WAQAR AHMED

20P-0750

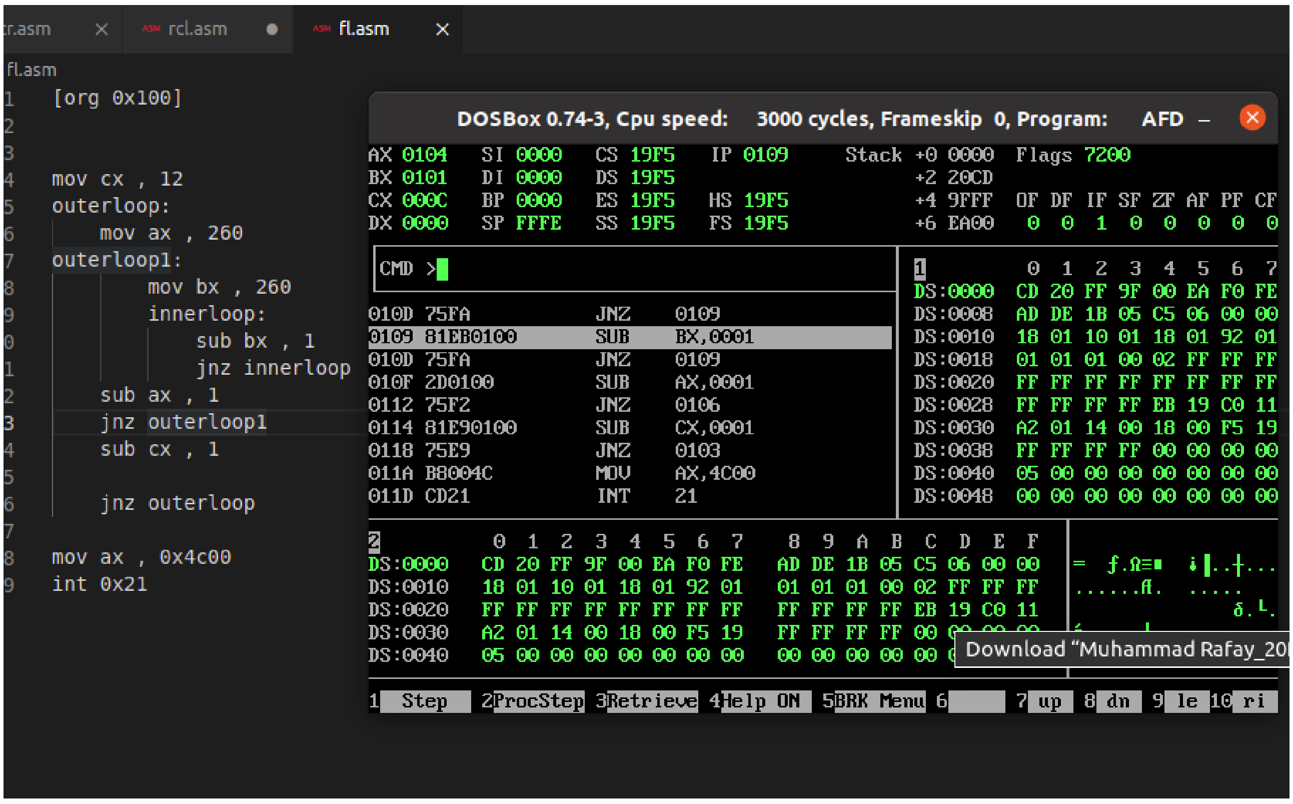
Report – 5

Flags:

The Flags Register :

