Assignment 1;

GOAL

----------A template has been given for this assignment----------
We will be creating a program that simulates a simple memory game. In this memory game, a grid of face down cards is created. Every element of the grid has a matching pair. The user's goal is to find the location of all pairs. For example,

```
1  4  3
2  3  1
4  2  5
6  6  5
```

For every iteration of a loop, the grid is displayed, and the user is prompted to enter two coordinates; x and y. where x is for row and y is for column.

For example, the coordinates could be (row, column) and start at 0.

(0, 0)	(0, 1)	(0, 2)
(1, 0)	(1, 1)	(1, 2)
(2, 0)	(2, 1)	(2, 2)
(3, 0)	(3, 1)	(3, 2)

Once the coordinates are entered, the two cards at the corresponding coordinates will be compared to each other

1) If the cards match each other, the user has found a pair, so we keep them flipped over
2) If the cards do not match, we temporarily show the user the values of the cards at the coordinates, and then we flip the cards over again

This loop should continue until the user finds all the pairs, or until they choose to quit.

OUR APPROACH

One "difficult" part of this lab is figuring out how to keep track of what cards have been found and what cards haven't.
Remember that matched pairs need to be displayed, while all other pairs should be hidden. For example,

```
*  *  *                                  *  *  1
*  *  *      --> User matched a pair -->  *  *  *
*  *  *                                  1  *  *
*  *  *                                  *  *  *
```

So how to we keep track of what numbers to display and what numbers to put stars over? The approach used in the program below is to maintain two separate 2D arrays: one will hold the actual numerical values and the other will hold booleans that represent whether or not the value has been matched. Think back to parallel 1D arrays. This is a similar approach.

THE FLOW OF OUR PROGRAM

---

1) Welcome the user and ask him to enter his full name.

2) Create 2D array of values. First put everything in order and shuffle the location of the elements. This step is given.

3) Create the 2D array of Booleans that represents whether or not the card is matched. This should be initialized to all FALSE at first. Why? Because no cards are matched initially!

4) Start a while loop that does the following

a. Display the current state of the game board. Remember that non-matched pairs should be "facedown" and that matched pairs should be "faceup"

b. Prompt the user to enter the coordinates of two cards

c. If the values of the two coordinates match, then say that a match has been found and flip the cards over forever. If the values do not match, temporarily flip over the cards to show the user the values, then flip them over again.

5) Save the user's name and how many turns it took to win the game into an output text file. If he quits before winning save "Don't give up" message in the file.

A template for the assignment is provided. You must not delete anything there.
You must have at least three functions:

```
void InitializeCards(int cards[][LENGTH]);
void ShowCards(int cards[][LENGTH], bool faceup[][LENGTH]);
int main ( )
```

you may add more functions if you like.
Submit the .cpp file only.

*********/

Snippets of a sample run:

Welcome to my game!

Please enter your full name: abeer alameer

Find all the matching pairs on the board!

```
  0 1 2 3
 =========
```

```
0 | * * * * |
1 | * * * * |
2 | * * * * |
3 | * * * * |
  =========
```

Enter x and y position of the first card: 0 1

Enter x and y position of the second card: 2 2

```
   0 1 2 3
  =========
0 | * 4 * * |
1 | * * * * |
2 | * * 3 * |
3 | * * * * |
  =========
```

No match.

Flipping cards over again.

Enter 'q' to quit or press any key to continue...