KIT107 Programming **2019**

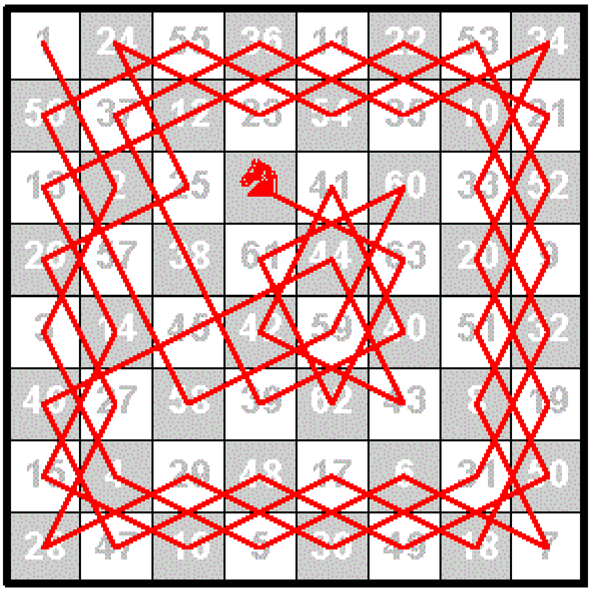
Assignment 3

Due Date

The assignment is due at 3PM Wednesday October 16th 2019 *and should be completed in pairs using the Pair Programming technique* and not by division of labour. Use the "Assignment 3 Pairs" groups on MyLO to find and register your pair.

Context

The Knight's Tour is a challenge to move a knight from the board game of Chess around the board landing on all squares but reaching each square only once. Knights move in an 'L' shape (of two squares in one direction and one square at right angles to it or vice-versa).



Task

Your task is to implement the *Knight's Tour* problem in C so that the computer will find the solution for you.

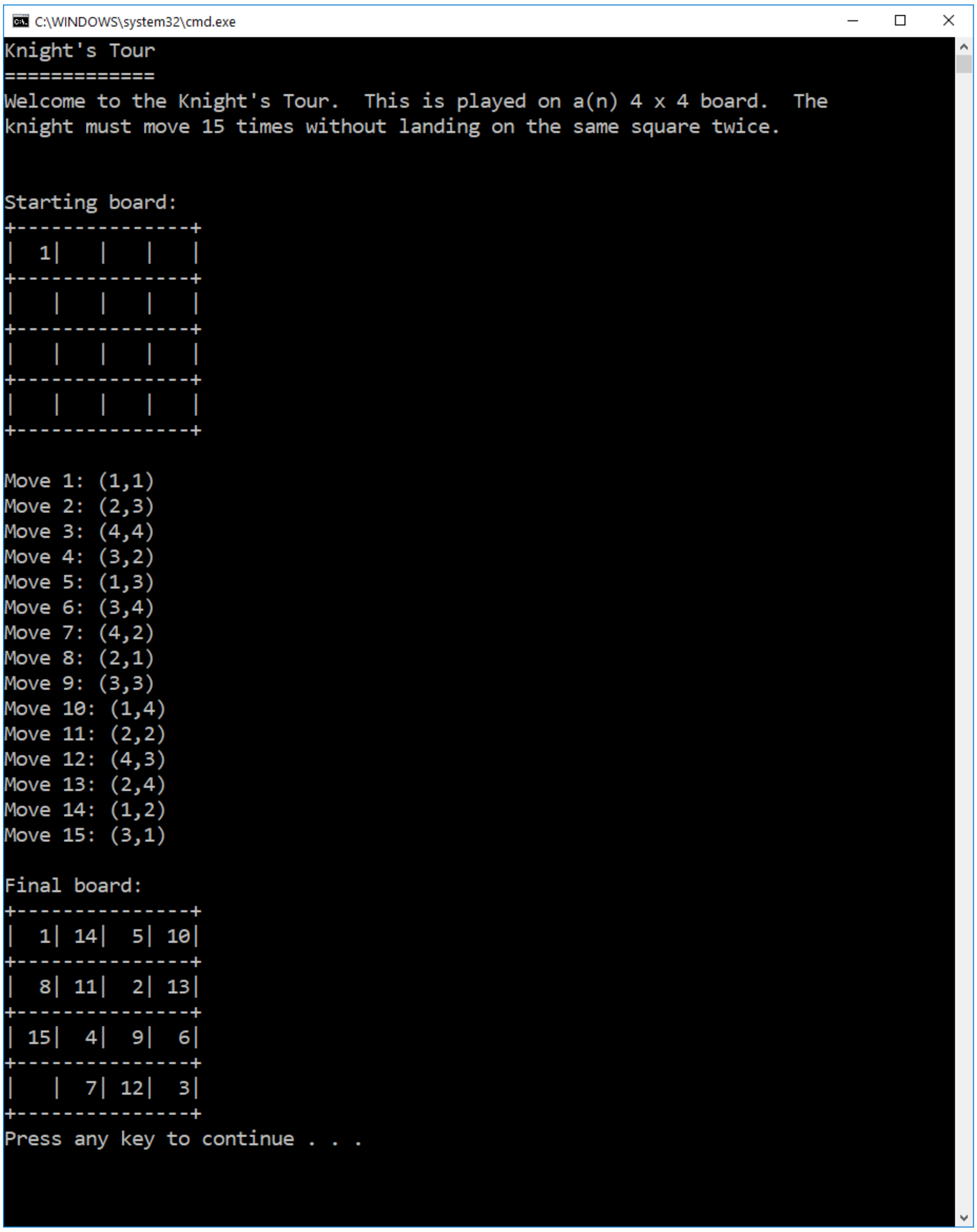
The computer will select moves by creating a *game tree*. A game tree consists of all the possible states of the game and in this assignment the computer determines any possible solution to be the best one. Each node of the game tree has children that indicate the states of the game that follow from the state of the parent for each possible move, i.e. each node in the tree possesses at most 8 children of any node but as more moves are made, or if a smaller board is used, then there will become less.

