P1 Knight's Tour

Due Jan 18 by 3:20pm **Points** 100
Available Jan 8 at 12am - Mar 16 at 11:59pm 2 months

NOTE: *Please try to solve this problem by working on your own or by talking with your classmates. This is a very famous problem, and so there are solutions to this problem in many books and on the web, but if you solve the problem by looking at those solutions then you robbing yourself of the joy learning and accomplishment of struggling with a problem/concept and finally gaining the victory over the problem.*

Knight's Tour:

In the game of (International) Chess the knight moves in an unusual pattern of 2 squares horizontally and then 1 square vertically or 2 squares vertically and then 1 square horizontally. The 8 possible moves for a knight are shown below:

		8		1		
	7				2	
			Kn			
	6				3	
		5		4		

Somewhere along the line the question was asked: Could a knight move to every location on a chess board without ever landing on a square more than once. This challenge is known as the Knight's Tour.

Your assignment is to come up with a recursive solution to the Knight's Tour. As your knight travels the board record the squares that he has touched by storing the move count, so for example the first square that the knight starts from would be marked with a 1 and the next square that the knight moves to would be marked with a 2 and so on until on a normal 8x8 chess board the knight moves to the last square and stores a 64 there.....the knight would have them completed his tour. For this assignment, it is ok to use some global variables for recording some of the statistics you may want (move count, tries, board, ect).

Remember in recursion you call a function to solve a problem, then the function will call itself to solve the next step of the problem. This will go on until the function gets to some ending solution. In this case, your function would be a move for the knight, and it would call itself for the next place to try moving to, which would call itself for the next place to move, and so on, and so on.

Programming Note:

- Think about what information needs to be passed into the function and what information needs to be returned from the function.
- What are the things that you need to do in a square?

For output from the program, you need to print out a map of where the knight visited in his tour. Here is an example of a Knight's Tour for a 5x5 board starting from the upper left corner:

```
Yeehaw!!! after 41 tries
and 16 bad moves
1 20 17 12  3
16 11  2  7 18
21 24 19  4 13
10 15  6 23  8
25 22  9 14  5
```

I do also want you to display the number of tries (these were squares the knight actually moved i.e. an empty square on the board).

Debugging Suggestions: For debugging use a 5x5 board to see if the recursive calls are working. This is nice for following the algorithm for a few moves. Have the board print out after each move when you are first testing to verify that your algorithm is working. Once you are convinced your algorithm is good, you can have the board printed periodically to give you an update (I had mine print out after every 10 million tries... some of the tours on an 8x8 board can take a while.... starting at location 4, 4 (when first row and col are 0) the program ran over night and had tried 62,200,000,000 moves and had a ways to go before solving it with this brute force algorithm). The solution for a tour starting at 0,0 did not take that long; and only took 3,242,065 tries to find the solution.

I solved the problem by using a function prototype like the following:

```
bool moveKnight( int row, int col, int movNum);
```

The boolean return value I used as a signal of the results of the function; false meant this call could not move further, true meant that the end was found (eventually) from this move. This is just a way of thinking of it, and your are not required to have the same function prototype.

Turn in: A paper copy of your executable with a print out of the tour taken by your knight on an 8x8 board as a comment below your program.

Global Data: Usually I do not allow global data that is not constant, but for this program you can make the

board and the number of tries as global variables.

Ways to lose points:

- if your file does not contain the program header with a program description and short function descriptions to accompany the function prototypes(for functions used with main).
- your code should also be consistently indented as talked about in class, and shown in the book, and class
- you should use good variable names (descriptive, and start with lower case letter)
- proper placement of { and } (a } should not be placed at the end of a line)
- no staple to keep your papers together (folding a corner or using a paper clip are not good enough)

Comments: Comments are a way of documenting a program (explaining who did what and how). All programs for the rest of the course are required to have the following header documentation and inline documentation to explain any tricky pieces of code.

```
////
// Author:       Your Name
// Section:      A or S
// Assignment:   #
// Description:  Short description of what the program accomplishes
// (at least a couple sentences – also you should say where the input
//  data comes from, what information is output, and where the output
//  data is sent to (the screen or a file)
////

#include <iostream>.......the rest of the program
```