$$\begin{tikzcd} & B & AL & & ESi \pm 1 \\ LODS \arrow[ur] \arrow[r] \arrow[dr] & W & AX & DS:ESi \arrow[l] & ESi \pm 2 \\ & D & EAX & ESi \pm 4 & \end{tikzcd}$$

$$\begin{tikzcd} & B & AL \\ STOS \arrow[ur] \arrow[r] \arrow[dr] & W & AX \\ & D & EAX \end{tikzcd}$$

Stores the corresponding register in ES:EDi

MOVS

SCAS

CMPS – compares the size of ESi with EDi

$$\begin{tikzcd} & 0 \quad + \quad \text{(go increasing)} \\ DF \arrow[ur] \arrow[dr] & \\ & 1 \quad - \quad \text{(go decreasing)} \end{tikzcd}$$

STD – set DF = 1

CLD – DF = 0

1. Lowercase to uppercase:

```
data segment
    s db 'a', 'b', 'C', 'd', 'E'
    len equ $ - s
    dis equ 'a' - 'A'
    d times len db 0

code segment
      mov ESI, s
      mov EDI, d
      mov ECX, len
      jecxz end_program
      search
          lodsb        ; AL = element , ESI += 1
          cmp AL, 'a'
          jb no_change                        'a' ≤ AL ≤ 'z'
                cmp AL, 'z'
                ja no_change
                      sub AL, dis

          no_change:
             stosb  ;[EDI] = AL , EDI += 1
    loop search
    end_program:
```

2. Copy a string of doublewords in reverse order

```
data segment
    s dd 1, 100, -15, 4, 11
    l equ ($-s)/4
    d times l dd 0
code segment
    mov ECX, l
    mov ESi, s+(l-1)*4 ; we start from the last elem
    mov EDi, d
    jecxz end
    lp:
        std
        lodsd ; EAX <- [ESi], ESi -= 1*4
        cld
        stosd ; [ES:EDi] <- EAX , EDi += 1*4
    loop lp
    end:
```

3. Find the positions of an element e in an array of words

data segment:

```
               1     3   4           7
    s  dw    1, 2, 3, 2 , 2, 4 , 6, 2
    l  equ  ($ - s) / 2
    e  dw  2
    p  times  l   db  0
```

code segment:

```
    mov ECX, l
    mov EDi, s
    mov ESi, p
    mov AX, [e]
    mov BL, 0
    jecxz  end_loop
    find:
        SCASW            ; cmp between AX and the value from
                                                      EDi
        jne next                                 EDi += 2
        mov [ESi], BL
        inc ESi
        next:
        inc BL
    loop find;
    end_loop:
```

4. Two strings of words are given. Concatenate the low bytes of the words from the first string to the high bytes of the words from the second string and sort them in ascending order in signed representation

```
S1   dw   1023h, 415Ah, 27h, 0F725h, 1FEh
l1   equ ($ - S1)/2
S2   dw   58h, 0A12h, 1470h
l2   equ ($ - S2)/2
d times l1+l2 db 0
code segment:
    start:
    mov ESi, S1
    mov ECX, l1
    mov EDi, d
    jecxz end_1
    lp_1:
        lodsb     ; AL <- [ESi], ESi += 1
        stosb
        lodsb
    loop lp_1    end_1:
```

```
mov ESi , s2
mov ECX, l2
jecxz end_2
lp_2:
     lodsw   ;  AX ← [ESi]
     xchg  AL, AH   ;   AL ⇔ AH (swaps them)
     stosb  ; [EDi] ← AL
 loop lp_2
 end_2:
 mov ECX , l1+l2-1
 mov ESi , d
 jecxz end_4
 outer_loop:
      push ECX
      lodsb      ; AL ← d[i]   , ESi+=1
      mov EDi , ESi
      jecxz end_3
       inner_loop:
            cmp AL, [EDi]
            jle skip
            xchg AL, [ESi]
  skip:     mov [ESi-1] , AL
```

```
for i=0, n-1
  for j=i+1, n
    if d[i] > d[j]
      swap(d[i], d[j])
```

OR

```
mov BL, [EDi]
stosb ; [EDi] ← AL
mov AL, BL
mov [ESi-1], BL
```

```
inc EDi
loop inner_loop
end_3:
pop ECX
loop outr_loop
end_4:
```