CS/COE 0447 Fall 2009
Lab 4: Functions
Solution

```
#Problem 1
          .text
          li $a0, 0xFFFF0000  #LED memory starts at this address
          li $a1, 0x55555555  #LEDs to turn on
          jal setLED
          li $v0, 10
          syscall

setLED:        sw $a1, 0($a0)
          jr $ra



#Problem 2
          .data
ok:       .asciiz   "The values match!"
not_ok:        .asciiz   "The values don't match!"

          .text
          li $a0, 0xFFFF0000  #LED memory starts at this address
          li $a1, 0x55555555  #LEDs to turn on
          jal setLED          #Jump and link to setLED
          jal getLED          #Jump and link to getLED
          bne $a1, $v0, else  #Return values should be in $v0
          la $a0, ok          #Load ok string if equal
          j end
else:          la $a0, not_ok      #Load not_ok string if not equal
end:      li $v0, 4           #Print the string
          syscall
          li $v0, 10          #Exit
          syscall

setLED:        sw $a1, 0($a0)
          jr $ra

getLED:   lw $v0, 0($a0)
          jr $ra



#Problem 3
          .text
          li $a0, 0xFFFF0000  #LED memory starts at this address
          li $a1, 0x55555555  #LEDs to turn on
          jal setLED          #Jump and link to setLED
          jal notLED          #Jump and link to notLED
          li $v0, 10          #Exit
          syscall

setLED:   sw $a1, 0($a0)
          jr $ra

getLED:   lw $v0, 0($a0)
          jr $ra

notLED:   move $t4, $ra
          jal getLED
          nor $a1, $v0, $zero
          jal setLED
          jr $t4



#Problem 4
          .text
          li $a0, 0xFFFF0000  #LED memory starts at this address
          li $a1, 0x55555555  #LEDs to turn on
          li $a2, 5      #Number of words to store
          jal setLEDRange     #Jump and link to setLEDRange
          li $v0, 10          #Exit
          syscall

setLED:   sw $a1, 0($a0)
          jr $ra

getLED:   lw $v0, 0($a0)
          jr $ra

notLED:   move $t4, $ra
          jal getLED
          nor $a1, $v0, $zero
          jal setLED
          jr $t4

setLEDRange:   move $t4, $ra
loop:          beq $a2, $zero, end
          jal setLED
          addi $a0, $a0, 4
          addi $a2, $a2, -1
          j loop
end:      jr $t4
```