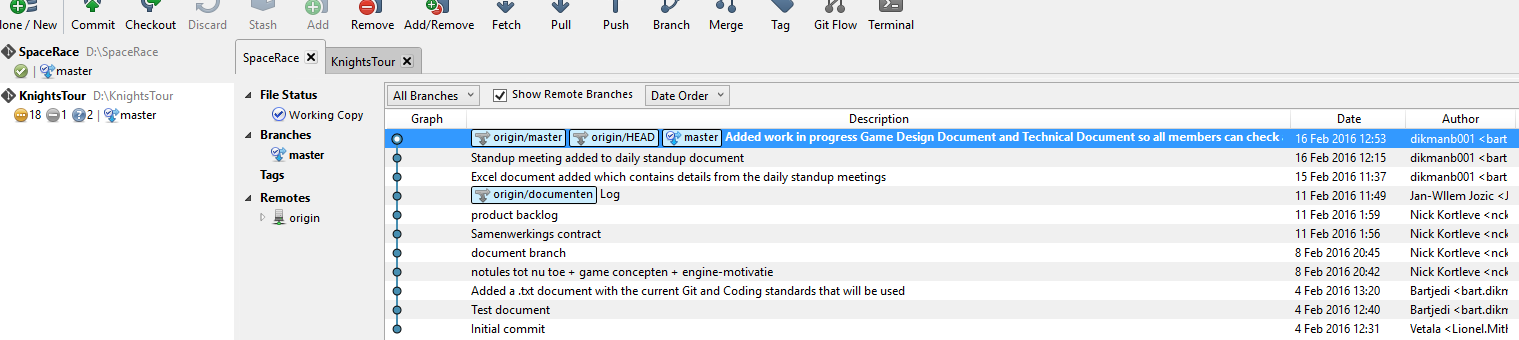
Bart Dikmans

500681415

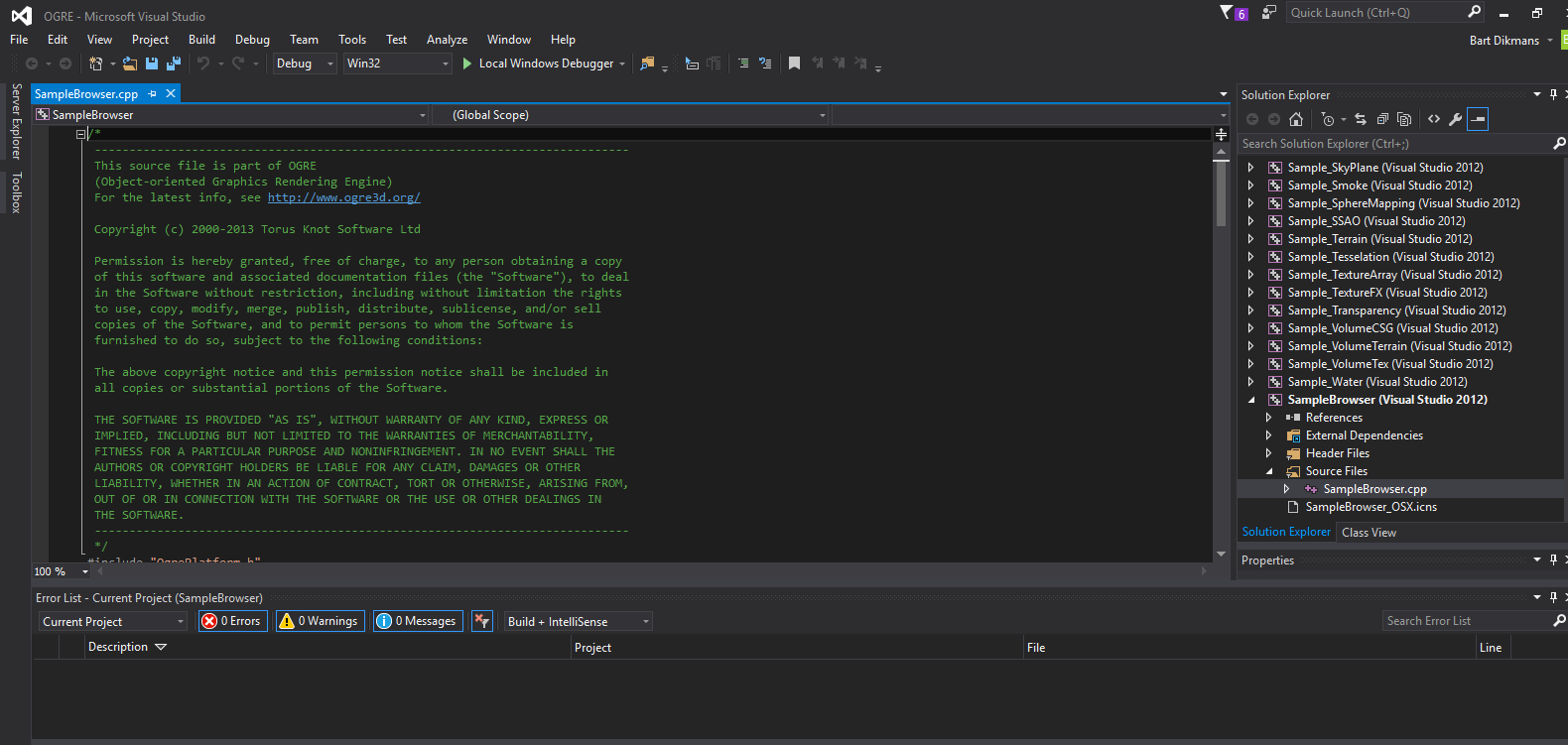
17-02-2016