The Flags Register contain four flags of particular interest: – Zero flag (set when the result of an operation is zero). – Carry flag (set when the result of unsigned arithmetic is too large for the destination operand or when subtraction requires a borrow). – Sign flag (set when the high bit of the destination operand is set indicating a negative result). – Overflow flag (set when signed arithmetic generates a result which ifs out of range).

```
[org 0x100]


mov cx , 12
outerloop:
        mov ax , 260
outerloop1:
                mov bx , 260
                innerloop:
                        sub bx , 1
                        jnz innerloop
        sub ax , 1
        jnz outerloop1
        sub cx , 1

        jnz outerloop

mov ax , 0x4c00
int 0x21
```



```
[org 0x100]


mov cx , 12
outerloop:
    mov ax , 260
outerloop1:
        mov bx , 260
        innerloop:
            sub bx , 1
            jnz innerloop
    sub ax , 1
    jnz outerloop1
    sub cx , 1

    jnz outerloop

mov ax , 0x4c00
int 0x21
```

```
DOSBox 0.74-3, Cpu speed:    3000 cycles, Frameskip  0, Program:    AFD

AX 0104   SI 0000   CS 19F5   IP 0109      Stack +0 0000  Flags 7200
BX 0101   DI 0000   DS 19F5                      +2 20CD
CX 000C   BP 0000   ES 19F5   HS 19F5            +4 9FFF  OF DF IF SF ZF AF PF CF
DX 0000   SP FFFE   SS 19F5   FS 19F5            +6 EA00   0  0  1  0  0  0  0  0

CMD >                                     1          0  1  2  3  4  5  6  7
                                          DS:0000   CD 20 FF 9F 00 EA F0 FE
010D 75FA            JNZ    0109          DS:0008   AD DE 1B 05 C5 06 00 00
0109 81EB0100        SUB    BX,0001       DS:0010   18 01 10 01 18 01 92 01
010D 75FA            JNZ    0109          DS:0018   01 01 01 00 02 FF FF FF
010F 2D0100          SUB    AX,0001       DS:0020   FF FF FF FF FF FF FF FF
0112 75F2            JNZ    0106          DS:0028   FF FF FF FF EB 19 C0 11
0114 81E90100        SUB    CX,0001       DS:0030   A2 01 14 00 18 00 F5 19
0118 75E9            JNZ    0103          DS:0038   FF FF FF FF 00 00 00 00
011A B8004C          MOV    AX,4C00       DS:0040   05 00 00 00 00 00 00 00
011D CD21            INT    21            DS:0048   00 00 00 00 00 00 00 00

2          0  1  2  3  4  5  6  7    8  9  A  B  C  D  E  F
DS:0000   CD 20 FF 9F 00 EA F0 FE   AD DE 1B 05 C5 06 00 00   =  ƒ.Ω≡■  ↓▌..┼...
DS:0010   18 01 10 01 18 01 92 01   01 01 01 00 02 FF FF FF   ......Æ.  .....
DS:0020   FF FF FF FF FF FF FF FF   FF FF FF FF EB 19 C0 11             δ.└.
DS:0030   A2 01 14 00 18 00 F5 19   FF FF FF FF 00
DS:0040   05 00 00 00 00 00 00 00   00 00 00 00 00

1 Step  2ProcStep  3Retrieve  4Help ON  5BRK Menu  6  7 up  8 dn  9 le  10 ri
```

In this we move counter loop 12 times in cx register.we move 260 in ax and bx register .
When the bx register is subtracted by one the zero flag will not set , variable which means in inner loop variable will run the loop will be run 260 times the the counter 12 will be subtracted one by one after this the register value of ax will also be decrease one by one.

