Topic:

ASSIGNMENT – 03

Problem – 01

⇒ A set of N data bytes is stored in m/m location starting from 2501_H. The value of N is stored in 2500_H. Write a program to store these data bytes from m/m location 2600_H if D_0 or D_7 is 1, otherwise reject the data byte.

⇒ Address	Command	Operation	Hex Code
2000	LXI H, 2500_H	L←00, H←25	21 00 25
2003	MOV C, M	C ← M[HL]	4E
2004	LXI D, 2600_H	E←00, D←26	11 00 26
2007	INX H	Increment HL	23
2008	MOV A, M	A← M[HL]	7E
2009	RAR	Rotate Right	1F
200A	JNC 2013_H	Jump not carry	D2 13 20
200D	MOV A, M	A ← M[HL]	7E
200E	STAX D	[DE] ← A	12
200F	INX D	Increment DE	13
2010	JMP $201B_H$	Jump	C3 1B 20
2013	MOV A, M	A ← M[HL]	7E
2014	RAL	Rotate Left	17
2015	JNC $201B_H$	Jump not carry	D2 1B 20
2018	MOV A, M	A ← M[HL]	7E
2019	STAX D	[DE] ← A	12
201A	INX D	Increment DE	13
201B	DCR C	C ← C−1	0D
201C	INX H	Increment HL	23
201D	JNZ 2008_H	Jump Not zero	C2 08 20
2020	HLT	stop	76

Algorithm =>

```
        LXI  H, 2500H      //  Load address in HL pair
        MOV  C,M           //  Move M[HL] to C
        LXI  D, 2600H      //  Load address in DE pair
        INX  H             //  Increment HL pair.
Loop:   MOV  A,M           //  Move M[HL] to A.
        RAR                //  Rotate Right Accumulator
        JNC  Loop1         //  Jump if not carry to loop1
        MOV  A,M           //  Move M[HL] to A
        STAX D             //  Store A in address of DE pair
        INX  D             //  Increment DE pair
        JMP  Loop2         //  Jump to loop2
Loop1 : MOV  A,M           //  Move M[HL] to A
        RAL                //  Rotate Accumulator Left.
        JNC  Loop2         //  Jump to loop2 if no carry
        MOV  A,M           //  Move M[HL] to A.
        STAX D             //  Store A in address of DE pair
        INX  D             //  Increment DE pair
Loop2 : DCR  C             //  Decrement C by 1
        INX  H             //  Increment HL pair
        JNZ  Loop          //  Jump to loop if no zero
        HLT                //  Stop
```

Example :-

9b, FF, 3E, 5E, 5F

Answer → 9b, FF, 5F

Pooblem - 02

⇒ There are N data bytes stored from m/m location 2200_H. The value of N is stored in $21FF_H$. Write an 8085 program to find the sum of integers whose LSB and MSB are 1. Store the result in 2500_H and 2501_H.

⇒

Address	Command	Operation	HexCode
2000	LXI H, $21FF_H$	$L \leftarrow FF_H$, $H \leftarrow 21_H$	21 FF 21
2003	MOV C, M	C ← M[HL]	4E
2004	INX H	Increment HL	23
2005	MVI D, 00_H	D ← 00_H	16 00
2007	MVI B, 00_H	B ← 00_H	06 00
2009	MOV A, M	A ← M[HL]	7E
200A	RAR	Rotate right	1F
200B	JNC $201D_H$	Jump not carry	D2 1D 20
200E	MOV A, M	A ← M[HL]	7E
200F	RAL	Rotate Left	17
2010	JNC $201D_H$	Jump not carry	D2 1D 20
2013	MOV A, M	A ← M[HL]	7E
2014	ADD B	A ← A+B	80
2015	MOV B, A	B ← A	47
2016	JC $201C_H$	Jump if carry	DA 1C 20
2019	JMP $201D_H$	Jump	C3 1D 20
201C	INR D	D ← D+1	14
201D	INX H	Increment HL	23
201E	DCR C	C ← C-1	0D
201F	JNZ 2009_H	Jump not zero	C2 09 20
2022	MOV A, B	A ← B	78
2023	STA 2500_H	[2500] ← A	32 00 25
2026	MOV A, D	A ← D	7A
2027	STA 2501_H	[2501] ← A	32 01 25
202A	HLT	Stop	76

