

CSCI1120 Grading Scheme

Assignment 5 Twin Knights Tour

2. Test cases

2.1 Constructor

Test the public constructor function for the class.

Case	Constructor	Outputs	Description
1	<code>TwinKnightsTour game(0,0,3,3);</code>	None	Check: <code>board[0][0] = "A";</code> <code>board[3][3] = "a";</code> for other <code>i, j</code> : <code>board[i][j]='.'</code> <code>posR1 = 0;</code> <code>posC1 = 0;</code> <code>posR2 = 3;</code> <code>posC2 = 3;</code> <code>steps1 = 0;</code> <code>steps2 = 0;</code> <code>consec1 = 0;</code> <code>consec2 = 0;</code>
2	<code>TwinKnightsTour game(5,0,1,3);</code>	None	Check: <code>board[5][0] = "A";</code> <code>board[1][3] = "a";</code> for other <code>i, j</code> : <code>board[i][j]='.'</code> <code>posR1 = 5;</code> <code>posC1 = 0;</code> <code>posR2 = 1;</code> <code>posC2 = 3;</code> <code>steps1 = 0;</code> <code>steps2 = 0;</code> <code>consec1 = 0;</code> <code>consec2 = 0;</code>

2.2 Function Unit Test for print

Test the `print()` member function.

Case	Board content	Std out	Description
------	---------------	---------	-------------

1	<pre> 0 1 2 3 4 5 0 A d . . . D 1 . . B . . . 2 c . . . C . 3 4 . b 5 . . . a . . </pre>	<pre> Output: 0 1 2 3 4 5 0 A # . . . @ 1 . . B . . . 2 c . . . C . 3 4 . b 5 . . . a . . </pre>	Note the spacing!
2	<pre> 0 1 2 3 4 5 0 A d K . G D 1 L . B E J . 2 c . e H C F 3 . i . . f I 4 k b . h . . 5 . . j a . g </pre>	<pre> 0 1 2 3 4 5 0 A d K . G D 1 @ . B E J . 2 c . e H C F 3 . i . . f I 4 # b . h . . 5 . . j a . g </pre>	Note the spacing !
3	<pre> 0 1 2 3 4 5 6 7 8 9 10 11 0 A s f e . . . 1 . . B . . t D . . r o d . 2 . . . g D . . F c . . q . 3 . I . . C h b . . j m . . 4 G . . l a . . . 5 J . H . . i . . k . . . 6 R 7 . K . S . O 8 . . . L . P 9 T . . . Q 10 N 11 . U . M </pre>	<pre> 0 1 2 3 4 5 6 7 8 9 10 11 0 A # s f e . . . 1 . . B . . g D . . r o d . 2 . . I . . C h b . . j m . . 3 G . . l a . . . 4 J . H . . i . . k . . . 5 J i l a . . . 6 R 7 . K . S . O 8 . . . L . P 9 T . . . Q 10 N 11 . @ . M </pre>	A board with size 12*12.
4	<pre> 0 1 2 3 4 5 6 7 8 9 10 11 0 A f e . . . 1 . . B E . . . 2 . . y . g D . . d . AF 3 . I . . C x b w . . AC . 4 . z . h G u j a v . Z . AB 5 J . H . . i l a v . Z . AB 6 . . aa . i l a v . Z . AB 7 . K . . . t k . W . Y 8 . . ab o . m . s . AA V 9 . . . L n . . q T . X 10 . . . P # N o R . r U 11 . . . M . Q . . p S . </pre>	<pre> 0 1 2 3 4 5 6 7 8 9 10 11 0 A f e . . . 1 . . B E . . . 2 . . y . g D . . d . @ 3 . I . . C x b w . . AC . 4 . z . h G u j a v . Z . AB 5 J . H . . i l a v . Z . AB 6 . . aa . i l a v . Z . AB 7 . K . . . t k . W . Y 8 . . ab o . m . s . AA V 9 . . . L n . . q T . X 10 . . . P # N o R . r U 11 . . . M . Q . . p S . </pre>	Note the spacing and 2-width letters!

2.3 Function Unit Test for isValid()

Test the isValid() member function.

Cas e	Inputs	Outputs	Description
1	<pre> Board: 0 1 2 3 4 5 0 A # . . . @ 1 . . B . . . 2 c . . . C . 3 4 . b 5 . . . a . . Knight=@ r=10 c=-1 posR1=0 posC1=5 posR2=0 posC2=1 consec1=0 </pre>	False	Out of board

	consec2=1		
2	Board: <pre> 0 1 2 3 4 5 0 A # . . . @ 1 . . B . . . 2 c . . . C . 3 4 . b 5 . . . a . . </pre> Knight='*' r=2 c=2 posR1=0 posC1=5 posR2=0 posC2=1 consec1=0 consec2=1	False	The wrong knight character
3	Board: <pre> 0 1 2 3 4 5 0 A # . . . @ 1 . . B . . . 2 c . . . C . 3 4 . b 5 . . . a . . </pre> Knight='#' r=0 c=3 posR1=0 posC1=5 posR2=0 posC2=1 consec1=0 consec2=1	False	The move is not L-shaped.
4	Board:	False	The move is the third consecutive move for "@" (1 st move: @ 1 3 2 nd move: @ 2 5)