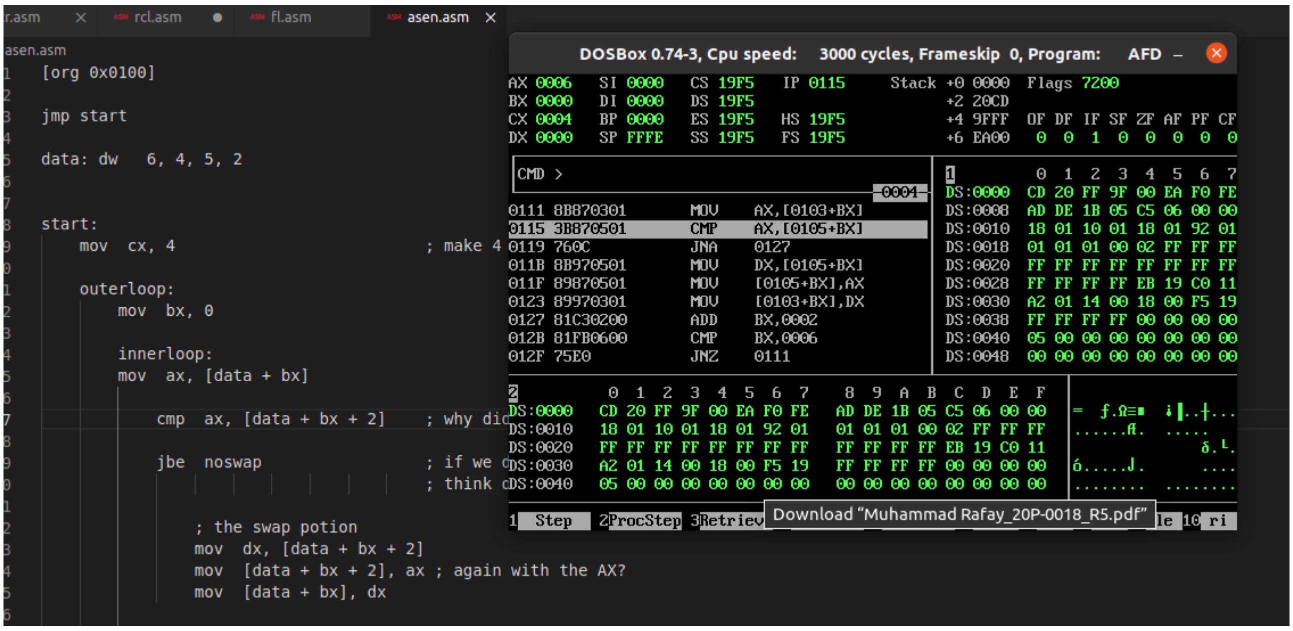
CMP Instruction :

•The CMP instruction sets the flags as if it had performed subtraction on the operand.
• Neither operand is changed.
• The CMP instruction takes the forms: CMP reg, reg CMP mem, reg CMP reg, mem CMP mem, immed CMP reg, immed.

Ascending sorting :

```
[org 0x0100]

jmp start

data: dw   6, 4, 5, 2



start:
   mov  cx, 4                          ; make 4 passes, has to be outside the loop!

   outerloop:
      mov  bx, 0

      innerloop:
      mov  ax, [data + bx]

         cmp  ax, [data + bx + 2]    ; why did we move the value to AX?

         jbe  noswap                  ; if we don't have to swap, we just jump over the swap thing
                                      ; think of this as the "if"

            ; the swap potion
            mov  dx, [data + bx + 2]
            mov  [data + bx + 2], ax ; again with the AX?
            mov  [data + bx], dx

         noswap:
         add  bx, 2
         cmp  bx, 6
```

```
    jne  innerloop


  ; check outer loop termination
  sub cx, 1
  jnz outerloop


; exit system call
mov  ax, 0x4c00
int  0x21
```



