a// Add 16 bit number

LXI H, 20FF  
LXI D ,5642  
  
MOV A,L  
ADD E  
MOV L,A  
  
MOV A,H  
ADC D  
MOV H,A

....................................................................................................................................  
// Check Divisibility  
  
    LXI H,3000  
  MOV A,M  
  MVI B,03  
  
LOOP:   SUB B  
  CMP B  
  JZ YES  
  JC NO  
  JMP LOOP  
  
YES:   INX H  
  MVI M,01  
  HLT  
  
NO:   INX H  
  MVI M,00  
  HLT  
  
  
# ORG 3000  
# DB 09H  
  
................................................................................................................................  
  
check even or odd  
    MVI A,51  
  ANI 01  
  JZ EVEN  
  MVI B,01  
  JMP END  
  
EVEN:   MVI B,02  
  
END:   HLT  
................................................................................................................................  
  
Check Even or Odd Parity  
  
// Parity Flag (P): The Parity flag is set if the number of 1 bits in the ;Accumulator (A) is even (even parity). It is cleared if the number of 1 ;bits is odd (odd parity)  
  LXI H,3000  
  MOV A,M  
  ANA A  
  JP EVEN\_PARITY  
  MVI B,01  
  JMP END  
  
EVEN\_PARITY:   MVI B,02  
  
END:   HLT  
  
#ORG 3000  
#DB 00H  
  
..........................................................................................................................  
  
//largest of the numbers  
LXI H, XX01H        ; Load starting address of the data block into HL pair  
MOV A, M            ; Load the first data byte into accumulator (assume it’s the largest initially)  
INX H               ; Move to the next memory location  
  
LXI D, XX25H        ; Load end address of the data block into DE pair  
  
LARGEST\_LOOP:  
    CMP M           ; Compare current data byte with A (A holds the largest so far)  
    JNC SKIP        ; If A >= M, skip updating A  
    MOV A, M        ; If A < M, update A with the new larger value  
  
SKIP:  
    INX H           ; Move to the next memory location  
    MOV E, L        ; Copy the lower byte of HL to E  
    MOV D, H        ; Copy the upper byte of HL to D  
    CMP D           ; Compare DE with the end address XX25H  
    JNZ LARGEST\_LOOP ; Repeat until HL reaches the end address  
  
STA XX26H           ; Store the largest number at memory location XX26  
  
  
#ORG 2201  
#DB 01H,02H,03H,04H,05H,06H,07H,08,09H,0A,0B,0C,AA,BB,CC,0D,  
  
...............................................................................................................................  
// moving data  
    LXI H,2050  
  LXI D,2060  
  MVI B,06  
  
NEXT:   MOV A,M  
  STAX D  
  INX H  
  INX D  
  DCR B  
  JNZ NEXT  
  HLT  
# ORG 2050  
# DB 01H,02H,03H,04H,05H,06H  
................................................................................................................................// Moving data in reverse order  
LXI H,2050  
LXI D,2065  
  
MVI B,06H  
NEXT:  MOV A,M  
  
STAX D  
  
INX H  
DCX D  
DCR B  
JNZ NEXT  
HLT  
  
#ORG 2050  
#DB 01H,02H,03H,04H,05H,06H  
  
.................................................................................................................................// swapping of data  
  
    LXI H,2050  
  LXI D,2060  
  MVI B,06  
  
NEXT:   MOV C,M  
  LDAX D  
  MOV M,A  
  MOV A,C  
  STAX D  
  INX H  
  INX D  
  DCR B  
  JNZ NEXT  
  HLT  
# ORG 2050  
# DB 01H,02H,03H,04H,05H,06H  
# ORG 2060  
# DB 07H,08H,09H,0A,0BH,0CH  
  
.................................................................................................................................  
//Multiplication of 2 8 bit numbers   
    LXI H,3000  
  MOV A,M  
  INX H  
  MOV E,M  
  MVI D,00  
  LXI H,0000  
  
  CPI 00  
  JZ RESULT  
LOOP:   DAD D  
  DCR A  
  JNZ LOOP  
  
RESULT:   SHLD 3002  
  RST 5  
  
..................................................................................................................................  
count the numbers of ones  
  
MVI A,0AH  
MVI B,08H       ; Copy the original data to register B for processing  
MVI C, 00H     ; Initialize the count of 1's to 0 in register C  
  
COUNT\_ONES:  
    RRC         ; Rotate the accumulator right; the LSB goes to the Carry flag  
    JNC NEXT    ; If Carry flag is 0, skip incrementing the count  
    INR C       ; If Carry flag is 1, increment the count in register C  
  
NEXT:  
    ; Reload the original value from B (shifted 1 position) into A  
    DCR B       ; Decrement B (to track loop count for 8 bits)  
    JNZ COUNT\_ONES ; Repeat the loop until all 8

//ADD WITH CARRY (IN DECIMAL USING DAA)

LXI H,3000H  
MOV A,M  
INX H  
MVI B,00H  
ADD M  
DAA  
JNC LOOP  
INR B  
LOOP:STA 3003H  
MOV A,B  
STA 3002H  
HLT

------------------------------------------------------------------------------------------

//PRODUCT OF 2 8-BIT NUMBERS  
  
LXI H,3000H  
MOV B,M    
MVI A,00H ; initializing so  that the accumulator does not contain garbage value  
MVI D,00H ; for carry  
INX H  
MOV C,M  
LOOP2:ADD B  
JNC LOOP  
INR D  
LOOP: DCR C  
JNZ LOOP2  
STA 3003H  
MOV A,D  
STA 3004H  
HLT  
  
-----------------------------------------------------------------------------------------------  
//DIVISION OR DIVISIBILITY  
  
LXI H,3000H  
MOV A,M  
INX H  
MOV B,M  
MVI C,00H  
LOOP: SUB B  
INR C  
CMP B  
JNC LOOP  
STA 3003H  
MOV A,C  
STA 3004H  
HLT  
  
------------------------------------------------------------------------------------------------

//EVEN ODD  
  
LDA 3000H  
RAR  
JC LOOP  
MVI A,00H  
STA 3001H  
HLT  
LOOP: MVI A,01H  
STA 3001H  
HLT

-----------------------------------------------------------------------------------------------

//TRANSFER DATA FROM 1 ADDRESS TO ANOTHER  
  
LXI H,3000H  
LXI D,3010H  
MVI C,05H  
LOOP: MOV A,M  
STAX D  
INX H  
INX D  
DCR C  
JNZ LOOP  
HLT  
  
-----------------------------------------------------------------------------------------------

//EXCHANGE OF DATA  
  
LXI H,3000H  
LXI D,3010H  
MVI C,05H  
LOOP:MOV A,M  
MOV B,A  
LDAX D  
MOV M,A  
MOV A,B  
STAX D  
INX H  
INX D  
DCR C  
JNZ LOOP  
HLT  
  
------------------------------------------------------------------------------------------------  
//FIBONACCI  
  
LXI H,3000H  
LDA 2000H  
MOV D,A  
MVI B,00H  
MVI C,01H  
MVI A,00H  
LOOP: MOV M,A  
ADD C  
MOV B,C  
MOV C,A  
MOV A,B  
INX H  
DCR D  
JNZ LOOP  
HLT  
  
------------------------------------------------------------------------------------------------

//FACTORIAL  
  
LXI H,3000H  
MOV B,M  
MVI D,01H  
NEXT: CALL MUL  
DCR B  
JNZ NEXT  
INX H  
MOV M,D  
HLT  
MUL:  
MOV E,B  
XRA A  
LOOP: ADD D  
DCR E  
JNZ LOOP  
MOV D,A  
RET