Assignment 5

CS2340.005 Fall 2022

Floating Point and Sorting

As you have seen, floating-point numbers are handled differently from integers. There are three parts to this program.

**Part 1:** The main loop asks the user for a double-precision number. Use system call 7 to read the number. If the number is zero, exit this loop and go to part 2. If the number is not zero, store it in memory. I hesitate to say “store it in an array,” but that is what you will be doing. You can use the .space directive to allocate this. Since assembly language doesn’t really understand anything but memory (no type checking) this works. Keep track of how many numbers you have. Do not print the number. You may assume there will be no more than 100 numbers entered.

**Part 2: Sorting.** Write a function (do not fall through to in-line code) that sorts the numbers from smallest to largest. You can use any sorting algorithm you know; I suggest bubble sort because it is easy and you probably know it. Parameters to this function are: $a0 contains the count. You can code this function in the same module as your main program so it can use the name of your list of entered numbers.

**Part 3: Printing.**  Write a function, called from your main program, that prints the sorted numbers one per line followed by the count of numbers entered, the sum and the average. You can sum and average the numbers in this function. You can also code this function in the same module as your main program so it can use the name of your list of entered numbers. Register $a0 will contain the number of items in your list.

To recap: Your main function contains a loop that asks for numbers and puts them in a list. When zero is entered, exit the loop and call the two functions, then exit.

Your program will be tested with valid floating point numbers only, so no need for error checking. Your program must not crash for valid input. (Your program will crash if you press Enter alone to exit. This is normal behavior, not an error, so don’t do it when you test. Enter zero.)

**To hand in through eLearning:** A Zip file named CS2340-Asg5-<netID>.zip where you replace <netid> with your netID. For example, if I were to hand it in, the name would be CS2340-Asg4-jxc064000.zip. This must contain ALL files necessary to assemble and run your program.

|  |  |
| --- | --- |
| Grading Criteria | |
| Comments and variable names | 15 |
| Correctly sorts the file | 20 |
| Correct sum, count, and average | 30 |
| Correct program structure | 25 |
| Does not crash for valid input | 10 |
| Total | 100 |

Sample output if the numbers entered were 1.1, 3.3, and 2.2

Sorted list:

1.1

2.2

3.3

Count: 3

Sum: 6.6

Average 2.2

Additional grading criteria:

1. Sort not implemented at all: -20
2. Sort implemented but does not work correctly: -3 to -15
3. Inline code instead of functions. You must write a separate sort function: -10 to -25
4. Spaghetti code: -2 to -10
5. Issues with comments and/or variable names: -1 to -15.