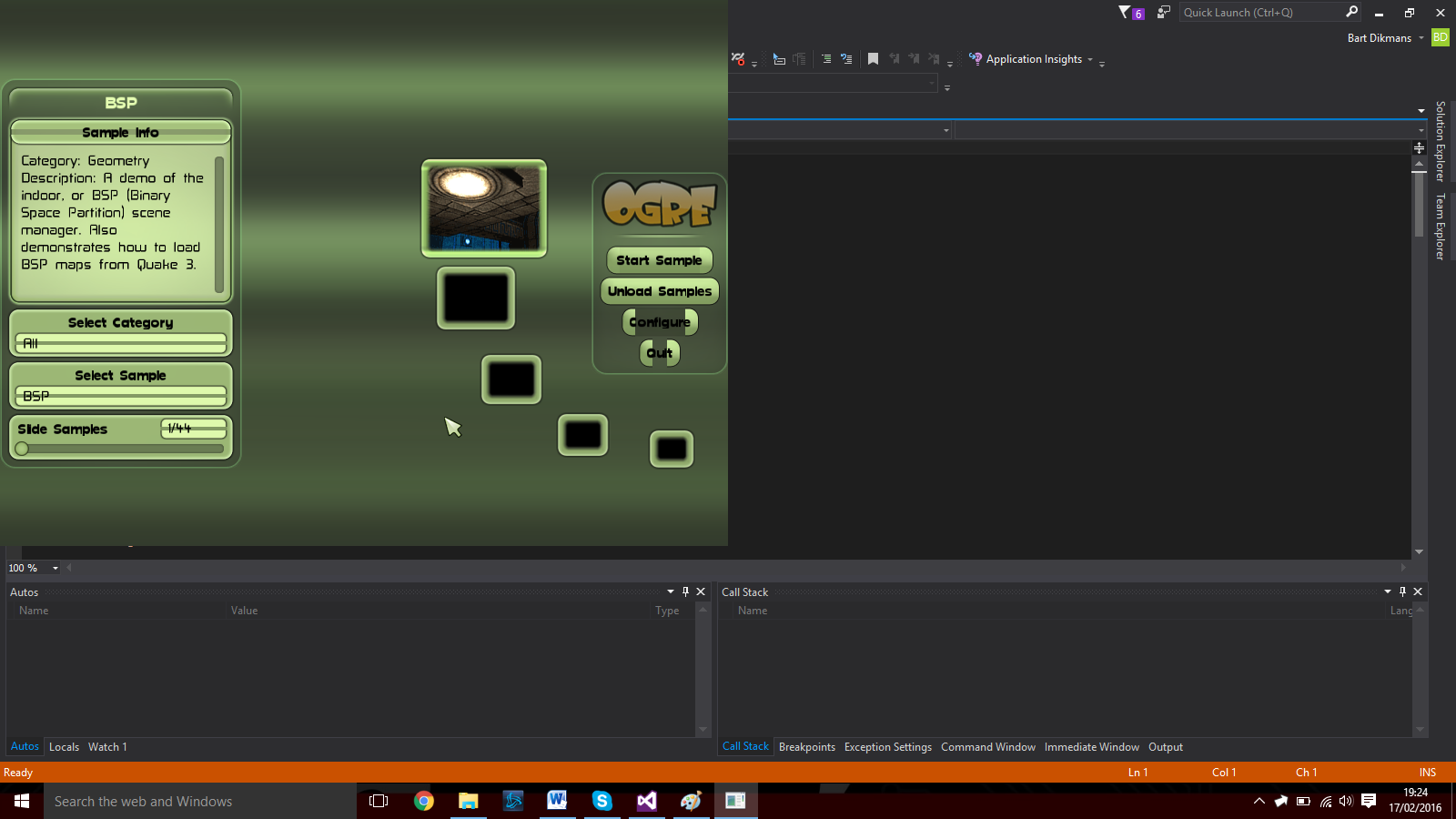
# Screenshots of working GIT

Screenshot taken in Sourcetree on 17-02-2016:



