Programming Lab 4 – Decisions

These program descriptions are a little more open-ended. General guidelines are that all the "work" of the task is to be done via class methods. The test program will then consist of calls to those methods. You can decide if you want to use the test program to get user input or if you want to use a method for that. Displaying results, however, should be via class methods.

80% level

1. Write a program that prints all real solutions to the quadratic equation $ax^2 + bx + c = 0$. Have the user input *a*, *b*, and *c* and apply the quadratic formula. If the discriminant is negative, display a message that there are no real solutions. Implement a class `Quadratic` whose constructor receives the coefficients *a*, *b* and *c* as doubles. Supply the method **public void findRoots()** which calculates the roots if there are real roots, and the method **public void displayRoots()** if there are real roots. Supply a method **public Boolean hasSolutions()** that returns false if the discriminant is negative.

2. Write a program that takes user input describing a playing card using the following shorthand:

Notation	Meaning
A	Ace
2…10	Card Values
J	Jack
Q	Queen
K	King
D	Diamonds
H	Hearts
S	Spades
C	Clubs

   Your program should print the full description of the card. For example:

```
Enter the card notation:
QS
Queen of Spades
```

   Implement a class `Card` whose constructor takes the user input and whose `getDescription()` method returns a description of the card.

3. Write a program that prints the question "Do you want to continue?" and reads a user input. If the user input is "Y", "Yes", "OK", "Sure", or "Why not?", print out "OK". If the user input is "N", or "No" then print out "Terminating". Otherwise, print "Bad Input". The case of the user input should not matter. For example, "y" or "yes" are also valid inputs. Write a class `InputChecker` for this purpose.

4. Write a program that reads in four strings and prints the lexicographically smallest and largest one, for example:

```
Enter four strings:
Charlie
Able
Delta
Baker
The lexicographic minimum is Able
The lexicographic maximum is Delta
```

   Hint: the class should keep track of the current maximum and minimum

5. Enhance the BankAccount class from the previous assignment by
   a. rejecting negative amounts passed to the `deposit` and `withdrawal` methods
   b. rejecting withdrawals that would result in a negative balance

6. Construct a class **RockPaperScissors**. The user input should be R, P, or S (not case sensitive). The computer will then play Rock, Paper, Scissors with the user by choosing R, P, or S at random, and then reporting that the computer wins, the user wins or that it was a draw.

90% level

7. Write a **UnitConverter** class. Legal units are in, ft, mi, mm, cm, m, km. (Hint – One approach is to define one method that converts to meters and another that converts from meters.)

```
Convert from?
in
Convert to?
mm
Value?
10
10 in = 254 mm
```

100% level

8. Implement the class CombinationLock. A combination lock has a dial with 26 positions labeled A…Z. The dial needs to be set three times, <u>one letter at a time.</u> If it is set to the correct combination, the lock can be opened. When the lock is closed again, the combination can be entered again. If a user sets the dial more than three times, the last three settings determine if the lock can be opened. Implement and test the following interface:

```
public class CombinationLock
{
```

```
/* Constructor which constructs a lock with a given combination.
      The combination is a string of three uppercase letters. */
public CombinationLock(String aCombination)
{
      ...
}

/* Method to set the dial to a position, which is a string consisting
      of a single uppercase letter. */
public void setPosition(String aPosition)
{
      ...
}
/* Method to try unlocking the lock */
public void unlock( )
{
      ...
}

/* Method to check if the lock is unlocked. Return true if the
      lock is currently open. */
public boolean isOpen( )
{
      ...
}

/* Method to close the lock */
public void lock( )
{
      ...
}
```