```
# lab 3 exercise 3
        .data
x:  .word -5
y:  .word 12
z: .word 0
result: .word 0
newline: .asciiz "\n"

oboe:   .word 0x0B0E
base: .word 0
ball: .word 0
beef:  .word 0

mask1:  .word 0x0F00
mask2:  .word 0xFF00
mask3:  .word 0xF000
.text


    lw $t0, x  # x in $t0
    lw $t1, y  # y in $t1


    # slt $t2, $t0, $t1   # if x < y, $t2 = 1, else $t2 = 0 OLD CODE
    # sw $t2, z           # store answer in z OLD CODE

    # (2) $t0 is -2094967296 when x is 2.2 billion. The weird number occured because MIPS signed 32-bits
    # The range of signed 32-bit integers is -2,147,483,648 to 2,147,483,647.
    # 2.2 billion exceeds this limit so integer overflow occurs.


    # PART 3
    sltu $t2, $t0, $t1
    sw $t2, result

    # PART 4
    sll $t4, $t1, 5  # y*32
    sll $t5, $t1, 4  # y*16
    sll $t6, $t1, 3  # y*8
    sll $t7, $t1, 1  # y*2

    add $t4, $t4, $t5  # =y*32 + y*16
    add $t4, $t4, $t6  # =y+ y*8
    add $t4, $t4, $t7  # =y+ y*2
    add $t4, $t4, $t1  # =y+ y

    sw $t4, z          # Store z

    # PART 5 oboe to beef

    la $t0, oboe        # Load address
    lw $t1, 0($t0)      # Load value into $t1


    lw $t2, mask1       # Load mask 0x0F00
    and $t4, $t1, $t2   # Isolate (0x0B00)
    sll $t4, $t4, 4     # Shift left to get 0xB000

    or $t4, $t4, 0x0A5E

    # Store result of ba5e
    la $t0, base
    sw $t4, 0($t0)

    lw $t2, mask2       # Load mask 0xFF00
    and $t4, $t4, $t2   # Isolate (0xBA00)

    or $t4, $t4, 0x0011

    #store ba11
    la $t0, ball
    sw $t4, 0($t0)

    lw $t2, mask3       # Load mask 0xF000
    and $t4, $t4, $t2   # Isolate (0xB000)

    or  $t4, $t4, 0x0EEF   # Make last bytes 0x0EEF

    # Store result
    la $t0, beef
    sw $t4, 0($t0)
```