Q No 1:

```
;wap(8085) to count the numbers for which upper nibble is higher than the
;lower nibble, store count at end of table of 50 bytes located at C050H

START:  LXI H,C050H
        MVI D,32H       ;50 dec counter
        MVI E,00H       ;counter inc if higher nibble > lower

NEXT:   MOV A,M         ;load from mem
        ANI F0H
        RLC
        RLC
        RLC
        RLC
        MOV B,A         ;higher nibble
        MOV A,M         ;load from mem
        ANI 0F          ;lower nibble
        CMP B
        JNC SKIP
        INR E

SKIP:   INX H
        DCR D
        JNZ NEXT
        MOV M,E
        HLT
```

Q No 2

```
;wap(8085) to transfer 20 bytes of data in a table to another table by
interchanging D1 and D4 bits of each byte.

START:  LXI H,C000H
        LXI B,C050H
        MVI D,14H       ;20 dec down counter

NEXT:   MOV A,M         ;load from mem
        ANI 10H         ;0001 0000
        RRC
        RRC
        RRC
        MOV E,A         ;D4 in D1 position
        MOV A,M         ;load from mem
        ANI 01H         ;get D1 in D1 position
```

```
        CMP B
        MOV A,M         ;flags remain unaffected
        JNC SKIP
        XRI 21          ;0001 0010

SKIP:   STAX B
        INX H
        INX B
        DCR D
        JNZ NEXT
        MOV M,E
        HLT
```

Q No 3

```
;There are two tables with 20 data with starting address 9000H and 9020H
;wap(8085) to add the elements of first table with the content of second
;and store sum and carry into the third and fourth table indexing 9040H
;and 9060H respectively.

START:  LXI H,9000H     ; use L as increment counter
        MVI E,00H       ; for sum

NEXT:   MVI D,00H       ; for carry
        MOV B,H         ; calculating BC= HL+20
        MOV A,L         ; calculate 2nd data loc
        ADI 20H         ; put it in function call
        MOV C,A         ; address complete
        LDAX B          ; A <- [BC]
        ADD M
        MOV E,A         ; E=sum
        JNC SKIP        ; skip if carry is not set
        MVI D,01H       ; set carry

SKIP:   MOV A,C         ; calculate carry save loc
        ADI 20H         ; put it in function call
        MOV C,A         ; address complete
        MOV A,D         ; put carry in A
        STAX B          ; save carry in [BC]
        MOV A,C         ; calculate carry save loc
        ADI 20H         ; put it in function call
        MOV C,A         ; address complete
        MOV A,E         ; put sum in A
        STAX B          ; save carry in [BC]
        INX H
```

```
                MOV A,L         ; get l in accumulator
                CPI 14          ; sets flag Z if equal
                JNZ NEXT        ; jmp if flag Z not set
                HLT
```

QNo 4

```
; WAP(8085) to find the square of ten 8-bit numbers
; which are < 0FH, from table at location C090H.
; Store the result from the end of table

START:  LXI H,C090H     ; source table
        LXI D,C09AH     ; destination table
        MVI C,0AH       ; counter

NEXT:   MVI A,00H       ; initialize to zero
        MOV B,M         ; counter loop to M

LOOP:   ADD M           ; square loop A + [HL]
        DCR B           ; tracks added loop
        JNZ LOOP        ; end of square loop
        STAX D          ; result saved in [DE]
        INX H           ; HL+1
        INX D           ; DE+1
        DCR C           ; c-1
        JNZ NEXT        ; jmp if flag Z not set
        HLT
```

QNo5

No programming questions in 2073 Bhadra on page 5

QNo6

```
; WAP on 8085 to calculate the number of ones
; in the upper nibble of ten 8-bit numbers
; stored in a table.
; Store the count of ones in a location just
; after the table

START:  LXI H,C000H     ; source table
        MVI C,0AH       ; counter
        MVI B,00H       ; counter 1 in higher nibble

LOOP:   MOV A,M         ; initialize to zero
        MVI D,04H       ; counting for 4 bit shift
```

```
NIBBLE:   RLC
          JNC SKIP        ; jump on carry not set
          INR B           ; 1 found

SKIP:     DCR D
          JNZ NIBBLE      ; loop again to shifter
          INX H           ; point to next no
          DCR C           ; dec counter
          JNZ LOOP
          MOV M,B         ; mov 1 counter result to M
          HLT
```

