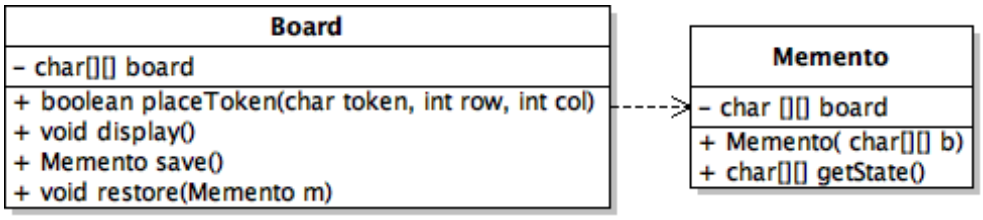
CECS 277 – Lab 14

Tic-Tac-Toe – Memento Pattern

Create a program that plays the game Tic-Tac-Toe. Use the Memento pattern to implement an undo feature that allows the user to backtrack to the previous turn.



Board class –

1. Make a 3x3 char array to store the board. Initialize it with all spaces.
2. placeToken method passes in the token (an 'x' or an 'o'), the row, and column in the array the token will be placed. Check the row and column to make sure the values are not out of bounds, and also check that the location is empty before placing the token there. Return true if the token was successfully placed.
3. display method displays the board in the typical tic-tac-toe fashion. Add index values for the row and column to make it easier for the user to enter.
4. save method returns the Memento storing the current state of the board.
5. restore method resets the board to a previous state using the memento.

Memento class –

1. Has a 3x3 char array to store the state of board.
2. Constructor passes in a board to store. Make a copy of the array.
3. getState method returns the char array.

Main class –

1. Create a stack to store the mementos of the previous states of the board.
2. Have a menu to prompt the user to place a token, to revert, or to quit.
3. Placing a token should automatically alternate between placing an 'x' or an 'o'.
4. Placing a token should prompt the user for the row and column. If it is an invalid location (out of bounds, or a token already there), then prompt again.
5. Create a Memento and add it to the stack whenever it is 'x's turn, save the state before the token is placed so it can revert back to before the move.
6. Reverting should pop the previous Memento off the stack and restore the board to that point. Check to see if there are any Memento's to revert to.
7. End the program when the user quits.

Extra Credit –

1. +1 point – have the game randomly place the 'o' token. Choose a random row and column and put the 'o' token in that spot. If it is an invalid location, then try another spot (unless the board is full).
2. +1 point – create methods in the Board class to: check for a win (3 in a row in any direction), that returns the winning token, and to check for a cat's game (no winners).

Example Output:

```
  0 1 2
0  | |
  -----
1  | |
  -----
2  | |
1. Place Token
2. Revert
3. Quit
1
Place an x
Row: 1
Col: 1
  0 1 2
0  | |
  -----
1  |x|
  -----
2  | |
1. Place Token
2. Revert
3. Quit
1
Place an o
Row: 0
Col: 2
  0 1 2
0  | |o
  -----
1  |x|
  -----
2  | |
1. Place Token
2. Revert
3. Quit
1
Place an x
Row: 2
Col: 2
  0 1 2
0  | |o
  -----
1  |x|
  -----
2  | |x
1. Place Token
2. Revert
3. Quit
1
Place an o
Row: 0
Col: 1
  0 1 2
0  |o|o
  -----
1  |x|
  -----
2  | |x
1. Place Token
2. Revert
3. Quit
1
Place an x
Row: 1
Col: 0
  0 1 2
0  |o|o
  -----
1 x|x|
  -----
2  | |x
1. Place Token
2. Revert
3. Quit
2
  0 1 2
0  |o|o
  -----
1  |x|
  -----
2  | |x
1. Place Token
2. Revert
3. Quit
```