This assignment uses many data structures (trees, linked-lists, stacks, and queues) to solve the puzzle. You should create and traverse the game tree using a stack and queue as intermediate data structures. When a solution is found the solution should be displayed. The program will be text only.

A *Visual Studio* project file is available for download from MyLO. The project contains many header (`.h`) and source (`.c`) files. All required files are present. *You should not add any further files or change any code which is provided (other than the settings in* `assig_three219.h`*).*

You need to complete the functions within the program files which have been declared but for which the function bodies are missing and to update the header comments of the source files that you complete with your names, student IDs and ratio of effort.

Please note: there is not enough memory granted by the operating system to the running program to solve the depth-first problem on an 8x8 board or larger, or the breadth-first problem on a 6x6 board or larger. A sample run of the game is shown below on a 4x4 board. Please also note that there is no solution available for a tour of length 16 on a 4x4 board.



```
Knight's Tour
=============
Welcome to the Knight's Tour.  This is played on a(n) 4 x 4 board.  The
knight must move 15 times without landing on the same square twice.


Starting board:
+---------------+
|  1|   |   |   |
+---------------+
|   |   |   |   |
+---------------+
|   |   |   |   |
+---------------+
|   |   |   |   |
+---------------+


Move 1: (1,1)
Move 2: (2,3)
Move 3: (4,4)
Move 4: (3,2)
Move 5: (1,3)
Move 6: (3,4)
Move 7: (4,2)
Move 8: (2,1)
Move 9: (3,3)
Move 10: (1,4)
Move 11: (2,2)
Move 12: (4,3)
Move 13: (2,4)
Move 14: (1,2)
Move 15: (3,1)

Final board:
+---------------+
|  1| 14|  5| 10|
+---------------+
|  8| 11|  2| 13|
+---------------+
| 15|  4|  9|  6|
+---------------+
|   |  7| 12|  3|
+---------------+
Press any key to continue . . .
```

Marking scheme

Task/Topic	**Maximum mark**
Program operates as specified	
`square_state.c` correctly completed	2
`game_state.c` correctly completed	10
`t_node.c` correctly completed	5
`game_tree.c` correctly completed	21
`stack.c` correctly completed	5
`queue.c` correctly completed	5
Program Style	
Does not unnecessarily repeat tests or have other redundant/confusing code	6
Uses correctly the C naming conventions	6
Alignment of code and use of white space makes code readable	6
Always uses blocks in branch and loop constructs	6
Meaningful identifiers	6
Variables declared at the top of functions	6
Each variable declaration is commented	6
Comments within the code indicate the purpose of sections of code (but DO NOT just duplicate what the code says)	6

Program Style

Your program should follow the following coding conventions:

