```
//
// Ezekiel Kim
// Lab 10
// Program to find largest number
// Ezekiel Kim
// /2023
//
        .global _start // Provide program starting address

        .data
                dbX:    .quad   0
                dbY:    .quad   0
                szX:    .skip   21
                szY:    .skip   21
                szMsg1: .asciz  "Enter x: "
                szMsg2: .asciz  "Enter y: "
                szGt:   .asciz  " > "
                szEq:   .asciz  " == "
                szVar1: .asciz  "x"
                szVar2: .asciz  "y"
                szCol:  .asciz  " : "
                chCr:   .byte   10
        .text

_start:
// Prompt user for X
        ldr     x0,     =szMsg1 // String to print
        bl      putstring
        ldr     x0,     =szX    // Receive x value
        bl      getstring
        ldr     x0,     =szX
        bl      ascint64
        ldr     x1,     =dbX    // Convert to int
        str     x0,     [x1]    // Store into dbX
// Prompt user for Y
        ldr     x0,     =szMsg2 // String to print
        bl      putstring
        ldr     x0,     =szY    // Receive y value
        bl      getstring
        ldr     x0,     =szY
        bl      ascint64
        ldr     x1,     =dbY    // Convert to int
        str     x0,     [x1]    // Store into dbX
// Compute
        ldr     x0,     =dbX    // Load dbX
        ldr     x0,     [x0]    // Dereference
        ldr     x1,     =dbY    // Load dbY
        ldr     x1,     [x1]    // Dereference
        cmp     x0,     x1
        b.eq    _equal
        b.lt    _lessthan
        b.gt    _greaterthan

_equal:
        ldr     x0,     =szVar1 // Print X
        bl      putstring
        ldr     x0,     =szEq   // Print equal sign
        bl      putstring
        ldr     x0,     =szVar2 // Print Y
        bl      putstring
        ldr     x0,     =szCol  // Print semicolon
        bl      putstring
        ldr     x0,     =szX    // Print X
        bl      putstring
        ldr     x0,     =szEq   // Print equal sign
        bl      putstring
        ldr     x0,     =szY    // Print Y
        bl      putstring
```

```
        ldr     x0,     =chCr   // Print carriage return
        bl      putch
        b.al    _end

_lessthan:
        ldr     x0,     =szVar2 // Print Y
        bl      putstring
        ldr     x0,     =szGt   // Print gt sign
        bl      putstring
        ldr     x0,     =szVar1 // Print X
        bl      putstring
        ldr     x0,     =szCol  // Print semicolon
        bl      putstring
        ldr     x0,     =szY    // Print Y
        bl      putstring
        ldr     x0,     =szGt   // Print gt sign
        bl      putstring
        ldr     x0,     =szX    // Print X
        bl      putstring
        ldr     x0,     =chCr   // Print carriage return
        bl      putch
        b.al    _end

_greaterthan:
        ldr     x0,     =szVar1 // Print X
        bl      putstring
        ldr     x0,     =szGt   // Print gt sign
        bl      putstring
        ldr     x0,     =szVar2 // Print Y
        bl      putstring
        ldr     x0,     =szCol  // Print semicolon
        bl      putstring
        ldr     x0,     =szX    // Print X
        bl      putstring
        ldr     x0,     =szGt   // Print gt sign
        bl      putstring
        ldr     x0,     =szY    // Print Y
        bl      putstring
        ldr     x0,     =chCr   // Print carriage return
        bl      putch
        b.al    _end


// Setup the parameters to exit the program and then call Linux to do it.
_end:
        mov     x0,     #0      // Sets return code to 0
        mov     x8,     #93     // Service command code 93 terminates
        svc     0       // Call linux to terminate the program
        .end
```