EE 2310 Laboratory Experiment #6: Letter Insertion Routine

1. **Introduction:** Experiment #6 furthers your experience in more complex assembly language programs. You will work outside the lab to do the programming with your partner. Once again, student teams are urged to work together (although you can start out working separately), prior to submitting your program to the class TA.

2. **Goal of this exercise:** Design a recursive assembly language program. Program a recursive algorithm which will place random letters in an already-existing alphabetical list. A single letter will be input from the console at a time, which the program will then insert into a list of alphabetized letters defined in the data declaration.

3. **Experimental Equipment List:** The following items are required:
   - **Pervin text for reference.**
   - **SPIM installed on your computer.**

4. **Pre-Work:**
   - **Make sure that you have completed all software reading assignments (Pervin and P&H).**
   - **Make sure that you and your partner know how the program is to work (if necessary, flow-chart the program to make sure that you understand the sequence of events as the program executes).**
   - **Lab partners should work together to verify the program before submitting to the TA.**

5. **Program Description:** Write a program as follows:
   - **The program accepts a single-letter keyboard input at a time. The letter inputs must be lower case a through z (no upper-case or non-letter characters allowed; only lower-case letters). Only one letter must be input at a time; make sure that the character is ONLY a lower-case letter.**
   - **In the data declaration, declare an alphabetized string of letters (named "str"), a-z.**
   - **The program must input a character from the keyboard, determine its place in the string, and make a space in the string for this character, inserting it in the proper alphabetical order.**
   - **After the alphabetized string of 26 letters, declare a sequence of 30 nulls (using the ".space" directive), as expansion space for the string as you insert characters into the alphabetized list.**
   - **At any point, you can ask to print the current string (i.e., with as many letter inserts as have been made so far) by inputting capital P from the keyboard.**
   - **Finally, this program MUST use a recursive routine to insert the letter in the correct point in the program. Actually, using recursion is not the only way to accomplish this programming task, but it does make it relatively straightforward.**

6. **Comments About The Program:** This more difficult program represents a nice "design" opportunity. It should not be too difficult if you have consistently done class homework and Lab. 5. Remember to store the contents of $ra on the stack each time you jal, since your program may call itself quite a few times as it finds the correct location for each input letter.

7. **Experimental Procedure:** As before:
   - **Review the program specification and diagram the program flow if necessary to understand it.**
   - **Partners write the program, working together as much as possible.**
   - **Once the program runs, make sure that the output string is correctly alphabetized.**
   - **When your program runs correctly, submit one copy in NotePad to your TA via email.**

8. **Laboratory Report:** A laboratory report is not required.