# Screenshots of working Ogre3d Engine SDK





# Requirements

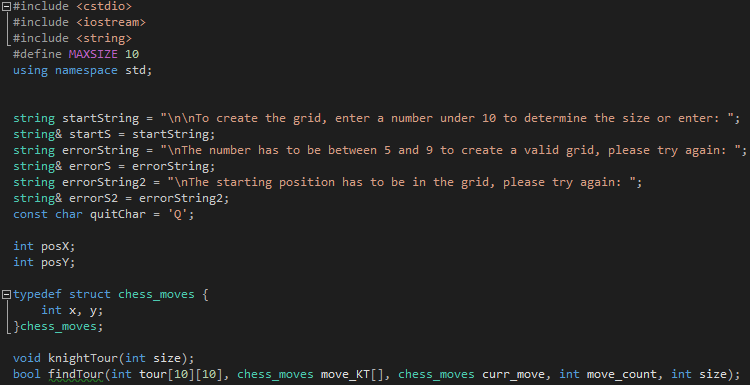
|  |  |
| --- | --- |
| Programming language has to be C++ | The entire program is written in C++ |
| The Program makes use of pointers and References | There is a pointer and there are multiple references as you can see in the create function and in the first screenshot |
| There is a visual representation | There is a visual solution in the cmd prompt through an ASCII method |
| The board is at least 5x5 in size | The user can insert input that has to create a grid that is at least 5x5 in size as can be seen in the create function |
| The knight has to move according to its own movement rules | The knights moves are defined in the knightTour function and are representative of the knight moves in chess |
| The Knight can only visit each square once | The algorithm can only visit every square once |
| **Exemplars:** | **Exemplars:** |
| Board size can be set by the user | The user can set the board size in the create function. |
| The user can decide where the knight starts | The user can decide where the knight starts in the create function |
| Uses a more efficient solution than brute force | The backtracking algorithm technically uses brute force through all options until a solution is found. So this exemplar isn’t fully met. |

# Code Explanation

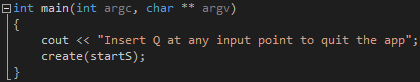
These are the includes and used variables. MAXSIZE has been defined as 10 at the top so that changing this becomes simpler.

The next few lines are all strings which are used for error texts and as a welcoming text. We also define the character which will quit the application as the constant character ‘Q’. next to that we declare two int variables which will contain the starting x and y positions that the user wants.

After this we define the chess\_moves construct and tell it that it will always require 2 variables. The X and Y. After this we also mention 2 of the functions we will be using and define which inputs they will need.

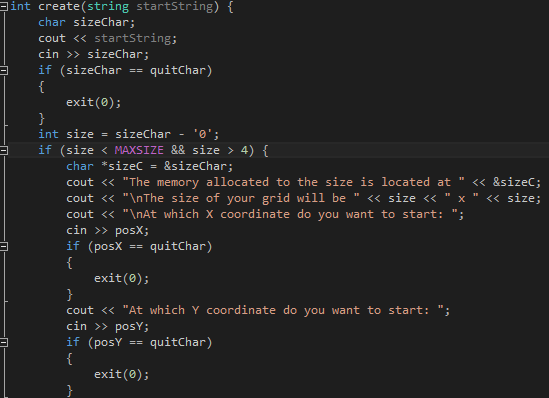


The main function is called upon once and immediately calls upon the create function sending the reference to the startString string.



This is the create function. The first line prints the string parameter and checks for the users input. If the input is the quitChar (Q) the program will stop. This happens multiple times during the create function due to it having multiple opportunities where the user can give inputs. We also only get the first character from the users input. This is there to help make sure the users input is a valid input because It limits the possible inputs.

After checking the input it converts the input char to an integer. This requires the –‘0’ to make it a real number. We check if the number is between the boundaries we have set for the grid. After this we get the starting coordinates from the user.



