Matt Skeins

Average.asm

I started implementing this project by asking for a user to enter an integer representing the number of floats they would wish to average. Then I take that number and shift it left by 2 (multiply by 4). This gave me the amount of space to allocate into my array. Using a syscall I allocated space in the heap and assigned a memory address to a register. Once I allocated my array I jumped to my looping function. Inside my looping function I kept track of a counter and the total spaces in the array(incremented by 4 to move spaces in array). Every iteration it would check if my counter equals the total spaces in the array. If not, ask for a float input from the user. This input would then be stored in the array at the space the counter represents. Once my counter is equal to the total space in the array it jumps to the average function. The average function just keeps track of the counter from the jump function. If the counter equals zero the function will branch to exit. If it does not equal zero it will pull the float from the array at position counter and store it in a $f register. The float is then added to another register that has the sum of all the floats. After this completes it will jump to the exit function the exit function takes the integer input from the user and converts it to a float. Then it divides the sum of all floats inputted by the now converted integer to receive the average. Then it will print the average to the screen and finish.

I had a lot of issues with his project even though it was straightforward. My main confusion was how to create the array of floats. This may have been a result of me not entirely understanding how floats are stored in memory. This caused me to really look over the slides about storing floats. My biggest takeaway was a strong understanding of the float structure in memory which I don't think I would have if I didn't look back over the material to understand.