APCS Lab 5 – Iterations (Looping)

For all of these, make sure the user gets clear instructions and descriptions.

80% level

1. Rewrite the test program and class for Problem 4 in Lab 4 so that the user continues to enter words until the user enters the word “end”. Then the class displays the lexographic minimum and maximum. The words to be tested could potentially start with either lower case or upper case letters.

   A sample run could look like:

   ```
   This program will keep track of the lexographic
   minimum and maximum of a list of words you will enter,
   without regard to upper and lower case. When your list
   is complete, enter “end”.
   Enter a word:
   Delta
   Enter a word:
   able
   Enter a word:
   Charlie
   Enter a word:
   end
   The lexographic minimum is able
   The lexographic maximum is Delta
   ```

2. Rewrite the RoachPopulation class for Problem 5 in Lab 3:The wait method should now be **wait(integer n),** where n is the number of successive wait periods. And the spray method should now be **spray(integer n)** where n is the number of successive sprayings. Adapt your test program accordingly. Round any non-integer roach count down to the next integer.

   A sample run could look like:

   ```
   This program keeps track of a roach population. You
   will specify a starting population and then number of
   waiting or spraying periods. Each spraying will
   decrease the population by 10%. Each waiting period
   will double the population.
   What is the initial roach population?
   100
   How many sprayings would you like?
   3
   The roach population is now 72
   How many waiting periods would you like?
   4
   ```

```
The roach population is now 1152
```

3. Write a class called **Stats**. The constructor will take no input. There will be a method **addData(double a)** which will be used to add a value from the test program. Methods **getCount(), getAverage()** and **getStandardDeviation()** will return the appropriate values as doubles. The formula you can use to calculate standard deviation is $s = \sqrt{\frac{\sum x_i^2 - \frac{\left(\sum x_i\right)^2}{n}}{n-1}}$. The test program will ask the user for the number of values to be entered, then loop to enter those values. After the values are entered the test program will display the count, the average and the standard deviation.

   A sample run could look like:

   ```
   This program will find the average and standard
   deviation of a list of numbers you will enter.
   How many values will you enter?
   5
   Enter your values one at a time:
   3
   4
   5
   6
   7
   You entered 5 values.
   The average is 5.
   The standard deviation is 1.58114
   ```

4. Write a program that asks the user for an integer and then prints out all its prime factors.Use a class **FactorGenerator** with methods **nextFactor** and **hasMoreFactors.**
   Option: You may adapt the assignment to use methods of your own choosing.

   A sample run could look like:

   ```
   This program will find the prime factors of any
   integer.
   What is your integer?
   150
   The prime factors are:
   2
   3
   5
   ```

5. Write a program that prompts the user for n and prints the first n values of the Fibonacci Sequence on a single line, separated by commas. Assume the Fibonacci Sequence starts 1, 1, 2, 3,... (not 0, 1, 1, 2,...).Use a class **FibonacciGenerator** with a method **nextNumber**.
   Option: You may adapt the assignment to use methods of your own choosing.

   A sample run could look like:

   ```
   How many terms of the Fibonacci sequence do you want?
   10
   The Fibonacci terms are:
   1,1,2,3,5,8,13,21,34,55
   ```

90% level

6. Write and test a class that will reverse any string that a user enters.
7. Write a program called **CozaLozaWoza** which prints the numbers 1 to 110, 11 numbers per line. The program shall print "Coza" in place of the numbers which are multiples of 3, "Loza" for multiples of 5, "Woza" for multiples of 7, "CozaLoza" for multiples of 3 and 5, and so on. The output shall look like:

   ```
   1 2 Coza 4 Loza Coza Woza 8 Coza Loza 11
   Coza 13 Woza CozaLoza 16 17 Coza 19 Loza CozaWoza 22
   23 Coza Loza 26 Coza Woza 29 CozaLoza 31 32 Coza
   ......
   ```

   100% level

8. Write a program called TimeTable to produce the multiplication table of 1 to 9 as shown using nested for-loops:

```
 * |  1  2  3  4  5  6  7  8  9
-------------------------------
 1 |  1  2  3  4  5  6  7  8  9
 2 |  2  4  6  8 10 12 14 16 18
 3 |  3  6  9 12 15 18 21 24 27
 4 |  4  8 12 16 20 24 28 32 36
 5 |  5 10 15 20 25 30 35 40 45
 6 |  6 12 18 24 30 36 42 48 54
 7 |  7 14 21 28 35 42 49 56 63
```

```
8 |   8 16 24 32 40 48 56 64 72
9 |   9 18 27 36 45 54 63 72 81
```

9. Modify the class from Exercise 8 to do the table from 1 to 12, making sure the numbers line up in the table.