

## 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 :

```
.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 3<sup>rd</sup> element of the 9<sup>th</sup> row is referenced as *matrix*[8][2].

The address of *matrix*[8][2] is \_\_\_\_\_

4. Given this code fragment:

```
    mov  esi, OFFSET pal
    mov  ecx, len
    sub  ecx, 2
    cld
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

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040	&#32;	Space	64	40	100	&#64;	@	96	60	140	&#96;	`
1	1	001	SOH (start of heading)	33	21	041	&#33;	!	65	41	101	&#65;	A	97	61	141	&#97;	a
2	2	002	STX (start of text)	34	22	042	&#34;	"	66	42	102	&#66;	B	98	62	142	&#98;	b
3	3	003	ETX (end of text)	35	23	043	&#35;	#	67	43	103	&#67;	C	99	63	143	&#99;	c
4	4	004	EOT (end of transmission)	36	24	044	&#36;	\$	68	44	104	&#68;	D	100	64	144	&#100;	d
5	5	005	ENQ (enquiry)	37	25	045	&#37;	%	69	45	105	&#69;	E	101	65	145	&#101;	e
6	6	006	ACK (acknowledge)	38	26	046	&#38;	&	70	46	106	&#70;	F	102	66	146	&#102;	f
7	7	007	BEL (bell)	39	27	047	&#39;	'	71	47	107	&#71;	G	103	67	147	&#103;	g
8	8	010	BS (backspace)	40	28	050	&#40;	(	72	48	110	&#72;	H	104	68	150	&#104;	h
9	9	011	TAB (horizontal tab)	41	29	051	&#41;	)	73	49	111	&#73;	I	105	69	151	&#105;	i
10	A	012	LF (NL line feed, new line)	42	2A	052	&#42;	*	74	4A	112	&#74;	J	106	6A	152	&#106;	j
11	B	013	VT (vertical tab)	43	2B	053	&#43;	+	75	4B	113	&#75;	K	107	6B	153	&#107;	k
12	C	014	FF (NP form feed, new page)	44	2C	054	&#44;	,	76	4C	114	&#76;	L	108	6C	154	&#108;	l
13	D	015	CR (carriage return)	45	2D	055	&#45;	-	77	4D	115	&#77;	M	109	6D	155	&#109;	m
14	E	016	SO (shift out)	46	2E	056	&#46;	.	78	4E	116	&#78;	N	110	6E	156	&#110;	n
15	F	017	SI (shift in)	47	2F	057	&#47;	/	79	4F	117	&#79;	O	111	6F	157	&#111;	o
16	10	020	DLE (data link escape)	48	30	060	&#48;	0	80	50	120	&#80;	P	112	70	160	&#112;	p
17	11	021	DC1 (device control 1)	49	31	061	&#49;	1	81	51	121	&#81;	Q	113	71	161	&#113;	q
18	12	022	DC2 (device control 2)	50	32	062	&#50;	2	82	52	122	&#82;	R	114	72	162	&#114;	r
19	13	023	DC3 (device control 3)	51	33	063	&#51;	3	83	53	123	&#83;	S	115	73	163	&#115;	s
20	14	024	DC4 (device control 4)	52	34	064	&#52;	4	84	54	124	&#84;	T	116	74	164	&#116;	t
21	15	025	NAK (negative acknowledge)	53	35	065	&#53;	5	85	55	125	&#85;	U	117	75	165	&#117;	u
22	16	026	SYN (synchronous idle)	54	36	066	&#54;	6	86	56	126	&#86;	V	118	76	166	&#118;	v
23	17	027	ETB (end of trans. block)	55	37	067	&#55;	7	87	57	127	&#87;	W	119	77	167	&#119;	w
24	18	030	CAN (cancel)	56	38	070	&#56;	8	88	58	130	&#88;	X	120	78	170	&#120;	x
25	19	031	EM (end of medium)	57	39	071	&#57;	9	89	59	131	&#89;	Y	121	79	171	&#121;	y
26	1A	032	SUB (substitute)	58	3A	072	&#58;	:	90	5A	132	&#90;	Z	122	7A	172	&#122;	z
27	1B	033	ESC (escape)	59	3B	073	&#59;	;	91	5B	133	&#91;	[	123	7B	173	&#123;	{
28	1C	034	FS (file separator)	60	3C	074	&#60;	<	92	5C	134	&#92;	\	124	7C	174	&#124;	
29	1D	035	GS (group separator)	61	3D	075	&#61;	=	93	5D	135	&#93;	]	125	7D	175	&#125;	}
30	1E	036	RS (record separator)	62	3E	076	&#62;	>	94	5E	136	&#94;	^	126	7E	176	&#126;	~
31	1F	037	US (unit separator)	63	3F	077	&#63;	?	95	5F	137	&#95;	_	127	7F	177	&#127;	DEL

Source: [www.LookupTables.com](http://www.LookupTables.com)