```
################################################################################
# <Ben Hunt>, CS 2318-253, Assignment 2 Part 1 Program B
################################################################################
# Checks if a user-entered int between 0 & 255 (inclusive) is less than 32
# & displays 0 if so, otherwise (32 or higher) displays 1
# Uses Bitwise operations to acheive this goal. AND OR NOR XOR ect.
# 14 Lines or less added after #read input integer
############################## data segment ###################################
        .data
legend1:    .asciiz "0: less than 32\n"
legend2:    .asciiz "1: 32 or higher\n"
inPrompt:   .asciiz "Enter an integer between 0 and 255: "
outLab:     .asciiz "It is "
############################## code segment ###################################

.text
        .globl main
main:
        li $v0, 4
        la $a0, legend1
        syscall                     # print legend line 1
        la $a0, legend2
        syscall                     # print legend line 2
        la $a0, inPrompt
        syscall                     # print input prompt

        li $v0, 5
        syscall                     # read input integer


        ###############################################################
        # Write NO MORE THAN 14 lines of code that involve using
        # ONLY the following:
        # - syscall
        # - syscall supporting instructions (e.g.: li to load $v0)
        # - instruction to make a saved copy
        # - bit manipulating instructions (ANDing, ORing, XORing,
        #   NORing and shifting - only whatever that are needed)
        # so that the program will work just like the sample runs
        # shown at the bottom.
        # You MUST test your completed program for AT LEAST the
        # test cases shown (and include the result in hardcopy).
        ###############################################################

        move $t0, $v0          # $t0 = $v0 save current number in temporary variable
        srl $t0, $t0, 0x02     # shift $t0 right by 2 places
        or $t0, $t0, $v0       # or $t0 with $v0 (temporary variable from earlier)
        move $t1, $t0          # save result in $t1, $t1 = $t0
        srl $t0, $t0 0x01      # Shift $t0 right by 1 place
        or $t0, $t0, $t1       # or $t0 with $t1
        andi $t0, $t0, 0x3F    # and $t0 with 0x00111111 clear 2 MSB's
        srl $t0, $t0, 0x05     # shift $t0 right by 5 places

        la $a0, outLab      # put address into $a0
        li $v0, 4           # print string (It is)
        syscall

        move $a0, $t0          # $a0 = $t0 (Preparing for output,
                               # $v0 requires $a0 as argument. so have to set $a0)
        li $v0, 1              # print integer
        syscall
```

```
 67
 68
 69
 70
 71
 72
 73
 74
 75
 76
 77
 78
 79                 ###############################################################
 80
 81                 li $v0, 10              # exit
 82                 syscall
 83
 84  ############################ sample test runs ################################
 85  # 0: less than 32
 86  # 1: 32 or higher
 87  # Enter an integer between 0 and 255: 0
 88  # It is 0
 89  # -- program is finished running --
 90  #
 91  #
 92  # Reset: reset completed.
 93  #
 94  # 0: less than 32
 95  # 1: 32 or higher
 96  # Enter an integer between 0 and 255: 31
 97  # It is 0
 98  # -- program is finished running --
 99  #
100  #
101  # Reset: reset completed.
102  #
103  # 0: less than 32
104  # 1: 32 or higher
105  # Enter an integer between 0 and 255: 32
106  # It is 1
107  # -- program is finished running --
108  #
109  #
110  # Reset: reset completed.
111  #
112  # 0: less than 32
113  # 1: 32 or higher
114  # Enter an integer between 0 and 255: 64
115  # It is 1
116  # -- program is finished running --
117  #
118  #
119  # Reset: reset completed.
120  #
121  # 0: less than 32
122  # 1: 32 or higher
123  # Enter an integer between 0 and 255: 128
124  # It is 1
125  # -- program is finished running --
126  #
127  #
128  # Reset: reset completed.
129  #
130  # 0: less than 32
131  # 1: 32 or higher
132  # Enter an integer between 0 and 255: 255
```

```
# It is 1
# -- program is finished running --
######################### end sample test runs ############################
```