Project #3

Prof. John P. Baugh – CIS 2353 – Oakland Community College – OR

Points:	___________ / 125
Due:	March 4, 2018 at 11:59 p.m.

Objectives

- To understand sorting algorithms better

Instructions

You may **pick one** of the two following exercises to do as your project 3.

Option 1: Experimenting with Sorting

Implement (or use built-in algorithms for) `QuickSort`, and `InsertionSort`. Compare their running times on sets of integers of size 10, 25, 50, and 100. Which appears to perform better as the number of integers increases? Why did you get the results you did?

Make sure to run each algorithm at least 5 times on each set size (five times for set size of 10, five times for 25, etc. – for EACH algorithm.)

Hint: Make use of the `currentTimeMillis()` method of System.

E.g.,

```
long startingTime = System.currentTimeMillis();
long total = 0;
//do something here:  run an algorithm to test timing on
long stoppingTime = System.currentTimeMillis();
long elapsed = stoppingTime - startingTime;
System.out.println(elapsed);
```

This obtains the current time from the system at the beginning, then after the algorithm being tested is performed, get the stopping time. Then, find the difference, referred to as the **elapsed time**, and do something with it. Record in a table for this assignment.

Option 2: Radix Sort for Strings

Write a version of Radix Sort to sort an array of `String` objects. The String object will only contain ASCII characters, and the maximum size (length) of any `String` object is 20.

Hint: Use the `charAt()` method of the `String` class.

Run the algorithm on a set of at least 10 strings and have it print the results. See the Radix implementation in the book to see how it works. Note that it is not a comparison sort – you don't compare two strings. It involves looking at subcomponents of the items being sorted. In the case of strings, these subcomponents are characters.

Deliverables

- **Create a zip file** of your .java files or charts and answers to questions (in PDF or Word format) and turn in the zip file. Name the zip file "Project3" and D2L will take care of putting your name in it.

- You will also need **screen shots of your program working**, pasted inside of a PDF or Word (.doc or .docx) document (you can create PDF from Word documents using the Save As… option)
  - You can include **both** the screen shots and the answers to questions in the same document/PDF file

- Also, make sure your name is in comments on **each** Java file that you turn in. For example:

```
// Profunda MacPuffins
// CIS 2353
// Winter 2018
// Prof. John P. Baugh
```