

CS575 Design and Analysis of Algorithms

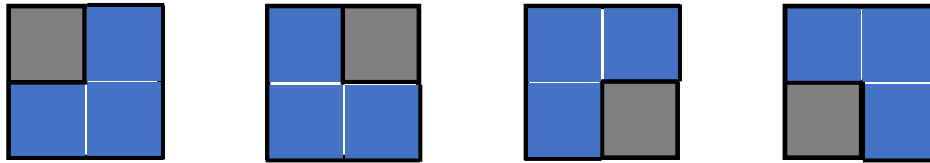
Spring 2020

Programming Assignment 2

Assigned: February 19, 2020
Due: Midnight Monday, March 9, 2020

Assignment

Given a natural number n and a chess board with 2^n squares on each side (the length of the side is 2^n) with a missing square in any location, you will use the divide and conquer approach to design and implement an algorithm (see the algorithm skeleton from the class slides) to cover the chess board (with a missing square) by an arrangement of trominoes (an L-shaped arrangement of three squares; see figure below) such that all trominoes are confined to the boundaries of the chess board, and no tromino overlaps another.



We will use x coordinate and y coordinate to record the location of the missing square. It is convenient to treat the origin $(0, 0)$ as the lower left corner of the board. For instance, the coordinate of the missing square in the first figure above is $(0, 1)$; the coordinate of the missing square in the second figure above is $(1, 1)$.

You will make a function called `tromino` as below.

```
void tromino /* function to do tiling */
( int x_board, /* x coordinate of board */
  int y_board, /* y coordinate of board */
  int x_missing, /* x coordinate of missing square */
  int y_missing, /* y coordinate of missing square */
  int board_size ); /* size of board, which is a power of 2 */
```

The main program should call `tromino(0, 0, x_missing, y_missing, board_size)`, which allows the user to input `board_size` by prompting the line “Please enter size of board (0 to quit):” first, then to input coordinates of missing square for `x_missing` and `y_missing` by prompting the line “Please enter coordinates of missing square (separate by a space):”.

For the `tromino` function below,

```
void tromino /* function to do tiling */
( int x_board, /* x coordinate of board */
  int y_board, /* y coordinate of board */
  int x_missing, /* x coordinate of missing square */
  int y_missing, /* y coordinate of missing square */
```

```
int board_size ) /* size of board, which is a power of 2*/
```

you will need to set up the base case for `board_size = 2`. What you need to do in the base case is to decide which L shape to be put in the three squares. Please print “LR” (Lower Right) in all three squares for the first L shape (see the figure above), print “LL” (Lower Left) for the second L shape (see the figure above), “UL” (Upper Left) for the third L shape (see the figure above), “UR” (Upper Right) for the forth L shape (see the figure above).

You will do the following four recursive calls for the four `half_size` (`board_size/2`) subboards: upper left subboard, upper right subboard, lower left subboard, and lower right subboard.

```
/* tile the four subboards */
```

```
tromino( x_board, y_board + half_size, x_upper_left, y_upper_left, half_size );  
tromino( x_board + half_size, y_board + half_size, x_upper_right, y_upper_right, half_size );  
tromino( x_board, y_board, x_lower_left, y_lower_left, half_size );  
tromino( x_board + half_size, y_board, x_lower_right, y_lower_right, half_size );
```

The main program should output the arrangement of trominoes at each coordinate location. For example, when `board_size=2`, `x_missing = 0`, and `y_missing = 1` (the first figure above), the output should be as follows (please use “MS” to stand for the missing square).

```
MS   LR  
LR   LR
```