

File Processing Assignments

I will be attending a conference on Friday, March 6. As a replacement for class, you will have the following “outside” assignment. You may either: 1. Meet as a group on Friday during the regular class time and work on this project together submitting a single solution, 2. Meet as a group on Friday and then continue working on this on your own submitting your personal solution, or 3. Work on this entirely on your own on your own submitting your own solution. You may not consult with others in the class about this project outside of the Friday time slot. This will count as three routine assignments.

1. For this program, you will write a set of three functions **display1()**, **display2()**, and **display3()** where the later functions refine earlier functions. **display1()** should read the contents of a file and display it on the screen. The name of the file will be hard coded into the function. **display2()** should prompt the user for the name of the file rather than have the filename hard coded into the function. Otherwise, it is the same as **display1()**. **display3()** will display the file with line numbers for each new line. Otherwise, it is the same as **display2()**.
2. Write an interactive program **copy()** that makes a copy of a file. It should prompt the user for the name of the existing file and then the name for the new file that will be created. The new file should be an exact duplicate of the original.
3. Write an interactive program that makes a copy of a file where the contents of the copy are the reverse of the original. That is, the last character in the first files will be the first character in the new file, the next-to-last character in the first will be the second character in the new file, etc. If you run this program twice, you should have a copy of the original file. Hint: use the second problem as a starting point.
4. Write a function **stats()**. This function should prompt the user for a file name and should display summary statistics for the file. Specifically, it should display the number of characters in the file, the number of lines in the file, and the number of words in the file. You may assume that the file is well formatted with single spaces between words and new-line characters at the end of each line. (As the course progresses, we’ll see a better way of approaching this problem.) There are two files on the class website that you can use to test your code. “`snippet.txt`” contains 4 lines, 25 words, and 106 non-blank characters while “`austen.txt`” contains 19 lines, 351 words, and 1454 non-blank characters. Hint: use the string library described earlier in Chapter 4.

To submit these programs, you should email me your solution. This should be a single email attachment of a text file that contains each function clearly identified. I should be able to paste this file into IDLE and run your code.

Due: Emailed by the start of class, Wednesday, March 11, 2009