QNo7

```
; WAP 8085 to generate multiplication table
; number stored at 8230H
; store the generated table starting at 8231H.
; For example, if location 8230H has 05H
; then store 05H at823l H, OAH at8232H and so on.

START:    LXI H,8230H     ; source data then dest table
          MVI C,10H       ; counter
          MVI A,00H       ; initialize for sum
          MOV B,M         ; b is counter for add loop
          MVI D,04H       ; counting for 4 bit shift
          MOV E,M         ; use for gen table

LOOP:     ADD E           ; generate multiply
          INX H
          MOV M,A         ; save data in mul table
          DCR C           ; dec counter
          JNZ LOOP
          HLT
```

QNo8

```
; WALP8085 Table1 with 16 data
; transfer to Table2 if 1's in num > 3,
; else store FFH in table2.

START:    LXI H,8085H     ; scr data table
          LXI D,8095H     ; dest table just after scr
          MVI C,08H       ; loop counter 8 bit
          MVI B,00H       ; count 1
```

```
LOOP:   MOV A,M
        MVI C,08H       ; loop counter 8 bit
        MVI B,00H

CNT1:   RLC
        JNC SKIP
        INR B

SKIP:   DCR C
        JNZ CNT1
        MOV A,B
        CPI 03H         ; sets carry A < 3
        JNC LABEL1      ; carry not set jmp
        MVI A,FF
        JMP LABEL2

LABEL1: MOV A,M

LABEL2: STAX D          ; store at dest table
        INX H           ; increment HL
        INX D           ; increment DE
        MOV A,L         ; checking L as a counter
        CPI 94H         ; sets Z if equal
                        ;L->94h end of table
        JNZ LOOP        ; jump if Z not set
        HLT
```

QNo9

```
; WALP8085 Table1 with 16 data
; transfer to Table2 if 1's in num > 3,
; else store FFH in table2.

START:  LXI H,8085H     ; scr data table
        LXI D,8095H     ; dest table just after scr
        MVI C,08H       ; loop counter 8 bit
        MVI B,00H       ; count 1

LOOP:   MOV A,M
        MVI C,08H       ; loop counter 8 bit
        MVI B,00H

CNT1:   RLC
        JNC SKIP
```

```
                INR B

SKIP:           DCR C
                JNZ CNT1
                MOV A,B
                CPI 03H             ; sets carry A < 3
                JNC LABEL1          ; carry not set jmp
                MVI A,FF
                JMP LABEL2

LABEL1:         MOV A,M

LABEL2:         STAX D              ; store at dest table
                INX H               ; increment HL
                INX D               ; increment DE
                MOV A,L
                CPI 94H             ; sets Z if equal
                                    ;L->94h end of table
                JNZ LOOP            ; jump if Z not set
                HLT
```

QNo10

```
; WALP8085 to exchange the bits D6 and D2
; of 200 bytes in memory location 8090H.

START:          LXI H,8000H         ; scr data table
                MVI C,C8H           ; counter 200 dec

LOOP:           MOV A,M
                ANI 40H             ; loop counter 8 bit
                RRC
                RRC
                RRC
                RRC
                MOV B,A             ; upper nibble
                MOV A,M
                ANI 04H             ; lower nibble
                CMP B               ; sets z if equal
                JZ CNT              ; jmp if z is set
                MOV A,M
                XRI 44H
                MOV M,A
```

```
CNT:    INX H
        DCR C
        JNZ LOOP        ; jump if Z not set
        HLT
```

