```
//
// Ezekiel Kim
// Exam 1
// Program to perform some calculations
// Ezekiel Kim
// /2023
//
        .global _start // Provide program starting address

        .data
                iX:     .quad   0
                iY:     .quad   0
                szX:    .skip   21
                szY:    .skip   21
                szPmp1: .asciz  "Enter x: "
                szPmp2: .asciz  "Enter y: "
                szMsg1: .asciz  "3 * ("
                szMsg2: .asciz  " + 2 * "
                szMsg3: .asciz  ") = "
                iRes:   .quad   0
                szRes:  .skip   21
                chCr:   .byte   10
        .text

_start:
// First take in values and convert them to integers
        ldr     x0,     =szPmp1 // Set address as szPmp1
        bl      putstring       // Print this string
        ldr     x0,     =szX    // Set address as szX
        bl      getstring       // Get the string
        ldr     x0,     =szX    // Set string to x0
        bl      ascint64        // Convert to integer
        ldr     x1,     =iX     // Get address of int X
        str     x0,     [x1]    // Store value into int x
// Take next value (y)
        ldr     x0,     =szPmp2 // Set address as szPmp2
        bl      putstring       // Print the string
        ldr     x0,     =szY    // Set address as szY
        bl      getstring       // Get the string
        ldr     x0,     =szY    // Set string to x0
        bl      ascint64        // Convert to integer
        ldr     x1,     =iY     // Get address of int y
        str     x0,     [x1]    // Store value into int y

// Make calculations and store in result
        ldr     x1,     =iX     // Get iX address
        ldr     x1,     [x1]    // Dereference
        ldr     x2,     =iY     // Get iY address
        ldr     x2,     [x2]    // Dereference
        add     x3,     x2,     x2      // y * 2
        add     x4,     x3,     x1      // x + 2y
        add     x5,     x4,     x4      // 2(x+2y)
        add     x6,     x5,     x4      // 3(x+2y)
        ldr     x4,     =iRes   // Load result address
        str     x3,     [x4]    // Store x3 into result

// Convert result to ascii
        ldr     x0,     =iRes   // Load iRes
        ldr     x0,     [x0]    // Dereference
        ldr     x1,     =szRes  // Load string of result
        bl      int64asc        // Convert int to string

// Print result
        ldr     x0,     =chCr   // load carriage return
        bl      putch           // Print CR
        ldr     x0,     =szMsg1 // Load first message
        bl      putstring       // Print the string
        ldr     x0,     =szX    // Load x string
```

```
        bl      putstring       // Print the string
        ldr     x0,     =szMsg2 // Load second message
        bl      putstring       // Print the string
        ldr     x0,     =szX    // Load y string
        bl      putstring       // Print the string
        ldr     x0,     =szMsg3 // Load third message
        bl      putstring       // Print the string
        ldr     x0,     =szRes  // Load result strin
        bl      putstring       // Print the string
        ldr     x0,     =chCr   // Load carriage return
        bl      putch           // Print CR

// Setup the parameters to exit the program and then call Linux to do it.
        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
```