CSSE 374: Lab 7-1

# Background

Assume that you work for a company that specializes in creating next generation command line games. So far, your company have developed two world famous games: number guessing game and word guessing game. Looking at the success of these two games, the upper management decided to develop a few more similar command lines games. You have been tasked with refactoring the codebase of these games to optimize it as much as possible for the future command line games using all the design patterns and principles you have learnt in the Software Design class.

# Design (C)

Review the source code and answer the following questions:

1. Does the code violate the “Separate what changes from what remains the same” principle? If so, how? If not, why not?
   1. *yes, the new games are closed in their own classes, but there is a bunch of repeated code*
2. Does the code violate the “Favor composition over inheritance” principle? If so, how? If not, why not?
   1. *no, they use a GameFramwork that uses the game*
3. Does the code violate the “Open for extension, closed for modification” principle? If so, how, if not why not?
   1. *yes, to add a new game, you have to modify the framework to check for the new game in the big switch statement, initilize it, and launch it*
4. Does the code violate the “Dependency Inversion principle”? If so, how? If not, why not?
   1. *Yes, because they do not program to an interface anywhere. Superclasses depend on concrete classes*
5. Does the code violate the “Hollywood principle”? If so, how? If not, why not?
   1. No, the individual games do not care about the framework

At this point, you should have a good understanding of what is wrong with the code. Design an alternative solution that supports better reuse of existing as well as similar new games. Get a TA to check off your UML diagram.

# Implementation (B)

Implement/refactor your code in the **Lab7-1/src/** folder.

# Testing Reusability (A)

Come up with a new command line game. It does not have to be very fancy. Implement the new game using your refactored code to convince yourself and the TA that you can add new feature without changing the refactored code.

# Deliverables

Demo your final code and sample runs to a TA to get the lab checked off.