Jackson Myers
Computer Architecture
Dr. Kwangsung Oh
Project 1

Brief Overview of Assembly Code

The program starts in main, where I define all of the parameter values for the program (input base address, count, output base address, and num). Then, to link $ra to main and jump to countNumbers, I use jal countNumbers. Inside of count numbers, we iterate through the input array. For every number in the array, we call the function increaseCount, which will increment that value as an index in the output array. This explains why the value limit is 49, so we can define the output array to be a size 50 with all zeroes. If the number from input is out of the range 0-49 inclusive, we print out an error message with the number that caused the error, and set the return value to 10 (signaling a program exit) then use syscall to actually perform the exit.

Now that the output array has the count of every number in the input array, we can return back to the main function. This return to the main function only works because I created a stack in the countNumbers functions that will store the return address and a couple other values. When we are ready to jump back to main, I can recover these values. This way, I have the proper return address to jump back to main.

Now, inside of main, the return we just did will bring us right back to where we left off. Now we want to call the printResult function. This function will print out only the counts of numbers that are greater than 1 in the output array. In the function, i represents what number it is, and to the right of the colon, we will print out the count of the number. Once we're done looping through the output, we want to return to main. Here, we don't need a stack because we're only one function deep, so $ra is holding the correct return address to get back to main. So we use jr $ra to return to main.

At this point, the program is done, so we can set $v0 to 10 to signal an exit and use syscall to officially exit the program.

That is all.