	<pre> 0 1 2 3 4 5 0 A # . . . D 1 . . B E . . 2 c . . . C @ 3 4 . b 5 . . . a . . Knight='@' r=4 c=4 posR1=2 posC1=5 posR2=0 posC2=1 consec1=2 consec2=0 </pre>		
5	<pre> Board: 0 1 2 3 4 5 6 7 8 9 10 11 0 A e . . . 1 . . B . . . f E . . # . 2 g D . . o d . . 3 . I . C . . F c 4 . . . h G b . . . n . . 5 J . H . . . j m 6 . . . i l a 7 . K . . R . . k 8 . . S . O 9 @ . L . Q 10 . . . P . N 11 . . . M Knight='#' r=2 c=8 posR1=9 posC1=0 posR2=1 posC2=10 consec1=2 consec2=0 </pre>	False	Try to move to a visited position
6	Board:	True	A valid case.

	<pre> 0 1 2 3 4 5 6 7 8 9 10 11 0 A f E . . . 1 . . B p . 2 g D . r o d . 3 . I . C . . F c . q . 4 . . . h G b . . # . . 5 J . H . . . j m . . . 6 i l a 7 . K . . R . . k . . . 8 . . S . O 9 T . L . . Q 10 . . . P . N 11 . @ . M Knight='#' r=5 c=10 posR1=11 posC1=1 posR2=4 posC2=8 consec1=0 consec2=1 </pre>		
--	--	--	--

2.4 Function Unit Test for hasMoreMoves()

Test the hasMoreMoves() member function.

Case	Board	Outputs	Description
1	<pre> Board: 0 1 2 3 4 5 0 A d K . G D 1 @ . B E J . 2 c . e H C F 3 . i . . f I 4 # b . h . . 5 . . j a . g consec1=0 consec2=2 posR1=1 posC1=0 posR2=4 posC2=0 </pre>	False	@ has no valid moves. # has made two consecutive moves.
2	<pre> Board: </pre>	True	@ has no valid moves. # still has two potential moves.

	<pre> 0 1 2 3 4 5 0 A d . . G D 1 . . B E J . 2 c # e H C F 3 . i . K f I 4 k b . h . @ 5 . . j a . g consec1=0 consec2=1 posR1=4 posC1=5 posR2=2 posC2=2 </pre>		
3	<p>Board:</p> <pre> 0 1 2 3 4 5 0 A d . . G D 1 # . B E J . 2 c L e H C F 3 . i . K f I 4 @ b . h . . 5 . . . a . g consec1=2 consec2=0 posR1=4 posC1=0 posR2=1 posC2=0 </pre>	True	@ has made two consecutive moves. # has only one valid move.
4	<p>Board:</p> <pre> 0 1 2 3 4 5 6 7 8 9 10 11 0 A e . . . 1 . . B . . . f E . . p . 2 . . . g D . r o d . . 3 . I . C . . F c . q . . 4 . . . h G b . . # n . . 5 J . H . . j m 6 i l a 7 . K . . R . . k 8 . . S . O 9 T . L . . Q 10 . . . P . N 11 . @ . M </pre>	True	@ has no valid moves. # still has several potential and valid moves.

	consec1=0 consec2=1 posR1=11 posC1=1 posR2=4 posC2=8		
--	---	--	--

2.5 Function Unit Test for move()

Test the move() member function.

Case	Inputs	Outputs	Description
1	Board: 0 1 2 3 4 5 0 A # . . . @ 1 . . B . . . 2 c . . . C . 3 4 . b 5 . . . a . . Knight='@' r=1 c=3 consec1=0 consec2=1 posR1=0 posC1=5 posR2=0 posC2=1 steps1=3 steps2=3	True	Need to check the data members. Consec1=1 Consec2=0 posR1=1 posC1=3 Board[1][3]=='E'
2	Board:	False	Not a valid move. No update on data members!

	<pre> 0 1 2 3 4 5 0 A # . . . D 1 . . B @ . . 2 c . . . C . 3 4 . b 5 . . . a . . Knight='@' r=1 c=4 consec1=1 consec2=0 posR1=1 posC1=3 posR2=0 posC2=1 steps1=4 steps2=3 </pre>		
3	<pre> Board: 0 1 2 3 4 5 6 7 8 9 10 11 0 A e . . . 1 . . B . . . f E . . . 2 . . . g D . . . d . . 3 . I . C . F c . . . 4 . . . h G b # . . . 5 J . H . . u j . . . 6 . . . i l a v . @ . 7 . K . . . t k . W . Y 8 . . . O . m . s . V 9 . . L . n . q T . X 10 . . . P . N o R . r U 11 . . . M . Q . . p S . Knight='@' r=8 c=10 consec1=0 consec2=2 posR1=6 posC1=9 posR2=4 posC2=6 steps1=26 </pre>	True	<pre> .Need to check the data members. Consec1=1 Consec2=0 posR1=8 posC1=10 Board[8][10]='AA' </pre>