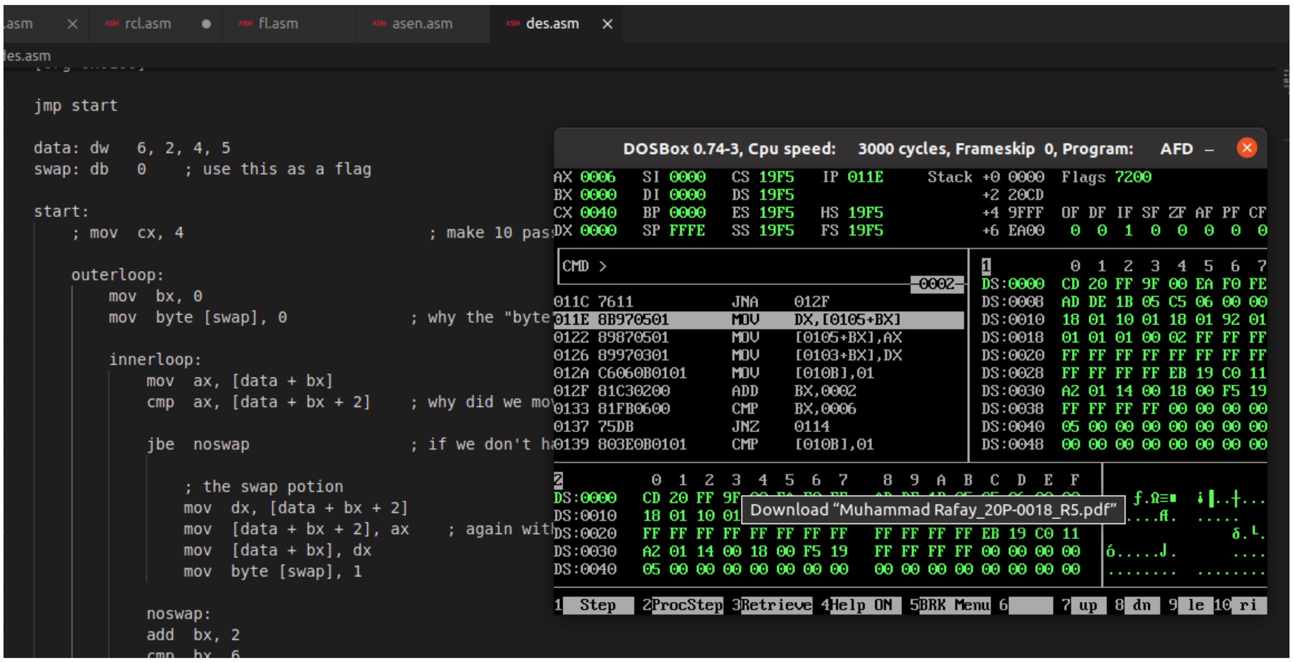
r.asm
rcl.asm
fl.asm
asen.asm
asen.asm
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD
Download “Muhammad Rafay_20P-0018_R5.pdf”

```
[org 0x0100]

jmp start

data: dw   6, 4, 5, 2


start:
    mov  cx, 4                          ; make 4

    outerloop:
        mov  bx, 0

        innerloop:
        mov  ax, [data + bx]

            cmp  ax, [data + bx + 2]    ; why did

            jbe  noswap                 ; if we d
                                        ; think d

                ; the swap potion
                mov  dx, [data + bx + 2]
                mov  [data + bx + 2], ax ; again with the AX?
                mov  [data + bx], dx
```

```
AX 0006   SI 0000   CS 19F5   IP 0115     Stack +0 0000  Flags 7200
BX 0000   DI 0000   DS 19F5                     +2 20CD
CX 0004   BP 0000   ES 19F5   HS 19F5           +4 9FFF  OF DF IF SF ZF AF PF CF
DX 0000   SP FFFE   SS 19F5   FS 19F5           +6 EA00   0  0  1  0  0  0  0  0

CMD >                                  0004   1          0  1  2  3  4  5  6  7
                                              DS:0000   CD 20 FF 9F 00 EA F0 FE
0111 8B870301       MOV    AX,[0103+BX]       DS:0008   AD DE 1B 05 C5 06 00 00
0115 3B870501       CMP    AX,[0105+BX]       DS:0010   18 01 10 01 18 01 92 01
0119 760C           JNA    0127               DS:0018   01 01 01 00 02 FF FF FF
011B 8B970501       MOV    DX,[0105+BX]       DS:0020   FF FF FF FF FF FF FF FF
011F 89870501       MOV    [0105+BX],AX       DS:0028   FF FF FF FF EB 19 C0 11
0123 89970301       MOV    [0103+BX],DX       DS:0030   A2 01 14 00 18 00 F5 19
0127 81C30200       ADD    BX,0002            DS:0038   FF FF FF FF 00 00 00 00
012B 81FB0600       CMP    BX,0006            DS:0040   05 00 00 00 00 00 00 00
012F 75E0           JNZ    0111               DS:0048   00 00 00 00 00 00 00 00

2           0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F
DS:0000    CD 20 FF 9F 00 EA F0 FE  AD DE 1B 05 C5 06 00 00   =  ƒ.Ω≡∎  ¡▌..┼...
DS:0010    18 01 10 01 18 01 92 01  01 01 01 00 02 FF FF FF   ......Æ.  .....
DS:0020    FF FF FF FF FF FF FF FF  FF FF FF FF EB 19 C0 11             δ.└.
DS:0030    A2 01 14 00 18 00 F5 19  FF FF FF FF 00 00 00 00   ó.....J.      ....
DS:0040    05 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........  ........

1 Step  2ProcStep  3Retriev  le 10 ri
```

