COMP 3023 – Assignment 1

Sushi Slam System Design

Game Class

The Game class drives the game and after instantiation, creates the shuffled deck and both players. It loops three times, each time creating two hands of cards and subsequently calling round(), which calls turn() ten times, after which it performs end of round scoring, hand clearing and tableau clearing. Turn() manages presentation of hands and adding to the tableaus. The Game class has five private member variables: _hand1, _hand2, _player1, _player2 and _roundNumber. Game has a 1 to 1 or many relationship with Player class and a 1 to 1 relationship with the Deck class. Calling the destructor on Game will also destroy Player and Deck. Game needs to include the Deck class.

Deck Class

The Deck class has a private member variable called _cards which is a collection of cards. It includes member functions for populating and shuffling the _cards collection. Its destructor deletes all the cards in its collection. Deck needs to include the Card class and all its sub-classes.

Player Class

The Player class has four private member variables: an array of names called _names, _name, _playerScore and _tableau. _name is initialized when a Player is constructed based on a random choice from the array. Player needs to includee the Card class. It includes member functions to add a card to its tableau and to clear its own tableau. It calculates scoring for the round. Player needs to include the Card class.

Card Class

The Card class is abstract and cannot be instantiated. It has a public enum called CardType, its values being the types of cards of its sub-types. As it cannot ever be instantiated, the ‘publicness’ of CardType is not an issue. It contains three virtual member functions that are to be overridden by the child card classes.

Card sub-types (DumplingCard, MakiRollCard, NigiriCard, SashimiCard, TempuraCard)

Each sub type predictably has functions that overide the parent abstract class. One returns the enum CardType, one returns a string representation of this CardType and one (called score) calculates the score of all its type in the tableau of the current round. Each score implementation is quite different due to the different ways each type of card rewards points to the player. Each card sub-type includes the parent Card class and the Player class

*** My original design included four additional classes that I deemed not necessary (Tableau, Score, Hand & Round). As I worked through implementing the system I felt including them would increase complexity. Removing them has greatly simplified my system and class diagram as a result.**