CSE116A,B Introduction to Computer Science II for Majors Fall 2000

Bina Ramamurthy

Project 2: Designing, Implementating and Testing A Large Scale Application
Preliminary Description

October 3, 2000

1 Objective

- Analyze a problem and design using object-oriented principles.
- Understand and design Abstract Data Types (ADT).
- Practice separation of interface and implementation.
- Creating and using Java Packages.
- Creating robust software using exceptional handling.

2 Problem Statement

Card games have been very popular for centuries. There are a number of games that are played with cards and each game has many variations. In all these games a collection of cards called the deck of cards is the central object. A deck of cards superbly exemplifies object reuse since the same deck of cards can be used to play many different games (applications).

You are required to design and implement a deck of cards abstract data type. You are also required to implement at least two applications, one of which is blackjack. The other application can be as simple as a high card game.

3 Project Description

3.1 Deck of cards

A deck of playing cards contains 4 suits (spades, hearts, diamonds, and clubs) of 13 cards each. The thirteen cards in each suit are A (Ace), K (King), Q (Queen), J (Jack), 10, 9, 8, 7, 6, 5, 4, 3, and 2. The common operations with a deck of cards include shuffling the cards (randomization), dealing or distributing the cards among a specified number of players and specified number of cards to a player, pick a card (top, bottom, random) from the pack. There may be other operations such as inserting a card in the pack, parting the card pack into to two and returning one, and provide count of the cards in the deck.

3.2 Application: Blackjack

This game has a dealer and one or more players. We will limit the number of players to 4. So there can be one or more players. We will use a single deck of cards and add one more deck if and when we run out of cards in a game.

The Bet and the Deal:
Cards are dealt one at a time, in rotation beginning at the dealer's left. Each player including the dealer gets two cards. We will assume that the cards are dealt face up except for the dealer who has one card up and one card down. Each players places his/her bet before the deal.
The Play:
The values of the cards are: Ace, 1 or 11, as the holder wishes; king, queen, jack, ten, 10 points each; any other card, its number. The object is to hold two or more cards that total 21 or a nearly 21 as possible without going over 21. For example, six, four, and Ace count 21; seven, four, and Ace count 12, to call ace 11 will put the player over 21. An Ace and a ten is 21 and also a *natural* and the bet is settled at once. The player who got a natural gets 1.5 times the bet amount, and the bet is settled right away. A player may bet only against the dealer. Both minimum and maximum are placed on the amount a player may bet.

After the initial deal, the dealer settles with each player separately. Each player in turn, may either *stand* (on his/her first two cards, or at any later time), or may want additional cards by saying *Hit me*. He/she may continue to draw additional cards, but once he/she says *I stand* he/she may draw no more cards. If the additional cards puts the player's total over 21, the player looses and the dealer collects the bet.

When every player except the dealer has either *stood* or gone over 21, the dealer turns up his/her face-down card. The dealer may take additional cards as long as his total is 16 or less and must stand when his/her total reached 17 or more.
Bet Settlement: If the dealer goes over, he pays each player who has stood, the amount they bet. If he/she stands on 21 or less, he/she collects from each player having a lower count, pays each player having a higher count, and has a stand-off with each player having the same count.

We will assume that after the game, cards are collected and re-shuffled.

4 Implementation Details

Your implementation will be very simple since the primary goal of this project is to exemplify the interface/implementation separation. For each design draw the class diagram before you proceed with coding.

1. Design and implement a package carddeck that contains the classes associated with a deck of playing cards. This package and the classes within it may be used for other applications. Any package will have associated exception classes. You may have to define this too.
2. Test the classes in carddeck completely before you move on to the design of the application.
3. Design and implement the application **Blackjack** that uses the carddeck package. Test it.
4. Design and implement any other application of carddeck. Test it.

5 Material to be Submitted

Will be given later along with detailed grading guidelines.

6 Due Date

11/10/2000 by midnight. On-line submission is required.