Advanced Object Orientated Programming

Project Assignment

Introduction

This report details the design and implementation in C++ of a Blackjack/Pontoon game, using Object-Orientated Programming techniques.

Program Structure – Classes and Inheritance

The table below shows all the classes created for this project, a brief description of each, their inheritances, and a list of members and member functions (not including constructors and destructors.

Class Name	Inherits from	Description	Members	Member Functions
Card	N/A	A generic card class which can be used in any card game. Represents a single card with a rank and suit.	rank :: Rank suit :: Suit	rankName : :string suitName :: string getRank :: Rank getSuit :: Suit printCard :: void
PontoonCard	Card (public)	A card class specialised for the game of Blackjack/Pontoon by the inclusion of a member variable representing its value in the game.	value :: int	getValue :: int
Deck [TEMPLATE]	N/A	A class for a deck of any type of object. This is a template class, and therefore allows for polymorphism, so the deck can be used as a data structure to store any type of card or other object.	deck :: deque (see * below)	cardsInDeck :: int addCardTo Top :: void firstCard:: iterator lastCard :: iterator shuffle :: void
PontoonHand	Deck<PontoonCard> (public)	A class representing a player or dealer's hand in Pontoon.	handValue :: int	printHand :: void getHandValue :: int resetValue :: int addCardToHand :: void
PontoonGame	N/A	A class containing the game logic for a game of Pontoon	deck :: Deck< PontoonCard>	playAgain :: bool newGame :: void dealTwo :: void

			player :: PontoonHand dealer :: PontoonHand gameLog :: Gamelog	twist :: void playerRound :: void dealerRound :: void game :: void
GameLog	N/A	A class for logging the results of the game and creating a log file once the game has ended.	startTime :: string endTime :: string roundResult :: string logName :: string rounds :: int wins :: int losses :: int results :: vector<string>	

*The Deck template class was based on the double ended queue (deque) from the C++ standard library. This was chosen as the basis for the Deck data structure because it has efficient insertion or removal of elements at the end or beginning (O(1)) [1] , which is suitable for the behaviour of a deck of cards, where cards are nearly always removed one at a time from the top and very rarely removed from the middle.

Program Structure – Additional Files

Aside from the classes listed above, two additional files were created for the project:

- Enums.h – A header file which defines two enumerated types – Rank and Suit, which are used to give a rank and suit to an object deriving from Card.
- TestMain.cpp – A file containing a main function which simply creates an instance of type PontoonGame, thereby beginning a session of Pontoon.

Shuffling Algorithm

When writing the algorithm included in the Deck template to shuffle the deck of cards, I envisaged how a real pack of cards is shuffled and attempted to implement this:

Firstly, all cards are removed from the deck and placed into a temporary storage vector. Then one by one a card is picked at random and added back to the deck, until all cards are in the deck. This represents a real pack of cards being strewn over a table of other surface and then piled up in a random order to form a newly shuffled deck of cards.

References

[1] http://en.cppreference.com/w/cpp/container/deque (Accessed February 2018)