QNo11

```
; WALP8085 to divide as [9070H]/[9071H]
; store rem-> [9072H] & quotient-> [9073H]

START:  LXI H,9070H     ; scr data table
        LDA 9071H       ; load accumulator direct
        MOV D,A         ; div in d
        MVI B,00H       ; quotient
        MOV A,M

LOOP:   SUB D           ; subtract until (A is -ve)
        JC NEG          ; borrow true
        INR B           ;
        JMP LOOP

NEG:    ADD D           ; fix it make A +ve
        MOV C,A         ; remainder
        STA 9072H
        MOV A,B
        STA 9073H
        HLT
```

QNo12

```
; WAP 8085 to convert 10 numbers stored at 4350H
; to binary and store the the result at4360H

START:  LXI H,4360H     ; scr data table
        LXI D,436AH     ; destination table
        MVI C,0AH       ; counter 10 dec

LOOP:   MOV A,M
        ANI F0H
        CPI 00H
        JZ STEP
        MOV A,M
        SUI 06H
        JMP SK
```

```
STEP:   MOV A,M

SK:     STAX D
        INX D
        INX H
        DCR C
        JNZ LOOP
        HLT
```

QNo13

```
;WAP 8085 to transfer 8-bit number table1 to
;table2 setting bit D5, if the number < 80H
;else transfer number by resetting bit D6. ,

START:  LXI H,8010H     ; scr data table
        LXI D,8020H     ; destination table
        MVI C,0AH       ; counter 10 dec

LOOP:   MOV A,M
        CPI 80H
        JC SETD6
        ORI 20H         ;SET D5 ORing
         JMP SKIP
SETD6:  ORI 40H         ;SET D6 ORing

SKIP:   STAX D
        INX D
        INX H
        DCR C
        JNZ LOOP
        HLT
```

QNo14

```
; WAP 8085 sort set of 3 data at 9040H in ascending order
; Store sorted table at 9054H
; move data
; set of 3 readings mean 6 byte data
; readings are 16 bit data
        LXI H,9040H     ; copy data to dest
        LXI B,9054H
```

```
                MVI D,0AH           ; 20 dec down counter

COPY:           MOV A,M
                STAX B
                INX H
                INX B
                DCR D
                JNZ COPY
; sorting
                MVI D,06H

AGAIN:          LXI H,9054H         ; sort at destination
                MVI C,05H           ; SUB counter main-1

                NEX: MOV A,M
                INX H
                CMP M               ; compare with 2nd data
                JC NOSWAP           ; A < M sets carry
                MOV B,M
                MOV M,A
                DCX H
                MOV M,B
                INX H

NOSWAP:         DCR C
                JNZ NEX
                DCR D
                JNZ AGAIN
                HLT
```

QNo15

```
; WAP 8085 to add all the no. from table of
; 8-bit numbers whose higher nibble > 6
; and store the 16 bit result just after the table.

                LXI H,A000H
                MVI C,20H     ; 32 dec items
                MVI D,00H     ; MS bit of result
                MVI E,00H     ; LS bit of result

NEX:            MOV A,M
                CPI 6FH             ; IF higher nibble less
                ; 70H>6FH to include all possibilities
```

```
        JC SKIP         ; do not add
        ; sum higher nibble >6
        ADD E
        MOV E,A         ; accumulate sum in E
        JNC SKIP        ; check carry
        INR D           ; increase for carry

SKIP:   INX H
        DCR C
        JNZ NEX
        MOV M,D
        INX H
        MOV M,E
        HLT
```

QNo16

```
; WAP 8085 to add corresponding data form two
; table if the data from the first table is
; smaller than the second table else
; subtract data of second table from
; the first table.
; assuming tables start at A000H and A010H
; counter used

        LXI SP,FFFFH    ; stack at max limit of mem
        LXI H,A000H     ; table 1
        XTHL            ; 2
        LXI H,A010H     ; table 2
        LXI D,A020H     ; destination table
        MVI C,0FH

NEXT:   MOV B,M         ; 2
        XTHL            ; 1
        MOV A,M
        CMP B           ; A=B sets Carry
        JZ SUM
; diff
        MOV B,M         ; 1
        XTHL            ; 2
        MOV A,M         ; 2
        SUB B
        JMP STORE
```

```
SUM:    ADD B
        XTHL            ; 2

STORE:  STAX D
        INX D
        INX H           ; 2
        XTHL            ; 1
        INX H           ; 1
        XTHL            ; 2
        DCR C
        JNZ NEXT
        HLT
```