- `const` variable identifiers should be used as much as possible, should be written all in upper case and should be declared before all other variables;
- Variable identifiers should start with a lower case letter, be meaningful, and variables should only be declared at the top of a function;
- Every `if` and `if-else` statement should have a block of code (i.e. collections of lines surrounded by `{` and `}`) for both the `if` part and the `else` part (if used);
- Every loop (`do`, `while`, `for`) should have a block of code;
- The keyword `continue` should not be used;
- The keyword `break` should only be used as part of a `switch` statement;
- Opening and closing braces of a block should be aligned;
- All code within a block should be aligned and indented 1 tab stop (or 4 spaces) from the braces marking this block;
- Commenting:
    - There should be a block of header comment which includes at least
        - file name
        - student names
        - student identity numbers
        - an indication of the ratio of effort, if this is missing 50:50 will be assumed
        - a statement of the purpose of the program
        - date
    - Each variable declaration should be commented
    - There should be a comment identifying groups of statements that do various parts of the task

    - Comments should describe the strategy of the code and should not simply translate the C into English

Style marks will be awarded proportionally, i.e. if you attempt only half the coding you can expect only half the style marks.

What and how to submit

Only one submission is required for the pair but both paper and electronic submission is required for each pair.

What to submit

Paper submission

- A signed cover page (blanks can be collected from the ICT office or from the ICT web site). *Your assignment will not be marked unless this is present*;
- A print-out of the six '`.c`' source code files for the program; and
- A statement from each person indicating what proportion of the work was theirs. If the work was done equally, then each person should claim 50% of the effort. It is expected that this will be the case and this will be assumed if no statement is submitted. If, however, one person has done very little, then this should be stated. A reason why this is the case should be given. For example, if one student does 80% of the work because their partner did not turn up to meetings, complete work by the agreed time, etc. then this should be stated. If one student does 30% of the work because the other student simply went off and did it all by themselves and without consultation then this should be stated.

Electronic submission

- You should submit the entire Visual Studio project folder compressed as a ZIP file. *Do not submit a RAR archive.*

How to submit

Paper submission

- Print the programs which you have modified *in landscape orientation*. Do not provide screen shots of the code or print with a coloured background.
- Firmly staple together all of the required documents (with the signed cover page on top) and place them in the appropriate submissions box near the ICT Help Desk. *Your assignment will not be marked unless this is present.*

Electronic submission

- The entire Visual Studio project folder should be converted into a ZIP file and the ZIP file should be submitted via MyLO by one member of the pair. Only one submission is required.
    - Log in to MyLO and navigate to the `Assignments` tool under the `Assessments` icon in the top tool bar.
    - Select `Assignment 3` from the list of available drop-boxes.
    - Click on `Add a File` and follow the instructions to attach your ZIPped project folder and then click `Add`. Then click `Submit`.
- If you want to resubmit, name your revised ZIPped project folder 'version 2' and repeat the paper and electronic submission process.

- The program is required to work under the Visual Studio environment in the labs. Any development environment may be used; once the programming is complete replace the altered programs in the downloaded Visual Studio project and submit it.

Remember that only one paper and electronic submission is required per pair.

Plagiarism and Cheating:

Practical assignments are used by the Discipline of Information and Communication Technology for students to both reinforce and demonstrate their understanding of material which has been presented in class. They have a role both for assessment and for learning. It is a requirement that work you hand in for assessment is substantially your own.

Working with others

One effective way to grasp principles and concepts is to discuss the issues with your peers and/or friends. You are encouraged to do this. We also encourage you to discuss aspects of practical assignments with others. However, once you have clarified the principles, your pair must express them in writing or electronically entirely by yourselves. In other words you must develop the algorithm to solve the problem and write the program which implements this algorithm with your partner and no one else (other than staff).

Cheating

- Cheating occurs if you claim work as your own when it is substantially the work of someone else.
- Cheating is an offence under the Ordinance of Student Discipline within the University. Furthermore, the ICT profession has ethical standards in which cheating has no place.
- Cheating involves two or more parties.
  - If you allow written work, computer listings, or electronic version of your code to be viewed, borrowed, or copied by another student you are an equal partner in the act of cheating.
  - You should be careful to ensure that your work is not left in a situation where it may be accessed/stolen by others.
- Where there is a reasonable cause to believe that a case of cheating has occurred, this will be brought to the attention of the unit lecturer. If the lecturer considers that there is evidence of cheating, then no marks will be given to any of the students involved. The case will be referred to the Head of Discipline for consideration of further action.

Julian Dermoudy, September 20th 2019.