ELEC 2543 Object-Oriented Programming and Data Structures

Exercise 8a

Topics: Inheritance

Due Date: Mar 27, 2017

Overview: In this exercise, you will extend the playing card game you developed in Lab 6b to introduce diversity in the card ranking and player strategy. This exercise will get you prepared for Assignment 2 as well.

Deck of Cards and Trump Suit

In the default deck, the card ranking is the one we have been using in Lab 6a and Lab6b. That is, spade > heart > club > diamond, and A > K > Q > J > 10 > 9 > … > 3 > 2. SA is the largest card and D2 is the smallest one. D6 > S5.

In some card games, other card ranking is adopted. For example, a **trump suit** can be identified, and cards of this trump suit are the larger than cards of other suit. For example, if the trump suit is diamond, DA will be the largest card, followed by DK, DQ, …, D2. That is, the ranking of the cards is: DA > DK > DQ > … > D3 > D2 > SA > HA > CA > SK > … > S2 > H2 > C2.

The Game

There are three players (P1, P2, and P3). In the beginning, each player is given *k* cards. The cards are drawn from a deck of cards with a certain trump suit. That is, the cards with the trump suit are the largest card in the deck.

In each round, P1 removes a card from his hand at random. P2 removes the largest card, while P3 removes the smallest card. The player whose card is larger wins this round.

Class Deck

This class represents the default deck of cards. An arraylist is used to keep the cards. In the constructor, a copy of each possible card (different combinations of suit and rank) is put in the arraylist.

Method cardRank(Card card) and compareCards(Card c1, Card c2) are provided to compare cards using the default ranking. More details are in the program. YOU ARE NOT ALLOWED TO MODIFY THIS FILE.

Class TrumpDeck

This class represents a deck that has a trump suit. Apply inheritance to develop this class. You can assume the trump suit is passed as parameter to the constructor. Instance variables, constructors, and methods should be defined as needed, but you should override as few parent class members as possible. NO SHADOW VARIABLES ARE ALLOWED.

Class Player

This class represents the players. This is an abstract class. An arraylist is used to keep the cards the player has. A static variable is used to keep track of what deck the players are playing with. This variable is static as ALL players are playing on the same deck.

YOU HAVE TO APPLY AN APPROPRIATE VISIBILITY MODIFIER TO EACH INSTANCE VARIABLE.

Methods addCard(Card card) and printHand() are similar to those defined in Lab 6b. Method Card drawCard() is declared to be abstract so that different kinds of players have to provide their own definitions.

Classes RandomPlayer, LCPlayer, and SCPlayer

These classes represent players with different strategies. RandomPlayer objects select a random card from his hand, LCPlayer always deals the largest card, while SCPlayer plays the smallest card.

Apply inheritance to develop these classes. Instance variables, constructors, and methods can be defined as needed, but you should override as few parent class members as possible. NO SHADOW VARIABLES ARE ALLOWED.

Class CardGame

This class defines the game logic. The necessary instance variables are provided. The constructor takes two parameters: *k* and the trump suit. Method public void play() simulates the game.

Driver TrumpCardGameDriver.java

The driver program has been provided. You do not have to modify it. A sample output is provided in Moodle.

Handin

Submit your TrumpDeck.java, Player.java, RandomPlayer.java, LCPlayer.java, SCPlayer.java, and CardGame.java by zipping all the files into one single zip file to Moodle before the deadline.