COMPUTER SCIENCE 12B (FALL, 2021)
ADVANCED PROGRAMMING TECHNIQUES IN JAVA
DUE: TUESDAY, OCT 19TH, 11.59PM
PROGRAMMING ASSIGNMENT 4

Program Description:

For this assignment, you will be a designer of objects and a user of objects.

You will be designing a program that allows you to play the card games War at the terminal against the computer, at Iraklis's Cyber Casino. To make designing the game easier, you will create two classes, which are specified here:

CARD CLASS

Card class - the class representing a single card in a deck. Every card has three properties. The first is a "suit," which is either "Hearts", "Diamonds," "Spades", or "Clubs". A card also has a color, where Hearts and Diamonds are red, and Spades and Clubs are black. Finally, each card has a value from 1-13 (inclusive). Some values have special names: the 1 is an Ace, 11 is a Jack, 12 is a Queen, and 13 is a King.

Your first job is to design a class for a card, with each of these properties. It should have these methods:

1. **public Card(int value, String suit)** - initializes the Card and it's variables. For example, Card(12, "Hearts") will create a red 12 of hearts.
2. **public int getValue( )** - returns value of the card
3. **public String getColor( )** - returns color of the card
4. **public String getSuit( )** - returns suit of the card
5. **public String toString()** - returns string representation of the card, E.G. "King of Hearts." Or "7 of clubs"

DECK CLASS

In an entire deck, there are 52 cards, each of the 13 values for the four different suits. You should implement a class to handle all of the logistics of the deck, so that as a user of the class, all you have to do is pick cards from the deck, and then discard them once you're done with them. This class has a bit more logic in it, so here's what you have to do.

1. **public Deck()**

   This method will initialize the deck - creating an array of each of the 52 possible cards. After generating them, you should also shuffle them in the array (shuffling method is defined below). This method should also initialize a discard pile array, which currently has nothing in it. You may want to initialize other variables too, to assist with your other methods.

2. **public void shuffle()**

   This method shuffles all of the items in the deck. You can be creative with your shuffling algorithm, but I suggest you implement this one : https://en.wikipedia.org/wiki/Fisher%E2%80%93Yates_shuffle#The_modern_algorithm
   (the first algorithm under "The Modern Algorithm"). This is in "pseudo code," which means it's not in a real programming language, so it will be your job to convert this into real java code. Make sure if there are some "null" elements in your deck, that you don't shuffle them into the deck.
3. **public Card drawNextCard()**

   This method will give you the next card in the deck. Initially, this will be the card at index 0, then index 1, then index 2… up until index 51. After index 51, drawNextCard should take all of the cards in the discard pile, put them in the deck array, empty the discard pile, shuffle them, and then return the first one.
4. **public void discard(Card c)**

   This method will add the card into the discard pile. Initially it will add the card to the first index of the discard pile, then the second index, etc…

Hints:

1. Remember that assignments for Arrays are by reference. If you want the deck to be a copy of everything in the discard pile, you can just do:
   currentDeck = discardPile.clone();
   Then to empty the discard pile, you can just say: discardPile = new Card[52];
2. Be careful with shuffling that you don't accidentally shuffle "null" objects into the deck. This won't happen in the beginning when the deck is full, but when you're making the contents of the discard pile into the new deck, there will probably be some "null" elements at the end.

Feel free to add any other methods to these classes.

CLIENT CODE

Once you have these classes done, and have tested them to make sure they work, you can start using these objects to write the game in our casino, in a new file Casino.java.

Simple War Game

This game is simpler than the normal game of war, but the object is to have a card with a higher value than the dealer. The user makes a bet. Then, both the user and the computer draw a card. If the user's card has a larger value than the computer's card, then the user wins, and gets the value of their bet added to their total. Otherwise (including the case of a tie), the computer wins, and the user loses their bet. After each round, discard the cards. Keep playing until the user runs out of money, or until they say they don't want to play anymore.

The main method

Finally, you should have a main method that allows the users to play these games. There are many choices that are left up to you regarding the design of the game. For example, what do you do when the user runs out of money? How do you decide how much the user starts with? Will you implement a betting minimum and maximum?

Grading:

You will be graded on

- **External Correctness:** The output of your program should match exactly what is expected. Programs that do not compile will not receive points for external correctness.
- **Internal Correctness:** Your source code should follow the stylistic guidelines linked in LATTE. Also, remember to include the comment header at the beginning of your program.

Submission:

Create a folder with the name <Firstname_LastnamePA4> containing all your source code files. (eg. Iraklis_TsekourakisPA4). Make sure that you include a header comment with your name in every single source code file. As a last step compress the folder into a .zip file named <Firstname_LastnamePA4.zip> and upload it on Latte by the day it is due. For late policy check the syllabus. This is an individual assignment, and any sign of collaboration will result in a zero. Please check the syllabus for the academic integrity policy.