

```
1: #Travis Ritter, Section: 01
2: .data
3: Prompt1: .asciiz "Enter N: "
4: AnswerMsg: .asciiz "Number of 1's: "
5: Error: .asciiz "Number must be >= 0"
6:
7: input: .word 0
8: output: .word 0
9:
10: .text
11:
12: main: #loads prompts, accepts user input, and prints results
13:     li $v0, 4 #load string print service
14:     la $a0, Prompt1 #load string into a0
15:     syscall # print
16:
17:     li $v0, 5 #load int reading service
18:     syscall #read an int
19:
20:     blt $v0, $0, error #if the user input is < 0, output an error,
21:                     #since the specifications ask for a positive number
22:
23:     sw $v0, input #store user input into global variable input
24:
25:     lw $a0, input #stores user input into function argument a0
26:     jal countOnes #jumps to countOnes function
27:     sw $v0, output #stores function return in global variable output
28:
29:     li $v0, 4 #load string print service
30:     la $a0, AnswerMsg #load message into a0
31:     syscall #print message
32:
33:     lw $a0, output #move the sum total to a0
34:     li $v0, 1 #load int print service
35:     syscall #print int
36:
37:     li $v0, 10 #load system exit service
38:     syscall #exit program
39:
40: countOnes: #function that counts the number of 1's in a positive binary number
41:     addi $sp, $sp, -8 #subtracting from the stack creates space, subtracting 8 bytes
42:     #makes space for two words
43:     sw $ra, ($sp) #stores the return address in the first position of the stack
44:     sw $s0, 4($sp) #stores the local variable (s0) on top of the ra,
45:     #this is the number to be added
46:
47:     li $v0, 0 #if the below branch is true, the function should return 0,
48:     #this is the base case
49:     beq $a0, 0, endFunction #when we shift the number to the right, eventually,
50:     #it will be all zeroes in that case, we branch away
51:     #as there are no 1's left to count
```

```
52:
53:     andi $s0, $a0, 1 #checks right most bit against 1, so if it is also 1,
54:         #this will return 1 we then store the result (0 or 1),
55:         #in s0 our local variable
56:     srl $a0, $a0, 1 #shift all the bits to the right and fill with zeroes
57:         #i.e 0101 -> 0010 -> 0001 -> 0000
58:
59:     jal countOnes #loop back to the function, and save return address
60:
61:     add $v0, $v0, $s0 #this is called recursively to add the 1's and 0's
62:
63: endFunction: #goes through and grabs the address of where to jump
64:     #and the number we are adding (will be 0 or 1).
65:     lw $ra, ($sp) #grab the first thing off the stack
66:         #(the return address to add statement)
67:     lw $s0, 4($sp) #grab the next thing off the stack (the local variable)
68:     addi $sp, $sp, 8 #adding takes space away from stack, we are adding 8 bytes,
69:         #so we are taking two words.
70:     jr $ra #jump to add statement
71:
72: error: #if the input is negative, output an error message, and exit
73:     li $v0, 4 #load string print service
74:     la $a0, Error #load string into a0
75:     syscall # print
76:
77:     li $v0, 10 #load system exit service
78:     syscall #exit program
```