CS 271 Computer Architecture and Assembly Language

Self-Check for Lecture#15

Solutions are posted

Given the following partial data segment, which starts at address 0x0200:

```
.data
list       DWORD   1, 2, 6, 24, 120, 720, 5040, 40320
x          DWORD   LENGTHOF list
y          DWORD   SIZEOF list
```

Show addresses in 4-digit hexadecimal.
Show contents in decimal.

1. x contains

2. y contains

3. The address of x is ________________ (hex)

4. Given this code fragment:
   
   mov    esi, OFFSET list
   mov    eax, [esi+5+TYPE list]

   eax contains ________________

5. Given this code fragment:
   
   mov    esi, OFFSET list
   mov    ebx, y
   sub    ebx, TYPE y
   add    esi, ebx

   [esi] contains ________________

6. Given this code fragment:
   
   mov    esi, OFFSET list
   mov    ebx, y
   sub    ebx, TYPE y
   add    esi, ebx
   mov    al, BYTE PTR [esi+1]

   The AL register contains ________________
Given the following partial data segment, which starts at address 0x0200:

```assembly
.data
matrix DWORD 20 DUP (5 DUP(?))
x DWORD LENGTHOF matrix
y DWORD SIZEOF matrix
pal BYTE "Hello world.", 0
len DWORD LENGTHOF pal
```

Show addresses in 4-digit hexadecimal.
Show contents in decimal.

1. `x` contains _______________

2. `y` contains _______________

3. In high-level language notation, the 3rd element of the 9th row is referenced as `matrix[8][2]`. The address of `matrix[8][2]` is _______________

4. Given this code fragment:
   ```assembly
   mov esi, OFFSET pal
   mov ecx, len
   sub ecx, 2
   std
   one:
   lodsb
   call WriteChar
   loop one
   mov ecx, len
   sub ecx, 2
   std
   two:
   lodsb
   call WriteChar
   loop two
   WriteChar displays the character in the AL register.
   What is displayed?
   ```
All keyboard input is character. The keyboard digits, '0', '1', '2', ..., '9' are ASCII codes 48, 49, 50, ..., 57. When a user enters a numeric value, it comes into memory as a string of digits.

This exercise is intended to show why it's necessary to convert to numeric representation before using the string of digits. An ASCII table is provided below.

To add two digit strings, you have to be sure that the strings both have the same length by zero-filling the shorter string on the left. Carry digits would be an additional nightmare. This example ignores both of those problems by adding two strings of equal length with no carry digits.

What is the (string) result of adding the following digit string, digit by digit?

```
a BYTE "2458", 0
b BYTE "6301", 0
```