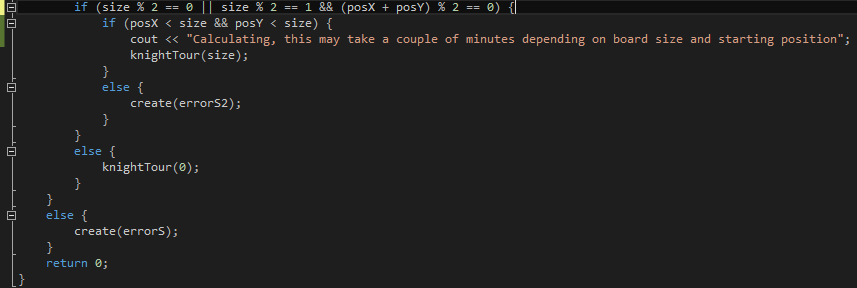
After getting all the inputs we check if the combination of inputs generates an impossible knights tour. This predetermined setup can only happen if both the size and one of the position inputs are an uneven number. For example: if the board size is 7x7 and the starting position is (0,1) both the size and one of the starting positions are uneven. The following 3 else statements are as followed:

If the x and y input positions are incorrect the second error message is printed and the create function starts again.

If the combination of inputs generates an impossible knights tour, the knightstour Is started on a 0x0 grid resulting in an instant message stating that there is no possible solution with the inserted variables.

If the size is bigger or smaller than the predefined variables for the minimal and maximum sizes of the grid the create function is restarted with the first error message stating that the grid size was invalid.

If all the valid inputs are correct the knightTour function is called and the input size is send as a parameter.



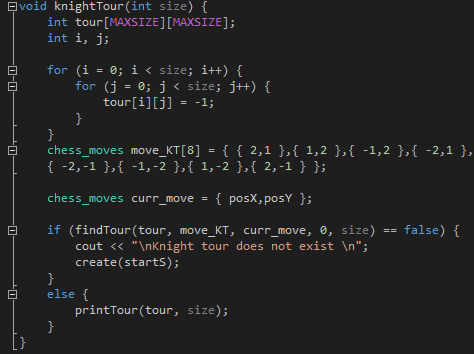
If all the inputs are correct the knights tour algorithm starts. At first the array tour is made and gets a maximum memory size. After this it generates the grid depending on the size parameter. After this we define the possible moves the knight can take aswell as the starting possition .

The move\_KT[8] is of the struct chess\_moves so it requires a X and a Y int variable. It is also an array of 8 elements because the knight has 8 possible moves.

The curr\_move is the starting position, it requires the users input for an x and Y position to fill the prerequisites set by the chess\_moves struct.

After these variables are set the recursive function findTour is called with the grid, possible moves, starting position, move count and the user defined size. If this function returns false this means there is no possible knights tour resulting in the output line aswell as the program starting over again at the create function.

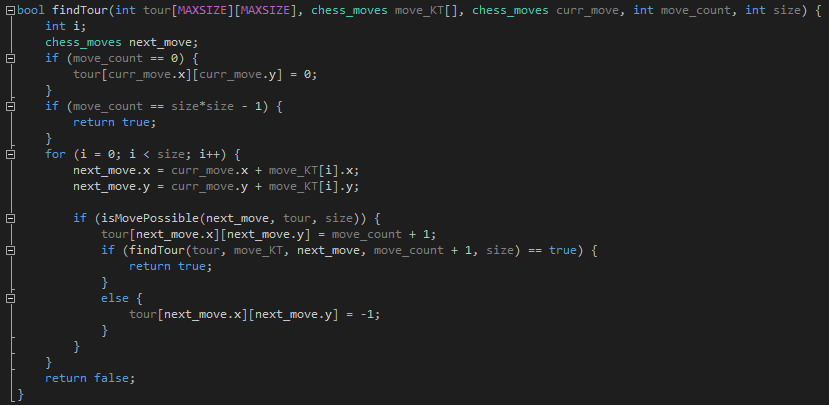
If there is a possible knight tour the program prints the knights tour grid.



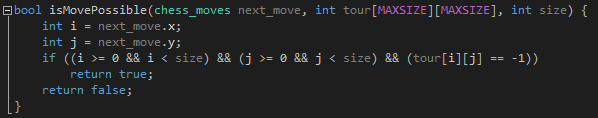
This is the findTour function. Its parameters are the grid that will be used, the possible moves, the starting position, the move count and the size that the grid will be.

If the first move has yet to be done the program recognises that this will be the starting position and marks it accordingly. If the program sees that the move count has reached the required number of moves it ends the loop.

When neither of these to conditions are met the function checks for the next possible move. If this move is possible it saves the location and marks it in the grid. If the program recognises that no moves are possible it goes back to the move before this to execute further from that point.



In this function the next move is checked. The next move is possible if the move remains on the board and if the location on the grid is currently marked -1. This means that the program can only mark a spot op the grid once.



This function prints the finished grid if a knights tour is possible. It has 2 for loops which cycle through every position in the grid. The if statement decides what kind of layout will be used for that spot on the grid. After printing the grid the create function is called so that the program can start over again.

