(SavingsAccount Class) Create a SavingsAccount class. Use a static data member annualInterestRate to store the annual interest rate for each of the savers. Each member of the class contains a private data member savingsBalance indicating the amount the saver currently has on deposit. Provide a member function calculateMonthlyInterest that calculates the monthly interest by multiplying the balance by annualInterestRate divided by 12; this interest should be added to savingsBalance. Provide a static member function modifyInterestRate that sets the static annualInterestRate to a new value. Write a driver program to test class SavingsAccount. Instantiate two different objects of class SavingsAccount, saver1 and saver2, with balances of $2000.00 and $3000.00, respectively. Set the annualInterestRate to 3 percent. Then calculate the monthly interest and print the new balances for each of the savers. Then set the annualInterestRate to 4 percent, calculate the next month's interest and print the new balances for each of the savers.

(Card Shuffling and Dealing) Create a program to shuffle and deal a deck of cards. The program should consist of class Card, class DeckOfCards and a driver program. Class Card should provide:

a) Data members face and suit of type int.
b) A constructor that receives two ints representing the face and suit and uses them to initialize the data members.
c) Two static arrays of strings representing the faces and suits.
d) A toString function that returns the Card as a string in the form "face of suit." You can use the + operator to concatenate strings.

Class DeckOfCards should contain:

a) An array of Cards named deck to store the Cards.
b) An integer currentCard representing the next card to deal.
c) A default constructor that initializes the Cards in the deck. There are 52 cards in a deck.
d) A shuffle function that shuffles the Cards in the deck. The shuffle algorithm should iterate through the array of Cards. For each Card, randomly select another Card in the deck and swap the two Cards.
e) A dealCard function that returns the next Card object from the deck.
f) A moreCards function that returns a bool value indicating whether there are more Cards to deal.

The driver program should create a DeckOfCards object, shuffle the cards, then deal the 52 cards.