QNo17

```
; WAP 8085 to count no. of -ve element in a data block
; block containing 16 bytes of data
; store the count at the end of the block
; if the count is greater than 8
; otherwise store 0.
; assuming tables start at A000H and A010H
; counter used
        LXI H,A000H     ; table 1
        MVI C,00H       ; count neg nos
        MVI B,0F        ; countdown

NEXT:   MOV A,M
        RAR             ; get MSB in carry
        JNC SKIP
        ; count neg nos
        INR C

SKIP:   INX H
        DCR B
        JNZ NEXT
        MOV A,C
        CPI 08H
        JNC JU
        MVI A,00H

JU:     MOV M,A
        HLT
```

QNo18

```
; WAP 8085 to transfer eight-bit numbers from
; 9080H to 9090H if if bit D5 is 1 & D3 is 0,
; Otherwise transfer data by changing bit D2 & D0
; 1->0 or 0->1
; Assume there are ten numbers.
; assuming tables start at A000H and A010H
; counter used

            LXI H,9080H     ; scr
            LXI D,9090H     ; dest
            MVI C,0AH       ; countdown

NEXT:       MOV A,M
            ANI 28H         ; mask for D5 & D3
            CPI 20H         ; check for D5=1, D3=0
            JNZ CHANGE      ; zero bit not set
            MOV A,M
            JMP STORE

CHANGE:     MOV A,M         ; data flip
            XRI 44H         ; flip D2 & D6

STORE:      STAX D
            INX H
            INX D
            DCR C
            JNZ NEXT
            HLT
```

QNo19

```
; There are two tables T1, T2
; in memory having ten eight bit data in each.
; WAP(8065) to find the diff. of corresponding
; element of these two tables. Store the result
; of each operation on corresponding element
; of the third table. Remember that
; the result should not be negative
; it should be |T1 - T2|.
; assuming tables start at A000H and A010H
; counter used

            LXI H,A000H     ; t1
```

```
                ; use DE as var mem pointer
                ; t2 at +10h
                ; diff at +20h

                MVI D,A0H       ; calculate E later
                MVI C,0AH       ; countdown
                ; working register b

NEXT:           MOV B,M
                MOV A,L
                ADI 10H
                MOV E,A         ; calc address of t2 element
                LDAX D
                CMP B           ; sets carry a if b<m
                JNC XYZ         ; jumps when C=0
                LDAX D
                MOV B,A
                MOV A,M
                SUB B
                JMP STORE

XYZ:            LDAX D
                SUB B

STORE:          MOV B,A         ; hold the diff
                MOV A,L         ; calc add of the dest
                ADI 20H
                MOV E,A         ; dest in DE
                MOV A,B         ; get diff in A
                STAX D
                INX H
                DCR C
                JNZ NEXT
                HLT
```

QNo20

```
; wap(8085) to convert and copy Lowercase ASCII
; codes To upper from memory location 9050H->90A0H
; if any otherwise copy as they are.
; Assume there are 50 codes in the source memory.
; [Note: ASCII Code for A:65...Z=90, a:97...2:122]

                LXI H,9050H
```

```
        LXI D,90A0H
        MVI C,32H       ; dec 50 characters

NEXT:   MOV A,M
        CPI 61H         ; ascii of a=61H
        JC SKIP         ; A < (a)
        CPI 7BH         ; ascii of z=7AH compare 7ah +01h
        JNC SKIP        ; A < (z+1)
        SBI 20H         ; (a-A) = 20h  effectively a-z -> A-Z

SKIP:   STAX D
        INX H
        INX D
        DCR C
        JNZ NEXT
        HLT
```