**Task:**

Managing a Combination for a lock. The program should accept several console commands (see table below) and create a lock sequence of the format:

Rx Lx Rx Lx Rx Lx Rx, where L and R represent the direction of the turning of the dial and the x represents a value between 0-49; a stopping spot on the dial.

* An example would be: L10 R41 L15 R21 L48 R1 L33

**Requirements:**

* A console program which accepts a command line parameter which specifies the name of a file in which to store the final combination.
* You must write your own Linked List class.
* Keep the combination numbers in a linked list, one in each separate node.
  + You may **not** use the classes provided in the STL for the list. They may be used to perform console I/O or data manipulations.
* The program begins by generating a starting combination of R0 L0 R0 L0 R0 L0 R0. Note:
  + There are exactly 7 numbers in the combination, ranging from 0-49. Numbers may be repeated.
  + Each number is associated with a direction of rotation of the lock’s dial. These directions are L (Left) and R (Right). So each entry is of the form <Direction><Value> (e.g. L14, R22)
  + The <Direction> value always alternates and the sequence always starts with R (for R15 L22 R41 L8, etc.)
  + The program must display the current lock sequence at all times, and clearly indicate which of the lock values represent the CURRENT working position.

The program shall accept these console commands to manipulate the various values in the lock sequence:

|  |  |
| --- | --- |
| Command | Program Action |
| Q (Quit) | The Q command signifies exit immediately without saving changes. |
| E (Exit) | The E command signifies exit and saving the combination in the text file passed in as a command line parameter. |
| G (Goto) | The G command allows you to set the position of the CURRENT working lock value to support the D and S commands. For example, G 4 sets the CURRENT working lock value to the fourth position in the sequence. The CURRENT working position should be indicated to the user at all times. |
| S (Substitute) | The S command allows the user to swap a new value in for an old value. The S is followed by one space followed by a valid <Value> (0-49). Any other variants should be rejected.   * The S should Swap the CURRENT lock value and replace it with the new value. The Direction must be preserved from the current value. |
| D (Delete) | The D command may be entered at the console to delete the CURRENT working value in the combination.   * The D should Delete the CURRENT working value and replace it with <Direction>0 (Zero). It should re-display the current value of the new combination. The direction should be preserved. For example, a value of R14 would become R0. |
| R (Reset) | The R command re-sets the combination to all zeroes, preserving the <Directions>; display it and set the CURRENT working position to the first number in the sequence |

**Notes**:

Remember, you may not use any libraries except iostream and anything to support using regular expressions to parse and validate user input.

Evaluation:

This assignment is worth 36 marks. Please see the marking rubric below.

Assignment Notes:

The assignment must be demonstrated to the instructor on or before the due date during class.

If your assignment is late please send an e-mail to the instructor, hal.o’connell@nscc.ca, to confirm submission. This e-mail will constitute the timestamp for evaluating any late penalty the assignment may incur.

See the **Marking Rubric** below.

| Criteria | Below Standard | Developing | Acceptable | | Professional | Marks |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | | 3 |
| E command | * Exit does not work | * Exit does not work completely, output file not created or contains invalid data | * Exit works as expected but file not processed from command line | | * Exit creates the output file specified as an argument and all data is correct. If the file exists it overwrites the old file. | \_\_\_\_\_\_ |
| Q Command | * Quit does not work | * N/A | * N/A | | * Quit works as expected | \_\_\_\_\_\_ |
| G Command | * GOTO does not work. | * Sets the current working position but does not indicate current position satisfactorily. | * N/A | | * GOTO works as expected |  |
| D Command | * Cannot delete lock codes from the combination | * Large errors exist with the Delete function * Program does not handle invalid code numbers or directions of rotation | * Codes can be deleted but direction is not correctly preserved * Other small errors exist with Delete Function | | * Delete functions without errors through full range of expected function. | \_\_\_\_\_\_ |
| S Command | * Cannot Substitute lock values. | * Large errors exist with the Delete function * Program does not handle invalid code numbers or directions of rotation | * Codes can be swapped but direction is not correctly preserved * Other small errors exist with Substitute Function | | * Substitute functions without errors through full range of expected function. * Substitute handles invalid line numbers * Code is error free | \_\_\_\_\_\_ |
| Aesthetics of Output | * incorrect or non existent use of whitespace in output * output is confusing and hard to follow | * fair use of   whitespace   * most output is clear, but poorly presented | * good use of whitespace * output is clear and well presented | | * excellent use of whitespace   output is clear and attractively presented | \_\_\_\_\_\_ |
| Readability | * source code is poorly organized and very difficult to read | * source code is fairly easy to read, but is hard to follow in some areas | * N/A | | source code is exceptionally well organized and easy to follow | \_\_\_\_\_\_ |
| Reusability | * source code cannot be reused * no functions or classes used | * Minimal reuse; Source not separated in to .h and .cpp files | * N/A | | source code could be easily reused with few modifications | \_\_\_\_\_\_ |
| Efficiency | * contains large portions that could have been easily reduced using a different method * too much code is replicated, copy /pasted | * tried some methods to improve efficiency * can explain what they attempted | * N/A | | * very clean and efficient code   can propose new ideas for improvement | \_\_\_\_\_\_ |
| Comments | * little to no comments used | * comments are used, some are meaningful and easily understood * some files and functions have headers | * N/A | | * not over/under commented * comments are meaningful and easily understood * files and functions have headers   Code is self-documenting | \_\_\_\_\_\_ |
| **Naming** Convention | * no standard naming convention followed | * a standard naming convention was used for part of the program, but deviated often | * N/A | | industry standard naming convention used throughout the program | \_\_\_\_\_\_ |
| Consistency | * no consistency in formatting or layout of source code | * source code formatting was present but inconsistent with whitespace, brackets, etc | * N/A | | source code formatting never deviated from the programmer’s layout | \_\_\_\_\_\_ |
| Total | | **36** |

0 - Assignment not submitted or work not original.