```
CS 2734, Org II, Fall 2002, First Exam, Answers
========================================================================
1.  (a) 82 (base 10) = 1010010 (base 2)
       -46 (base 10 = ~1010010 + 1 = 1111 1111 10101101 + 1 =
                                     1111 1111 1010 1110
    (b) 1011 1111 1110 0110  (48 more 0's)  (binary) =
        1 01111111110  0110  (48 more 0's)  (broken into fields) =

        (-1)^Sign x (1 + Significand) x 2^(Exponent - Bias) =
        (-1)^1    x (1 + 0.375000000) x 2^(1022     - 1023) =
        (-1)      x 1.375               x 2^(-1)            =
           -(11/8)x(1/2) = -11/16 = -0.6875
========================================================================
2.
# Answer to Exam 1, Problem 2
        .globl  main
main:   add     $s7, $0, $ra    # save return address

        .data
  A:    .word  4, 9, 25, 49, 121, 169, 289  # squares of first 7 primes
        .text
################## Answer to Problem 2 ##########################
        la      $s1, A          # start address of A
        addi    $s2, $0, 0      # running sum
        addi    $s3, $0, 0      # array index of A
        addi    $s4, $0, 7      # constant 7

 Loop:  lw      $t1, 0($s1)     # $t1 = A[$s3]
        add     $s2, $s2, $t1   # $s2 = sum of A[] so far
        addi    $s1, $s1, 4     # $s1 += 4
        addi    $s3, $s3, 1     # $s3 += 1
        bne     $s3, $s4, Loop  # branch back to Loop until $s4 == 7

        addi    $v0, $0, 1      # print the sum
        add     $a0, $0, $s2
        syscall
################## End of Answer to Problem 2 ###################
        addi    $v0, $0, 4      # print a newline
        la      $a0, Newln
        syscall

        add     $ra, $0, $s7    # restore return address
        jr      $ra
        .data
Newln:  .asciiz "\n"
################## Output #######################################
# ten42% spim -file exam1_2.s
# 666
################## End of output ################################

Another anwser:
################## Second Answer to Problem 2 ###################
        la      $s1, A          # start address of A
        addi    $s2, $0, 0      # running sum
        addi    $s3, $0, 0      # array index of A
        addi    $s4, $0, 7      # constant 7

 Loop:  mul     $t0, $s3, 4     # $t0 = array index * 4
        add     $t2, $t0, $s1   # $t2 = start of A + offset
        lw      $t1, 0($t2)     # $t1 = contents at start of A + offset
        add     $s2, $s2, $t1   # $s2 = sum of A[] so far
        addi    $s3, $s3, 1     # $s3 += 1
        bne     $s3, $s4, Loop  # branch back to Loop until $s4 == 7

        addi    $v0, $0, 1      # print the sum
        add     $a0, $0, $s2
        syscall
################## End of Second Answer to Problem 2 ############
========================================================================
3.
    # CS 2734, Computer Organization II, Fall 2002
    # MIPS program giving answer to Exam1, question 3
            .globl main
    main:   addu    $s7, $zero, $ra
            .data
    A:      .word   4, 9, 25, 49, 121, 169, 289  # squares of first 7 primes
            .text
            la      $s0, A          # address of A
    ########## Start of answer to Question 3 (second part) ########
            add     $a0, $s0, $0    # addr of A in $a0            #
            addi    $a1, $0, 7      # 7 in $a1
            jal     Avals           # call Avals                  #
    ########## End of answer to Question 3 (second part) ##########

            add     $a0, $v0, $0    # print $v0 just returned (5)
            li      $v0, 1
            syscall
            jal     Newl
    ########## Finish main####################
            addu    $ra, $zero, $s7
            jr      $ra

    ########## Start of answer to Question 3 (first part) #########
    Avals:                                                      #
            addi    $sp, $sp, -4    # room for $ra on stack      #
            sw      $ra, 0($sp)     # save $ra                  #
            lw      $t4, 0($a0)     # $t4 = A[0]                #
            lw      $t5, 4($a0)     # $t5 = A[1]                #
            add     $v0, $t4, $t5   # $v0 = A[0] + A[1]         #
            lw      $ra, 0($sp)     # restore $ra from stack    #
            addi    $sp, $sp, 4     # restore stack
            jr      $ra             # return                    #
    ########## End of answer to Question 3 (first part) ###########

    ########## write newline ######################################
    Newl:   li      $v0, 4
            la      $a0, Newline
            syscall
            jr      $ra
            .data
    Newline:    .asciiz  "\n"
    ############### output ###############################
    # ten60% spim -file exam1_3.s
    # 13
    ######################################################
========================================================================
4. Use
    (a)  add    $s1, $s2,   $0    or  addi $s1, $s2, 0
    (b)  addi   $s3, $0,   200
    (c)  beq    $0,  $0,  Loop
========================================================================
5. (a) Assume A is 1 and B is 0.  So upper switch connected to A is open
   (doesn't conduct), while the upper switch connected to B is closed.
   Since these are connected in series and one is open, no voltage goes
   to C from the source.  In the lower switches, A grounds the right switch,
   while B does not ground the left switch, but the grounds are connected in
   parallel, so C is grounded.  Thus the value at C is 0.
   If C is 0, this makes the upper switch conduct, giving voltage to D,
   while the lower switch does not conduct, so D is 1.
   (b) This is a NOR gate (the output at C is 1 unless both A and B are 0)
   connected to an inverter, so the two form an OR gate.
```