Descending sorting:

```
[org 0x0100]

jmp start

data: dw   6, 2, 4, 5
swap: db   0    ; use this as a flag

start:
   ; mov  cx, 4                         ; make 10 passes, has to be outside the loop!

   outerloop:
      mov  bx, 0
      mov  byte [swap], 0            ; why the "byte"?

      innerloop:
         mov  ax, [data + bx]
         cmp  ax, [data + bx + 2]    ; why did we move the value to AX?

         jbe  noswap                ; if we don't have to swap, we just jump over the swap
thing

            ; the swap potion
            mov  dx, [data + bx + 2]
            mov  [data + bx + 2], ax    ; again with the AX?
            mov  [data + bx], dx
            mov  byte [swap], 1

         noswap:
         add  bx, 2
         cmp  bx, 6
         jne  innerloop

      ; if we didn't swap even once, we should be done
      cmp  byte [swap], 1     ; don't need to load this in register?
      je   outerloop

      ; check outer loop termination
      ; sub cx, 1
      ; jnz outerloop
```



```
jmp start

data: dw   6, 2, 4, 5
swap: db   0    ; use this as a flag

start:
    ; mov  cx, 4                        ; make 10 pas

    outerloop:
        mov  bx, 0
        mov  byte [swap], 0             ; why the "byte

        innerloop:
            mov  ax, [data + bx]
            cmp  ax, [data + bx + 2]    ; why did we mo

            jbe  noswap                 ; if we don't h

                ; the swap potion
                mov  dx, [data + bx + 2]
                mov  [data + bx + 2], ax    ; again wit
                mov  [data + bx], dx
                mov  byte [swap], 1

            noswap:
            add  bx, 2
```

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD

```
AX 0006   SI 0000   CS 19F5   IP 011E     Stack +0 0000  Flags 7200
BX 0000   DI 0000   DS 19F5                     +2 20CD
CX 0040   BP 0000   ES 19F5   HS 19F5           +4 9FFF  OF DF IF SF ZF AF PF CF
DX 0000   SP FFFE   SS 19F5   FS 19F5           +6 EA00   0  0  1  0  0  0  0  0

CMD >                                     1         0  1  2  3  4  5  6  7
                               -0002-     DS:0000  CD 20 FF 9F 00 EA F0 FE
011C 7611        JNA    012F              DS:0008  AD DE 1B 05 C5 06 00 00
011E 8B970501    MOV    DX,[0105+BX]      DS:0010  18 01 10 01 18 01 92 01
0122 89870501    MOV    [0105+BX],AX      DS:0018  01 01 01 00 02 FF FF FF
0126 89970301    MOV    [0103+BX],DX      DS:0020  FF FF FF FF FF FF FF FF
012A C6060B0101  MOV    [010B],01         DS:0028  FF FF FF FF EB 19 C0 11
012F 81C30200    ADD    BX,0002           DS:0030  A2 01 14 00 18 00 F5 19
0133 81FB0600    CMP    BX,0006           DS:0038  FF FF FF FF 00 00 00 00
0137 75DB        JNZ    0114              DS:0040  05 00 00 00 00 00 00 00
0139 803E0B0101  CMP    [010B],01         DS:0048  00 00 00 00 00 00 00 00

2          0  1  2  3  4  5  6  7    8  9  A  B  C  D  E  F
DS:0000   CD 20 FF 9F                                          ƒ.Ω≡▪  ↓▌..┼...
DS:0010   18 01 10 01                                          ....ñ.  .....
DS:0020   FF FF FF FF FF FF FF FF   FF FF FF FF EB 19 C0 11            δ.└.
DS:0030   A2 01 14 00 18 00 F5 19   FF FF FF FF 00 00 00 00    ó.....J.    ....
DS:0040   05 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00    ........ ........

1 Step  2ProcStep 3Retrieve 4Help ON 5BRK Menu 6       7 up  8 dn  9 le 10 ri
```

Download “Muhammad Rafay_20P-0018_R5.pdf”