Algorithm :-

```
        LXI  H, 21FFH   // Load address in HL
        MOV  C,M        // Move M[HL] to C
        INX  H          // Increment HL
        MVI  D, 00H     // Move 00 to D.
        MVI  B, 00H     // Move 00 to B
Loop :  MOV  A,M        // Move M[HL] to A
        RAR             // Rotate Accumulator Right
        JNC  LOOP1      // Jump to Loop1 if not carry.
        MOV  A,M        // Move M[HL] to A.
        RAL             // Rotate Accumulator Right
        JNC  Loop1      // Jump to loop1 if not carry.
        MOV  A,M        // Move M[HL] to A
        ADD  B          // Add A with B
        MOV  B,A        // Move A to B
        JC   Branch     // Jump if carry to branch.
        JMP  Loop1      // Jump loop1
Branch : INR D          // Increment D by 1
Loop1 :  INX H          // Increment HL pair
        DCR  C          // Decrement C. by 1
        JNZ  Loop       // Jump to loop if not zero
        MOV  A,B        // Move B to A
        STA  2500H      // Store A in 2500H
        MOV  A,D        // Move D to A
        STA  2501H      // Store A in 2501H
        HLT             // Stop
```

Example ⇒

9b, FF, 3E, 5E, 5F.

Answer → 9b, FF

Problem - 03

Write an 8085 program to generate N^{th} fibonacci number using function and store it in 2050_H. The value of N (8-bit) is stored in memory 2060_H.

⇒

Address	Command	Operation	Hex Code
2000	LXI H, 2060_H	L←60, H←20	21 60 20
2003	MOV A, M	A ← M[HL]	7E
2004	MVI B, 01_H	B ← 01	06 01
2006	CMP B	Compare A & B	B8
2007	JZ 2027_H	Jump if zero	CA 27 20
200A	MOV A, M	A ← M[HL]	7E
200B	MVI D, 02_H	D ← 02_H	16 02
200D	CMP D	Compare A & D	BA
200E	JZ $202D_H$	Jump if Zero	CA 2D 20
2011	MOV A, M	A ← M[HL]	7E
2012	MVI C, 02_H	C ← 02_H	0E 02
2014	SUB C	A ← A − C	91
2015	MOV C, A	C ← A	4F
2016	MVI B, 00_H	B ← 00_H	06 00
2018	MVI D, 01_H	D ← 01_H	16 01
201A	MOV A, B	A ← B	78
201B	ADD D	A ← A+D	82
201C	MOV B, D	B ← D	42
201D	MOV D, A	D ← A	57
201E	DCR C	C ← C−1	0D
201F	JNZ $201A_H$	Jump not zero	C2 1A 20

2022	mov A,D	A←D	7A
2023	STA 2050_H	[2050]← A	32 50 20
2026	HLT	Stop	76
2027	MVI A, 00_H	A ← 00_H	3E 00
2029	STA 2050_H	[2050]← A	32 50 20
202C	HLT	stop	76
202D	MVI A, 01_H	A← 01_H	3E 01
202F	STA 2050_H	[2050]← A	32 50 20
2032	HLT	Stop	76

— o —

Algorithm :->

```
        LXI   H, 2060H        // Load address in HL pair
        MOV   A, M            // Move M[HL] to A
        MVI   B, 01H          // Move B1 to B
        CMP   B               // Compare A and B.
        JZ    2027H (Loop1)   // Jump zero to loop1
        MOV   A,M             // Move M[HL] to A
        MVI   D, 02H          // Move 02 to D
        CMP   D               // Compare A to D
        JZ    Loop2           // Jump to loop2 if zero
        MOV   A,M             // Move M[HL] to A
        MVI   C, 02H          // Move 02 to C
        SUB   C               // Subtract C from C
        MOV   C,A             // Move A to C.
        MVI   B , 00H         // Move 00 to B
        MVI   D, 01H          // Move 01 to D.
Loop :  MOV   A,B             // Move B to A
        ADD   D               // Add D to A
        MOV   B,D             // Move D to B
        MOV   D,A             // Move A to D
        DCR   C               // Decrement C.
        JNZ   loop            // Jump to loop if zero.
        MOV   A,D             // Move D to A
        STA   2050H           // Store A in 2050H
        HLT                   // Stop
Loop1 : MVI   A, 00H          // Move 00 to A
        STA   2050H           // Store A in 2050H
        HLT                   // Stop
Loop2 ; MVI   A, 01H          // Move 01 to A
        STA   2050H           // Store A in 2050H
        HLT                   // Stop
```

Problem - 04

⇒ Write a program to transfer ~~a~~ block of bytes of size N from location1 to location2 (location2 > location1) when the size of overlap between the two locations is defined by M. The value of N and M are stored in $201E_H$ and $201F_H$ respectively.

