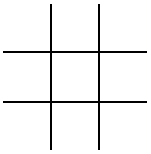
CSC 2110, Spring 2013
Programming Assignment 1
Assign date: February 6, 2013 -- Due: February 18, 2013 at class

Objectives:

- To gain more experience with user defined classes.
- To gain experience with collaborating objects and operator overloading.

Overview:

For this assignment you have to write a program that will allow two players to play the well-known **Tic-Tac-Toe** game.



The game will start with an empty board as shown above. A move by Player1 will be represented by an "X" and a move by Player2 will be represented by an "O". The player who gets three marks in a row (horizontal, vertical, or diagonal) will win the game. If all the nine positions are exhausted without a result, then it will be a tie. You have to use a 2-dimensional 3x3 array of characters to store a particular state of the game. You can use the NULL character ('\0' or ASCII value 0) to represent a blank position.

For this assignment (as with all assignments for this class), you must use the *object oriented design* paradigm to implement the **TicTacToe** game. You will need to have two classes: a **TicTacToe** class and a **Player** class. The **TicTacToe** class should contain the above mentioned 2D array for representing the game board, i.e., this class should have a 2D array of characters as its data member. You should have a zero-argument *constructor* in the **TicTacToe** class that will generate the initial game state (all the 2D array elements initialized to '\0'). You should overload the << operator to display the game board at any point. The game board should be displayed along with the grid as shown in the sample executions. There should also be a member function called **SetValue** that will have three arguments (a row index, a column index, and an integer representing the player index) and a *boolean* return type. This function will set the specified cell of the 2D array to an appropriate character depending on the player index. This function should also make sure that a player makes its move to a blank and valid (inside the array bounds) position. In case a player wants to make a move to an invalid or already occupied position, it should return false. In case of a valid move, it should return true. There should be another member function called **GetStatus** that will return an integer (0-3) representing the game status; 0 representing a tie, 1 representing a win by Player1, 2 representing a win by Player2, and 3 otherwise.

The **Player** class will be relatively simple and will have information such as the player's name and an integer representing whether he/she is Player1 or Player2. It should have a two-argument constructor to initialize the data members, a member function called **GetIndex** to return an integer value (1 or 2) depending on whether it's Player1 or Player2, and a member function called **GetName** to return the name of the player. There should also be a member function called **NextMove** that will allow a **Player** object to make a move. This function should accept a **TicTacToe** object by reference, prompt the player to enter their move, accept the move in the form of the row and column numbers of the position where they want to make the move (look at the sample execution), and make changes to the 2D array of the **TicTacToe** object. Note that this function will be calling the **SetValue** function of the **TicTacToe** class and will keep on re-prompting the player (as shown in the sample executions) to make another move as long as the **SetValue** function returns false. Note that the **Player** class should not have the **TicTacToe** object as its data member.

Use different header and source file for each class. Write a driver file containing the main. Inside your main, you have to prompt the players to enter their names, do a random toss to select one of them as the first player (you have to make use of random number generation), create a **TicTacToe** and two **Player** objects, and then continue the game (making use of the appropriate member functions) until a result is reached. Your output should follow the sample executions.

Deliverables:

- You need to zip all your source code (all header and cpp files) and submit it to the appropriate drop box on iLearn**.** The name of your zip file should be lastname_prog1_2110.zip (for example my filename would be mcdaniel_prog1_2110.zip.

Grading:

This lab is worth 100 points, distributed as follows:

Functionality (program works correctly, and meets the specifications)	85 pts
Program Style	15 pts
Meaningful variable names	
Proper indentation	
Correct program header	
Comments throughout the program	

**** NOTE: If your program doesn't compile your score for the assignment will be zero. Your submission will not be graded if you don't submit your code to iLearn by the due date (February 18, 2013).**

Sample Execution I:

```
Welcome to Tic-Tac-Toe
----------------------

First player, enter your name: John
Second player, enter your name: David

David won the toss.

   |   |
-----------
   |   |
-----------
   |   |

David, make your move: 1 1

 X |   |
-----------
   |   |
-----------
   |   |

John, make your move: 2 1

 X |   |
-----------
 O |   |
-----------
   |   |

David, make your move: 3 3

 X |   |
-----------
 O |   |
-----------
   |   | X

John, make your move: 3 2

 X |   |
-----------
 O |   |
-----------
   | O | X
```

```
David, make your move: 1 1
Illegal move, make another one: 2 1
Illegal move, make another one: 3 4
Illegal move, make another one: 2 2

 X |   |
-----------
 O | X |
-----------
   | O | X

David has won!
```

Sample Execution II:

```
Welcome to Tic-Tac-Toe
----------------------

First player, enter your name: Susie
Second player, enter your name: Jake

Susie won the toss.

   |   |
-----------
   |   |
-----------
   |   |

Susie, make your move: 3 3

   |   |
-----------
   |   |
-----------
   |   | X

Jake, make your move: 2 2

   |   |
-----------
   | O |
-----------
   |   | X

Susie, make your move: 1 1
```

```
X  |   |
-----------
   | O |
-----------
   |   | X

Jake, make your move: 2 3

 X |   |
-----------
   | O | O
-----------
   |   | X

Susie, make your move: 1 3

 X |   | X
-----------
   | O | O
-----------
   |   | X

Jake, make your move: 1 2

 X | O | X
-----------
   | O | O
-----------
   |   | X

Susie, make your move: 3 2

 X | O | X
-----------
   | O | O
-----------
   | X | X

Jake, make your move: 2 1

 X | O | X
-----------
 O | O | O
-----------
   | X | X

Jake has won!
```

Sample Execution III:

```
Welcome to Tic-Tac-Toe
----------------------

First player, enter your name: Robbie
Second player, enter your name: Bill

Bill won the toss.

   |   |
-----------
   |   |
-----------
   |   |

Bill, make your move: 4 1
Illegal move, make another one: 3 2

   |   |
-----------
   |   |
-----------
   | X |

Robbie, make your move: 1 2

   | O |
-----------
   |   |
-----------
   | X |

Bill, make your move: 1 3

   | O | X
-----------
   |   |
-----------
   | X |

Robbie, make your move: 3 3

   | O | X
-----------
   |   |
-----------
```

```
   | X | O

Bill, make your move: 2 3

   | O | X
-----------
   |   | X
-----------
   | X | O

Robbie, make your move: 2 2

   | O | X
-----------
   | O | X
-----------
   | X | O

Bill, make your move: 1 1

 X | O | X
-----------
   | O | X
-----------
   | X | O

Robbie, make your move: 3 1

 X | O | X
-----------
   | O | X
-----------
 O | X | O

Bill, make your move: 2 2
Illegal move, make another one: 2 1

 X | O | X
-----------
 X | O | X
-----------
 O | X | O

Game tied!
```