	steps2=23		
4	<p>Board:</p> <pre> 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 0 Y . W . . AB 1 . . N . V AG Z 2 M . . . O X U AH AA . AC 3 . . L Q T AF 4 . . S . . P AI . . AD 5 . K . . R A . AE AJ . . AP . . . 6 . af . F I . a B c . AL . . AQ . 7 . G J ag . . D . AK . e . AO . 8 ae bf bc H E ab a1 b C d AM . . . AR 9 bb . ad be ah . . aa . . f . AN . 10 . bd bg x ac aj i . AS . 11 . ba . . al w z . . j g . . . 12 . # am t y . ak v BA n . . h . AT 13 an s az aq av u p m at AW AZ k . AU . 14 . . ao ax q ar au BB o l . AV AY . . 15 . ay r . ap aw . as . @ AX . . . </pre> <p>Knight='#' r=10 c=0 posR1: 15 posC1: 9 posR2: 12 posR2: 1 steps1: 54 steps2: 59 consec1: 0 consec2: 1</p>	True	<p>Need to check the data members.</p> <p>posR1: 15 posC1: 9 posR2: 10 posR2: 0 consec1: 0 consec2: 2 Board[10][0]='bi'</p>

2.6 Gameflow Test

Here we use 2 test flows to test the implementation of gameflow.

2.6.1 Flow1 board size 6

- This will use our class implement and student's client code

	Inputs	Outputs	Description
1	3 3 3 3	Invalid position(s)!	Two knights with same position, when the board size is 6
2	1 2 5 6	Invalid position(s)!	Out of the board
3	0 1 4 5	<pre> 0 1 2 3 4 5 0 . @ 1 2 3 4 # 5 </pre>	initial the board and print the board
5	@ 2 2 # 2 4 # 4 3 # 3 5 @ 3 0 ... 21 steps	<pre> Move (knight row col): @ 2 2 0 1 2 3 4 5 0 . A 1 2 . . @ . . . 3 4 # 5 Move (knight row col): # 2 4 0 1 2 3 4 5 0 . A 1 2 . . @ . # . 3 4 a 5 Move (knight row col): # 4 3 0 1 2 3 4 5 0 . A 1 2 . . @ . b . 3 4 . . . # . a 5 Move (knight row col): # 3 5 Invalid move! </pre>	Can recurrently accept input and invaild input, and output “invalid move” for invaild input and ask users to enter next input.
6	# 3 3	<pre> Move (knight row col): # 3 3 0 1 2 3 4 5 0 . A J E f . 1 @ D g H . F 2 h I B e b . 3 C d i # G . 4 j . . c . a 5 . . k . . . No more moves! </pre>	When there is no more moves.

2.6.2 Flow2 board size 16

- This will use student's class implement and student's client code

	Inputs	Outputs	Description
1	16 2 5 6	Invalid position(s)!	Two knights with same position, when the board size is 6
2	3 3 3 3	Invalid position(s)!	Out of the board
3	5 5 6 6	<pre> 0 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 1 2 3 4 5 @ 6 7 8 # 9 10 11 12 13 14 15 </pre>	initial the board and print the board
4	@ 1 -1 @ 4 5 # 8 7 # 6 8 @ 6 7 ... All: 119 steps	<pre> Move (knight row col): @ 1 -1 Invalid move! Move (knight row col): @ 4 5 Invalid move! Move (knight row col): # 8 7 0 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 1 2 3 4 5 @ 6 a 7 # 8 9 10 11 12 13 14 15 Move (knight row col): # 6 8 0 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 1 2 3 4 5 @ 6 a . # 7 b 8 9 10 11 12 13 14 15 </pre>	Can recurrently accept input and invaild input, and output “invalid move” for invaild input and ask users to enter next input.
5	... # 10 0	<pre> Move (knight row col): 0 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 1 Y . . W . . AB 2 N . . V AG Z 3 M O X U AH AA . . AC . . . 4 L Q T . . AF 5 S . . P AI . . AD 6 . . K . . R A . . AE AJ AP . 7 . af . . F I . . a B c . . AL . . AQ . 8 . G J ag . . D . . AK . . e . . AO . 9 . se bf bc H E ab ai b C d AH . . . AR 10 ad be ah . . aa f . AW . 11 . . bd bg x ac aj i . AS . 12 . . ba . . al w z j g . . . 13 . . bh am t y . . ak v BA n . . . h . AT 14 . . s az aq av u p m at AW AZ k . AU . 15 . . ao ax q ar au BB o l . . AV AY . . 16 . . ay r . . ap aw . . as . . @ AX . . . No more moves! </pre> <p>No more moves!</p>	When there is no more moves.