```
lab7.s
             Mon Feb 13 18:07:02 2023
// Ezekiel Kim
// Lab 7
// Program to Load Memory from Data Chunk
// Ezekiel Kim
// /2023
//
         .qlobal _start // Provide program starting address
         .data
                  bA:
                          .byte 155
                  bFlag: .byte 1
                  chInit: .byte 'i'
                  u16Hi: .hword 88
                  u16Lo: .hword 45
                  wAlt: .word 16, -1, -2
                  szMsg1: .asciz "And Sally and I"
                  dbBig: .quad 9223372036854775807
                  szA: .skip 21
szFlag: .skip 21
                  szInit: .skip 21
                  szHi: .skip 21
szLo: .skip 21
                                  21
                 szAlt: .skip 21
szBig: .skip 21
szVar0: .asciz "bA"
                  szVar1: .asciz "bFlag"
                  szVar2: .asciz "chInit"
                  szVar3: .asciz "u16Hi"
                  szVar4: .asciz "u16Lo"
                  szVar5: .asciz "wAlt[0]"
                  szVar6: .asciz "wAlt[1]"
                  szVar7: .asciz "wAlt[2]"
                  szVar8: .asciz "szMsq1"
                  szVar9: .asciz "dbBiq"
                  szEq: .asciz " = "
                  chSq: .byte 39
                  chDq: .byte 34
                  chCr: .byte 10
        .text
_start:
// Load variable label
         ldr x0, =szVar0 // load szVar to print
                putstring // Print it to console
         bl
               x0, =szEq // load equal sign putstring // Print it to console
         ldr
        bl
// Print value
              x1, =bA // Load dbA

w0, [x1] // Load byte to x0

x1, =szA // Load string result

int64asc // Convert string

x0, =szA // Load string to x0

putstring // Print it to console

x0, =chCr // Load carriage return
         ldr
         ldrb
         ldr
        bl
         ldr
        bl
         ldr
                                   // Print carriage return
        h1
                 putch
// Load variable label
         ldr x0, =szVar1 // load szVar to print
                 putstring // Print it to console
         b1
                x0, =szEq // load equal sign
         ldr
                                  // Print it to console
        bl
                 putstring
// Print value
                         =bFlag // Load bFlag
         ldr
                x1,
                 w0, [x1] // Load byte to x0 x1, =szFlag // Load string result
         ldrb w0,
```

int64asc // Convert string

x0, =szFlag // Load string to x0

ldr bl

ldr

```
lab7.s
                       Mon Feb 13 18:07:02 2023
              bl putstring // Print it to console
ldr x0, =chCr // Load carriage return
bl putch // Print carriage return
// Load variable label
              ldr x0, =szVar2 // load szVar to print
                           putstring // Print it to console x0, =szEq // load equal sign
              bl
              ldr
                          putstring // Print it to console
              bl
// Print value
             ldr x0, =chSq // Load single quote
bl putch // Print it to console
ldr x0, =chInit // Load chinit
bl putch // Print it to console
ldr x0, =chSq // Load single quote
ldr x0, =chSq // Load single quote
bl putch // Print it to console
ldr x0, =chCr // Load carriage return
bl putch // Print carriage return
                                                         // Print carriage return
// Load variable label
               ldr x0, =szVar3 // load szVar to print
                       putstring // Print it to console x0, =szEq // load equal sign putstring // Print it to console
              bl
               ldr
              bl
// Print value
              ldr x1, =u16Hi // Load u16Hi
ldrh w0, [x1] // Load halfword to x0
ldr x1, =szHi // Load string result
bl int64asc // Convert string
ldr x0, =szHi // Load string to x0
bl putstring // Print it to console
ldr x0, =chCr // Load carriage return
                                                         // Print carriage return
              bl
                           putch
// Load variable label
              ldr x0, =szVar4 // load szVar to print
              bl
                           putstring // Print it to console
              ldr x0, =szEq // load equal sign
bl putstring // Print it to console
              t value

ldr x1, =u16Lo // Load u16Lo

ldrh w0, [x1] // Load halfword to x0

ldr x1, =szLo // Load string result

bl int64asc // Convert string

ldr x0, =szLo // Load string to x0

bl putstring // Print it to console

ldr x0, =chCr // Load carriage return

bl putch // Print carriage return
// Print value
// Load variable label
              1dr x0, =szVar5 // load szVar to print
                         putstring // Print it to console
x0, =szEq // load equal sign
putstring // Print it to console
              h1
              ldr
              bl
// Print value
              ldr x1, =wAlt // Load word wAlt[0]
ldrsw x0, [x1] // Load word to x0
ldr x1, =szAlt // Load string result
bl int64asc // Convert string
ldr x0, =szAlt // Load string to x0
bl putstring // Print it to console
ldr x0, =chCr // Load carriage return
bl putch // Print carriage return
                                                         // Print carriage return
// Load variable label
               ldr x0, =szVar6 // load szVar to print
              bl
                             putstring // Print it to console
                      x0, =szEq // load equal sign
```

ldr

```
putstring // Print it to console
        bl
// Print value
         ldr x1,
                            =wAlt // Load word wAlt[1]
                 x0, [x1, #4] // Load word to x0
x1, =szAlt // Load string result
int64asc // Convert string
x0, =szAlt // Load string to x0
         ldrsw x0,
         ldr
         bl
         ldr
         bl putstring // Print it to console
ldr x0, =chCr // Load carriage return
bl putch // Print carriage return
                              // Print carriage return
// Load variable label
         1dr x0, =szVar7 // load szVar to print
                  putstring // Print it to console
         bl
                  x0, =szEq // load equal sign
         ldr
                                      // Print it to console
         bl
                  putstring
// Print value
         ldr x1, =wAlt // Load word wAlt[2]
ldrsw x0, [x1, #8] // Load word to x0
ldr x1, =szAlt // Load string result
              int64asc // Convert string
x0, =szAlt // Load string to x0
putstring // Print it to console
x0, =chCr // Load carriage return
putch // Print carriage return
         bl
          ldr
         bl
         ldr
         bl
// Load variable label
         ldr x0, =szVar8 // load szVar to print
                   putstring // Print it to console
         bl
         ldr x0, =szEq // load equal sign bl putstring // Print it to console
// Print value
                 x0, =chDq // Load double quote
         ldr
         bl putch // Print doublequote
ldr x0, =szMsgl // Load string to x0
         bl
                 putstring // Print it to console
                  x0, =chDq // Load double quote
         ldr
                 putch // Print double quote
         bl
         ldr x0, =chCr // Load carriage return
bl putch // Print carriage return
                               // Print carriage return
// Load variable label ldr x0, =szVar9 // load szVar to print
                  putstring // Print it to console
         bl
                x0, =szEq // load equal sign putstring // Print it to console
         ldr
         bl
// Print value
                x1,
                            =dbBig // Load word wAlt[2]
          ldr
                 x0, [x1] // Load word to x0
x1, =szBig // Load string result
int64asc // Convert string
x0, =szBig // Load string to x0
          ldr
         ldr
         bl
         ldr
                 putstring // Print it to console
x0, =chCr // Load carriage return
putch // Print carriage return
         bl
         ldr
                                      // Print carriage return
         bl
// Setup the parameters to exit the program and then call Linux to do it.
                 x0, #0 // Sets return code to 0
x8, #93 // Service command code 93 terminates
         mov
                  0
                          // Call linux to terminate the program
         SVC
          .end
```