⇒ Address	Command	Operation	Hex Code
3000	LXI H, $201E_H$	L←1E, H←20	21 1E 20
3003	MOV A,M	A ← M[HL]	7E
3004	INX H	Increment HL	23
3005	MOV C,M	C ← M[HL]	4E
3006	MOV B,A	B ← A	47
3007	SUB C	A ← A − C	91
3008	ADD B	A ← A + B	80
3009	MOV C,B	C ← B	48
300A	DCR C	C ← C − 1	0D
300B	MVI B, 00_H	B ← 00_H	06 00
300D	INX H	Increment HL	23
300E	DAD B	Double Addition	09
300F	MOV B,C	B ← C	41
3010	INR B	B ← B + 1	04
3011	MOV C,A	C ← A	4F
3012	DCR C	C ← C − 1	0D
3013	LXI D, 2020_H	E ← 20, D ← 20	11 20 20
3016	INX D	Increment DE	13
3017	DCR C	C ← C − 1	0D

3018	JNZ 3016$_H$	Jump not zero	C2 16 30
301B	JMP 301E$_H$	Jump	C3 1E 30
301E	MOV A,M	A ← M[HL]	7E
301F	STAX D	[DE] ← A	12
3020	DCX D	Decrement DE	1B
3021	DCR B	B ← B−1	05
3022	DCX H	Decrement HL	2B
3023	JNZ 301E$_H$	Jump not zero	C2 1E 30
3026	HLT	Stop	76

— 0 —

Algorithms :-

```
             LXI  H, 201E_H      // Load address in HL pair
             MOV  A,M            // Move M[HL] to A
             INX  H              // Increment HL pair
             MOV  C,M            // Move M[HL] to C
             MOV  B,A            // Move A to B
             SUB  C              // Move and Subtract C from A
             ADD  B              // Add B with A
             MOV  C,B            // Move B to C
             DCR  C              // Decrement C by 1
             MVI  B,00           // Move 00 to B
             INX  H              // Increment HL pair
             DAD  B              // Double addition with B
             MOV  B,C            // Move C to B
             INR  B              // Increment B
             MOV  C,A            // Move A to C
             DCR  C              // decrement C by 1
             LXI  D, 2020_H      // Load address in DE pair
Loop1 :      INX  D              // Increment DE pair
             DCR  C              // Decrement C by 1
             JNZ  3016_H (Loop1) // Jump to Loop1 if not zero
             JMP  Loop           // Jump to loop
Loop  :      MOV  A,M            // Move M[HL] to A
             STAX D              // Store A in address stored in DE
             DCX  D              // Decrement DE pair
             DCR  B              // Decrement B
             DCX  H              // Decrement HL pair
             JNZ  loop           // Jump to loop if not zero
             HLT                 // Stop
```

Problem - 5

⇒ Write a program to flash "BCSE - II" in the address and data fields with flashing rate of 0.5 seconds.

⇒

Address	Command	Operation	Hex Code
2000	LXI SP, $20FF_H$	Initialize	31 FF 20
2003	CLEAR	clear display	CD 47 03
2006	XRA A	XOR with A	AF
2007	MOV B, A	B ← A	47
2008	LXI H, 2050	L←50, H← 20	21 50 20
200B	OUTPUT	display	CD D0 05
200E	MVI A, 01_H	A ← 01_H	3E 01
2010	MVI B, 00_H	B ← 00_H	06 00
2012	LXI H, 2054_H	L←54, H←20	21 54 20
2015	OUTPUT	display	CD D0 05
2018	LXI D, 0000_H	E←00, D←00	11 00 00
201B	DELAY	delay	CD BC 03
201E	CLEAR	Clear	CD 47 03
2021	LXI D, 0000_H	D←00, E←00	11 00 00
2024	DELAY	delay	CD BC 03
2027	JMP 2006_H	Jump	C3 06 20

—o—

Algorithm :-

```
          LXI  SP, 20FFH        // Initialize stack pointer

          CLEAR       //  Clear the display

START :   XRA  A     //  A is 00 to display character in address
                                                          field.
          MOV  B, A     //  Move A to B

          LXI  H, 2050H    //  Load address in HL pair

CALL :    OUTPUT          // Call output routine to display four
                             characters in address field.

          MVI  A, 01H    //  Move 01 to A

          MVI  B, 00H    //  Move 00 to B

          LXI  H, 2054H  //  Load address in HL pair

CALL :    OUTPUT          //  Display last two character of
                              in the data field.

      LXI  D, 0000       //  Display for about 0.5 seconds
      CALL : DELAY

      CALL : CLEAR   //  clear the display

          LXI D, 0000   //  Clear display for 0.5 seconds

          CALL : DELAY

          JMP  START       //  Jump to start.
```

=>

```
2050  ←  0b
2051  ←  0C
2052  ←  05
2053  ←  0E
2054  ←  01
2055  ←  01
```