Assignment 06 – Building a Sorted Linked List of String Objects in PEP/8$_2$ (Due: Monday May 9th 2016)

For this assignment you are to make use of the "Heap Components" to develop a PEP/8$_2$ source program that reads an "empty string terminated" sequence of string values and represents these values in a linked list such that these string values are arranged in lexicographic ascending order. As such, the program will implement what is known as an "Insertion Sort."

In part, you are to provide an implementation for the following subprogram:

```
address readSO();
```

This subprogram will read a string value and return the reference to the dynamically allocated (i.e. in the Heap) string object that represents the value read. Your implementation of this subprogram is to be consistent with my implementation of the `readStgs` subprogram posted in the AS05 folder on the CWS; the one that uses the heap. Really, you will be able to "borrow" much of the behavior and instructions from that version of `readStgs` to implement `readSO()`. This is because `readStgs` reads a sequence of string values and stores their references in an array, while `readSO` only needs to read a single string value.

(Hint: In the spirit of incremental development, I encourage you to consider refactoring the given `readStgs` so that it then calls upon `readSO`. This basically means factoring part of the instruction sequence out and placing in the subprogram and then calling upon it. In this way, you can then readily test your implementation of `readSO` within the context of AS05.)

In addition to the `readSO` subprogram, your program should have two other subprograms (at least).

```
address buildLst();

void prntLst(address head);
```

`buildLst` will essentially be the refactored version of `readStgs` that will read the input and build the linked list, returning as its result the address to the first node in the list (i.e. that would be the reference to the dynamically allocated node representing the lexicographically first string value). `prntLst`, as it's name indicates, is given the reference to the first node in a linked list of string objects and prints out the values in the list, one per line.

(Hint: Again in the spirit of incremental development, consider how you might develop and test `prntLst` independent of the other parts of this program. Consider the following declarations.)

```
head:    .ADDRSS     first     ;Reference to the first node in the list
;-------------------------------------------------------------------
first:   .ADDRSS     second    ;First Node  - reference to next node
         .ADDRSS     two       ;First Node  - reference to string object
second:  .ADDRSS     third     ;Second Node - reference to next node
         .ADDRSS     three     ;Second Node - reference to string object
third:   .ADDRSS     fourth    ;Third Node  - reference to next node
         .ADDRSS     four      ;Third Node  - reference to string object
fourth:  .ADDRSS     0         ;Fourth Node - reference to next node (null in this case)
         .ADDRSS     one       ;Fourth Node - reference to string object
;------------------------------------------- (String Objects follow)
one:     .BYTE       20
         .ASCII      "Washington, George\x00"
two:     .BYTE       12
         .ASCII      "Adams, John\x00"
three:   .BYTE       18
         .ASCII      "Jefferson, Thomas\x00"
four:    .BYTE       15
         .ASCII      "Madison, James\x00"
```

You are to use the "submission form" provided for Assignment 06 on the Course Web Site (CWS) to submit your work on this assignment. The following files are to be submitted:

- `readSO.pep2`
- `buildLst.pep2`
- `prntLst.pep2`
- `AS06.pep2`, the main component serving as the "main program"

As usual, all source program files must be appropriately and usefully commented. In particular, there must be a comment block at the beginning of each file that (a) indicates that it is part of a solution to Assignment 6 in CMPS 250 for Spring 2016, (b) identifies the person who developed the program and is submitting it, (c) acknowledges all persons (by their full names) who collaborated with the submitter in completing this assignment (or state that you worked alone), and (d) point out any flaws or deficiencies in the program of which you are aware.

Good luck,
P.M.J.