THPT3: Clock Solitaire Game Simulation

Overview

You are to write a program that simulates playing the solitaire card game Clock. We provide several sites at the end of this document that describe and demonstrate the rules; we suggest you familiarize yourself with the game first. Here's how we want you to implement the simulation.

Part 1 - Required Classes

Create a Card class that can represent each of the 52 playing cards: Ace, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, King … in each of the four suits: Spades, Hearts, Diamonds, and Clubs.

Implement the following methods:

- constructors
- getSuit()
- getValue()
- String toString(): ~~to print, e.g., "10 of Hearts" or "Queen of Diamonds" or "Ace of Spades"~~ **Update:** We are thinking that printing cards more compactly -- e.g., "10H" or "QD" or "AS" -- i.e. a one (or two) character value, followed by a one-character suit -- will make both debugging and our grading easier.

Create a Deck class that represents a standard deck of 52 playing cards.

Implement the following methods:

- constructors
- void shuffle(): randomly interchanges cards in the deck to simulate shuffling
- Card dealCard()
- int cardsLeft()
- String toString(): iterate through your deck array, printing all the cards, in a way that's convenient to check what cards are there; we think it'll be best to print all the cards on one line (or print them on four lines, if you want to get fancy); for cards that have been dealt, either don't print them, or print them separately, labeled as "dealt"

Create a Pile class that contains no more than five cards… some face down, and some face up.

Implement the following methods:

- constructors
- void addCardFaceDown(Card card)

- Card removeCard() - removes and returns the “top” face down card (null if there are none)
- int getNumberOfFaceDown()
- void addCardFaceUp(Card card)
- int getNumberOfFaceUp()
- String toString() - print the cards in the Pile; print the contents of a pile on one or two lines; label the portion of the pile that’s face up versus face down

Part 2 - The Clock Solitaire Game

Implement the Clock Solitaire Game simulation, with *all* the following capabilities. We strongly suggest you don’t ignore handling command line arguments, as this feature is simply to implement, and will be a great help in your debugging process, and in our grading. Name your primary class -- the one with the main() method -- `ClockSolitaire`.

Command-Line Arguments

Your program will accept two command line arguments. (These arguments are *optional,* and your program will use the “default” values if the argument(s) are not specified. And, implementing handling of command line arguments is *not* optional.)

1. Print level - one of the following three values:
    a. `verbose`
    b. `normal` (the default)
    c. `silent`
2. Number of games to play (default: 1)

Setup Tasks

- create thirteen Piles
    - these should be stored in an array of Piles (rather than thirteen discrete variables)
- create and shuffle a Deck of Cards
- deal out the deck to populate (initialize) the thirteen Piles with four face-down cards each
    - You may deal one card for each of the thirteen Piles, and then a second card, and so on…
    - or, you may deal four cards to fill the first Pile, four for the second Pile, and so on…
- print the thirteen Piles -- the game board (this will mainly uses Pile.toString()), if the print level is not “silent”; label each pile with its clock position; for simplicity, you can print one pile per line (you can get more creative, if you like)
- print the “score”: the number of Piles with at least one face-down card, if the print level is not “silent”; note a game is won when the “score” is zero
    - this will be 13 at the beginning of each game

Playing the Game

1. remove (top, face-down) card from Pile 13, the Kings Pile
2. add it, face-up, to the ("bottom" of the) correct Pile
3. if the print level is "verbose", print the game board -- the thirteen Piles -- as above
    a. also, keep track of and print the "step number," a counter that you increment each time you "remove-and-add" a card
4. remove (top, face-down) card from that same Pile
    a. one needs to check that there are face-down cards remaining
    b. if there are no face-down cards remaining, the game is over, if you're on Pile 13
    c. (if there are no face-down cards remaining, and you're not on Pile 13, then there's an error)
5. repeat this process by going to step 2

When Each Game is Done

- print the thirteen Piles -- the game board (as above), if the print level is not "silent"
- print the "score": the number of Piles with at least one face-down card, if the print level is not "silent"
    - a score of "zero" is a "win"; note this is rare
- increment a counter in a Scores array for the number of Piles with at least one face-down card
- repeat, from the top (the Setup), according to how many games to play, as specified in the command-line arguments

When You've Completed Playing the Required Number of Games

- print the number of games played
- print the Scores array, along with percentages, as follows; this will be 14 lines, with both the number and percentage of games that resulted in each score
- (print this for all print levels, including "silent")

References: Rules of the Game

Here are some sites that describe the rules of Clock, and one that allows you to "play" it online:

- https://www.thespruce.com/clock-solitaire-rules-412468
- https://en.wikipedia.org/wiki/Clock_Patience
- https://youtu.be/6AEJEf8L95g ; https://youtu.be/yUj320C9210
- http://cardgameheaven.com/single-player-games/clock.html
- http://www.novelgames.com/en/clocksolitaire/ (this site lets you play online)