L21-5625
Rohan Jawed

COAL Assignment

Q1)

a) mov [02], [22]

correct:

```
mov ax, [22]
mov [22], ax
```

b)

```
mov [word], al
mov al, 22
```

c)

```
mov bl, al
```

d)

```
mov bx, si
mov ax, [bx + di + 100]
```

Q2)

```
[org ox0100]
jmp start
array : db -3, 5, 10, 4, 6, -7, 1
size : db 7
max : db 0
start:
    mov al, 0
    mov bx, 0
l1:
    mov al, [array+bx]
    cmp [max], al
L2:
    add bx, 1
    cmp bx, 7
    jne L1
terminate:
    mov ax, 0x4c00
    int 0x21
```

Q3)

a) IDDD : 0436

IDDDO
+00436
1E206

b) 74FO : 2123

74F00
+02123
77023

d) 0000 : 6727

00000
+06727
06727

e) FFFF : 4336

FFFFO
+04336
10436

f) 1080 : 0100

10800
+00100
10900

g) AB01 : FFFF

AB010
+0FFFF
BB00F

Q4)

a) 1000

first : 0x10000

last : 0x1FFFF

0000 → first address

```
 10000
+00000
-------
 10000
```

FFFF → last address

```
 10000
 0FFFF
-------
 1FFFF
```

b) 0FFF

0000 → first

```
 0FFF0
+00000
-------
 0FFF0
```

first: 0x0FFF0

FFFF → last

```
 0FFF0
+0FFFF
-------
 1FFEF
```

last: 0x1FFEF

c) 1002

0000 → first

```
 10020
+00000
-------
 10020
```

first: 0x10020

FFFF → last

```
  10020
+ 0FFFF
  2001F
```

last: 0x 2001F

d) 0001

0000 → first

```
  00000
+ 00010
  00010
```

first: 0x 00010

FFFF → last

```
  11110
+ 0FFFF
  1000F
```

last: 0x 1000F

e) E000

0000 → first

```
  E0000
+ 00000
  E0000
```

0x E0000

FFFF → last

```
  E0000
+ 0FFFF
  EFFFF
```

last: 0xEFFFF

Q5)

a) 0x0100 + 0x000C
= 0x010C

b) 0x0100 + 0x1001
= 0x01101

c) 0x1001 + 0x0100
= 0x1101

d) 0x0100 + 0x0110
= 0x0200

Q6)

a) invalid
Subtraction between registers is not allowed

b) invalid
Subtraction between registers is not allowed

c) bx + 10
= 0x0110

d) bx - 10
= 0x00F0

f) bx + di
= 0x0101

e) bx + sp
= 0x100FF

Q7) a) CS : IP = OFF29

IP. OFF29 - OFE20

= 0109

b) Ax = 5

Q8)

i) A1[1700]

as we are moving data into registers ax

offset = 0017

DS = OFE2

Physical Address : OFE20

0017

→ OFE37

ii) A1[1208]

offset = 0B12

DS = OFE2

Physical Address

OFE20

+ 00B12

→ 10932

Q1)

```
[org 0x0100]
    jmp start
multiplicand : dq 63459263
multiplier : dq 86248974
result : dq 0

start :
    mov cl, 64
    mov bx, 1

checkbit :
    check bx, [multiplier]
    jz skip

    mov ax, [multiplicand]
    add [result], ax
    mov ax, [multiplicand+2]
    adc [result+2], ax

skip:
    shl word [multiplicand], 1
    rcl word [multiplicand+2], 1
    rcl word [multiplicand+4], 1
    rcl word [multiplicand+6], 1
    shl bx, 1
    dec cl
    jnz checkbit
    mov ax, 0x4C00
    int 0x21
```

Q10)

a) 68 bytes

b)

before ort:	3	10	2	0	7	5	8	
After ort:	8	0	2	10	7	5	3	