CS 485 25 points Assignment 6 MVP Memory Game **DUE: 6 pm April 19.**

You must work in your assigned group to complete this assignment.

You are to build a memory card game [1] using the MVP pattern. You will be given a working Model for the card game. You are free to change the Model (add an Observer, etc).

You must design a TextUI version of the game. The TextUI front end should display each card as a particular number.

The Model contains: Board.h, Card.h ,Player.h ,Model.h, as well as the associated .cpp file.
A very simple main.cpp exists to show you how to call the functions within the Model.

Rename the folder containing the VS solution to CS485_Memory_Group# and put that folder into Git (GitHub or GitLab). The starter UML that you are to update is in the root of the folder. Be sure to share the repository with me. Use the **x86** setting to build your code.

The projects in the Solution are:

12_ObserverExample:	Produces $(SolutionDir)\Debug\12_ObserverExample.lib
TextUI:	Produces $(SolutionDir)\Debug\TexUI.lib
MemoryGame_Model:	Produces $(SolutionDir)\Debug\MemoryGame_Model.lib
MemoryGame_TextUI:	Produces MemoryGame_TextUI.exe application

What do you need to turn in:

1. (22 pts) UML diagram of your design. A starter diagram containing the Model and TextUI is provided. The UML diagram only needs to contain:

> An abstract View interface (IMemory_View) and an abstract Presenter interface (IMemory_Presenter).
> Subclass each of these into Memory_View_TextUI and Memory_Presenter, respectively.
> Memory_View_TextUI must also inherit from TextUI.

The Presenter interface must define how the View will call down to the Presenter. The View interface defines how the Presenter updates the View and defines how the UI events are handled by the View.

2. (3 pts) Implement your game. You will need to produce the cpp files for the above classes as well as a very small driver. MemoryGame_Model: add IMemory_View.h, IMemory_Presenter.h, and Memory_Presenter.h/cpp. MemoryGame_TextUI: add Memory_View_TextUI.h/cpp

[1] https://en.wikipedia.org/wiki/Concentration_(game)

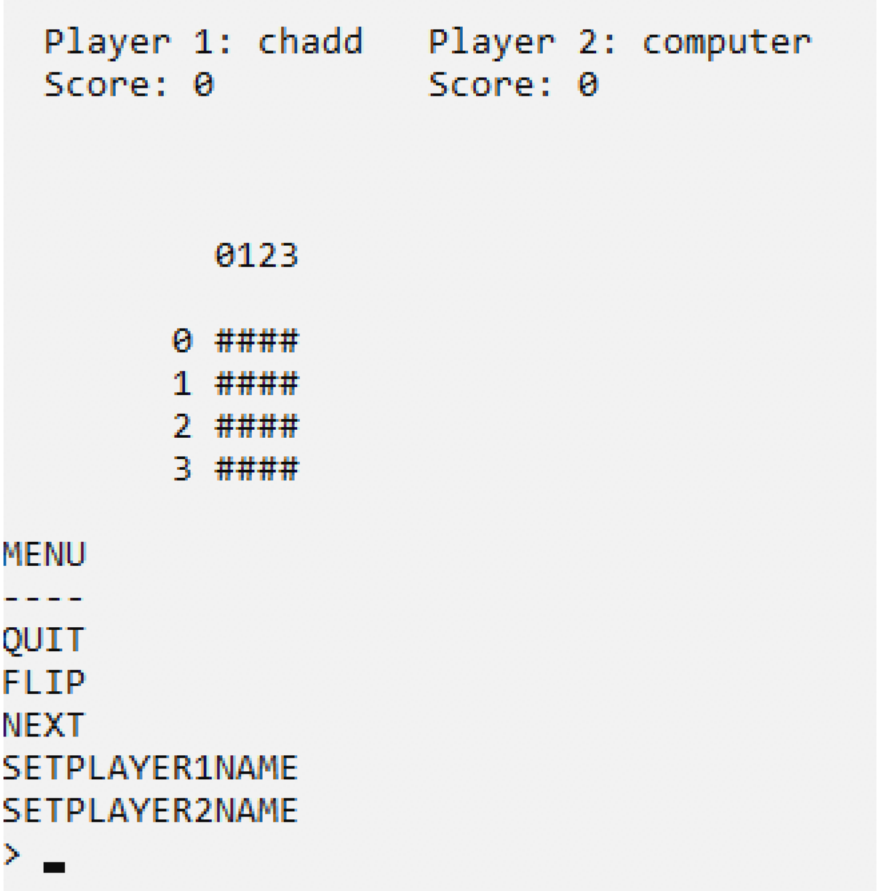
Example Output

An example output is shown below.

The menu options are as follows:
QUIT: quit the game
FLIP: flip one card over. This takes an x and y coordinate
NEXT: advance the turn to the next player. Should be used after Flip is called twice by a player.
SETPLAYER#NAME: set the name of the player. This takes a string as an argument.

These menu options each have direct analogues in the Model interface.

A (likely buggy) executable is posted on the class web page.



```
  Player 1: chadd   Player 2: computer
  Score: 0          Score: 0



          0123

        0 ####
        1 ####
        2 ####
        3 ####

MENU
----
QUIT
FLIP
NEXT
SETPLAYER1NAME
SETPLAYER2NAME
> 
```