COMP 196ABL – Project #8
24 points; due date - end of lab session you attend on March 27, 2019
Email your Java source file to: sgs@csun.edu

Processing Employee Information

For this project, we have a file of weekly employee input records that look like the following:

```
Martin  40  23.50
Smith   34  27.00
Johnson 44.5  21.75
Wong    47.5  23.40
```

Where the first word of each line is a single name of an employee, followed by the number of hours that employee worked this week and then the hourly pay for that employee. At least one white space character must separate each value read.

Download the file EmpPay.java from the week 8 folder and complete the follow:

1. A method called loadRecordsFromFile() is to be written that interactively requests the user for the name of an input data file which is then opens and reads, placing the data into the appropriate arrays declared in the main() method. After all of the data has been read in, the method returns a count of the number of employee records read in. If the user elects not to enter the name of a file, the method immediately returns the value zero.

2. A method called printRecords() is already written but that method calls outputRecords() which needs to be written. printRecords() causes the employee information to be written to the terminal window with the following format:

   ```
   Martin 40.0 23.5 0.0
   Smith 34.0 27.0 0.0
   Johnson 44.5 21.75 0.0
   Wong 47.5 23.4 0.0
   ```

   The format is: name, a blank, hours worked, a blank, rate, a blank, pay

3. A method called calculatePay() is to be written that calculates and stores for each employee their weekly pay based on the hours worked and pay rate information. If an employee has worked over 40 hours they are to receive 1.5 pay for the hours over 40.

4. A method called sortRecords() is to be written that performs a binary sort (with early completion test) on the employee records based on the specified sort key. sortRecords() is to utilize the support methods isAlessThanB() and swapAwithB(); both also needing to be written. You are prohibited from using any Java library routines related to performing a binary sort. (See next page for review of bubble sort algorithm.)

5. A method called writeRecords() is to be written that requests the name of an output file from the user and then writes a report identical to that for printRecords() except that this report is written to the output file. If the user elects not to enter the name of a file, the method immediately returns without creating a report.

Bubble Sort Algorithm

Given an array of length n, to sort the items of the array into ascending sequence (smallest in first position, largest in position n):

```
For each element at position k from first position to position (n-1) [top down scan]
        For each element at position p from position n to position (k+1)
                                                                [bottom up scan]
                If the value at position p < value at position (p-1)
                        Swap the values at positions p and (p-1)
```

The check for early completion is if no swaps occur for a given value of k then the sort has completed without k necessarily reaching (n-1).