You’ll do some typing (having things go through your fingers will help you remember; seriously, there are experimental studies on this) for this lab, and then leverage off the code to extend it.

1) In lab8a.s, create and run the following. We’ll use a system call to print a number, and to print out an ASCII string

```
data
x: .word 42
message: .asciiz "Hello world!\n"

.text
main:
    addi $sp, $sp, -4
    sw $ra, 0($sp)

    # Print the number
    lw $a0, x
    li $v0, 1
    syscall

    # print hello world
    la $a0, message
    li $v0, 4
    syscall

    # return to the operating system
    lw $ra, 0($sp)
    addi $sp, $sp, 4
    jr $ra
```

Things to note: we are saving the return address on the stack here.

And restoring the return address here. The stack is your friend!

2) Knowing what you know from the question above, and from lab 7 last week... create lab8b.s, a program that will print out the numbers 1 to 20, inclusive, one number per line.

You should use a loop.
3) Now we’ll do a subroutine to add up the numbers in an array. Use $a0 as a pointer to the array, and $a1 as the length of the array. The subroutine returns the sum in $v0. We copy that value into $a0, so that we can print it out.

.data
x:    .word 10
      .word 20
      .word 30

message:  .asciiz " is the total.\n"

.text
main:  addi $sp, $sp, -4
       sw $ra, 0($sp)

       la $a0, x
       li $a1, 3

       jal sum

       add $a0, $v0, $0
       li $v0, 1
       syscall

       la $a0, message
       li $v0, 4
       syscall

       lw $ra, 0($sp)
       addi $sp, $sp, 4
       jr $ra

sum:   li $v0, 0
loop:  beq $a1, $0, done

       lw $t0, 0($a0)
       add $v0, $v0, $t0

       addi $a0, $a0, 4
       addi $a1, $a1, -1
       j loop

done:  jr $ra

4) Modify the code for lab8d.s, adding register $a2, with the number 15. Modify the sum subroutine so that we add up ONLY numbers greater than the value in $a2. You’ll need to implement an "if" statement.