

Combinations

Almost everyone is familiar with the standard 3 number combination lock with a rotating dial. Typically, such locks have numbers between 0 and 49, inclusive on the dial. In order to open the lock, numbers of the combination are entered in turn by turning the dial in alternate directions. I.e. the dial must be turned in one direction until the first combination number on the dial lines up with a given mark, then the dial must be turned in the other direction until the second combination number on the dial lines up with the same mark, and finally the dial must be turned in the original direction until the third combination number on the dial lines up with the mark. Then and only then is the lock open.

Write an object oriented program that acts as a simple memory quiz game that uses a variable object of class type *Lock* to represent a combination lock. (See class *Lock* design requirements below.) In this game, the user is shown a randomly generated 3 number combination and the direction of the first turn, which is the combination of the closed lock. When the user signals readiness, the combination disappears from view and the user will be asked to enter the first combination number and the direction of the turn. The game tells the user if his entry was correct or not. An incorrect entry at any stage of the game resets the lock to the starting conditions and forces the user to begin again with the first combination number and direction as his next guess. A correct guess lets the user proceed to the next number and direction in the combination. When all three numbers and directions are entered correctly in the correct order, the lock will open and the game will stop.

Class Lock Design Requirements

- Class *Lock* must model a 3-number combination lock with a rotating dial.
- The state of a lock object variable must be represented by private data members of class *Lock*. (I. e. An object variable of type *Lock* must not be dependent on any non-member variables.)
- Class *Lock* must have public method constructors and / or method functions that allows the initial combination and initial direction to be set via passed values.
- Class *Lock* must have public method functions as follows:
 - Method function *Turn*. When called, this method function will receive as parameters the guess for the next turn (number and direction), then return true if the guess was correct, false otherwise. If at any time the combination number and direction do not match, the next call to turn will start over testing the first combination number and direction.
 - When called, method function *Open* will receive no parameters, but will return true if the lock is now open, false if it is not.

- Class *Lock* must function as a “black box”, i. e. no data other than the boolean information returned by method functions *Turn* and *Open* can be accessed from outside a variable object of type *Lock*.
- No user I/O is allowed inside of the method functions of class *Lock*.

Example run #1:

```
Combination:
  starts Right
  26, 18, 47
Press <Enter> to start the game!
***** CLEAR SCREEN *****
Enter direction and number: R 26
Correct!
Enter direction and number: L 18
Correct!
Enter direction and number: R 47
Correct!
The lock is open!
```

Example run #2:

```
Combination:
  starts Left
  43, 30, 20
Press <Enter> to start the game!
***** CLEAR SCREEN *****
Enter direction and number: R 43
Incorrect! Begin again!
Enter direction and number: L 43
Correct!
Enter direction and number: R 40
Incorrect! Begin again!
Enter direction and number: L 43
Correct!
Enter direction and number: R 30
Correct!
Enter direction and number: R 10
Incorrect! Begin again!
Enter direction and number: L 43
Correct!
```

Enter direction and number: R 30

Correct!

Enter direction and number: L 20

Correct!

The lock is open!

Test Run:

Run your program, deliberately making at least 3 mistakes before correctly entering the combination.

Submit a printout of your program code and a printout of the runs of your program using the test data. [A printout of a screen dump is OK. "Copy and Paste" of text output to an editor, then print from the editor is also OK.]