```
PRINT MACRO MSG
    MOV AH, 09H
    LEA DX, MSG
    INT 21H
    ENDM
;-------------------------------------------------------------------------------------------------------------------------------------------------------
DOSSEG
.MODEL SMALL
.STACK 100H

.DATA
;-------------------------VARIABLES-------------------------------------------------------------------------------------------------------------------
        MESSAGE_WELCOME         DB      10,13, 'WELCOME TO THE NUMBER SYSTEM CONVERSION! ', 13, 10
                        DB      'PLEASE CHOOSE HOW YOU WISH TO PROCEED:', 13, 10
                        DB      '1- BINARY', 13, 10
                        DB      '2- OCTAL', 13, 10
                        DB      '3- DECIMAL', 13, 10
                        DB      '4- HEXADECIMAL$', 13, 10
        JUMP                    DB      ?
        EXIT_KEY                DB      10,13,'PRESS ANY KEY TO EXIT...$'
        OPT_ERROR               DB      10,13,'INVALID OPTION!$'
;-------------------------------------------------------------------------------------------------------------------------------------------------------
;-------------------------BINARY VARIABLES-----------------------------------------------------------------------------------------------------------
        MENU_BIN                DB      10,13,'PLEASE INPUT YOUR CHOICE:',13,10
                        DB      '1:BINARY-->OCTAL',13,10
                        DB      '2:BINARY-->DECIMAL',13,10
                        DB      '3:BINARY-->HEXADECIMAL',13,10
                        DB      '0:EXIT',13,10,'$'
```

```
        MESS1_BIN       DB      10,13,'PLEASE INPUT A BINARY NUMBER:$'
        JUMP_BIN        DB      ?
;-------------------------------------------------------------------------------------------------------------------
;BIN_OCT-----------------------------------------------------------------------------------------------------------
---
        MESS2_BIN       DB      10,13,'EQUIVALENT OCTAL IS:$'
;-------------------------------------------------------------------------------------------------------------------
;BIN_DEC-----------------------------------------------------------------------------------------------------------
---
        MESS3_BIN       DB      10,13,'EQUIVALENT DECIMAL IS:$'
;-------------------------------------------------------------------------------------------------------------------
;BIN_HEX-----------------------------------------------------------------------------------------------------------
---
        MESS4_BIN       DB      10,13,'EQUIVALENT HEXADECIMAL IS:$'
;-------------------------------------------------------------------------------------------------------------------
;----------------------OCTAL VARIABLES----------------------------------------------------------------------------
-------
        MENU_OCT        DB      10,13,'PLEASE INPUT YOUR CHOICE:',13,10
                        DB      '1:OCTAL-->BINARY',13,10
                        DB      '2:OCTAL-->DECIMAL',13,10
                        DB      '3:OCTAL-->HEXADECIMAL',13,10
                        DB      '0:EXIT',13,10,'$'
        JUMP_OCT        DB      ?
;-------------------------------------------------------------------------------------------------------------------
;OCT-BIN-----------------------------------------------------------------------------------------------------------
----
        VAR_OCT_BIN     DB      0
        MESS1_OCT_BIN           DB      10,13,"ENTER AN OCTAL NUMBER: $"
        MESS2_OCT_BIN           DB      10,13,"EQUIVALENT BINARY IS: $"
;-------------------------------------------------------------------------------------------------------------------
;OCT-DEC-----------------------------------------------------------------------------------------------------------
-----
        MESS1_OCT_DEC           DB      10,13,'ENTER 1 TO 6 OCTAL DIGITS:$'
```

```
        MESS2_OCT_DEC           DB      10,13,'EQUIVALENT DECIMAL IS:$'
        MESS3_OCT_DEC           DB      10,13,'INVALID OCTAL NUMBER!$'
        VAR_OCT_DEC     DB      8,?,8 DUP(?)                    ;VARIABLE
WITH 3 SECTIONS.
        RESULT_OCT_DEC  DB      11 DUP('$')                     ;RESULT
COULD HAVE 10 DIGITS.
;-------------------------------------------------------------------------------------------------------------------
;OCT-HEX-----------------------------------------------------------------------------------------------------------
        MESS1_OCT_HEX           DB      10,13,"ENTER AN OCTAL NUMBER: $"
        MESS2_OCT_HEX           DB      10,13,'EQUIVALENT HEXADECIMAL IS: $'
        HEX_TABLE               DB      "0123456789ABCDEF"
;-------------------------------------------------------------------------------------------------------------------
;-----------------------DECIMAL VARIABLES-----------------------------------------------------------------------
        MENU_DEC                DB      10,13,'PLEASE INPUT YOUR CHOICE:',13,10
                        DB      '1:DECIMAL-->BINARY',13,10
                        DB      '2:DECIMAL-->OCTAL',13,10
                        DB      '3:DECIMAL-->HEXADECIMAL',13,10
                        DB      '0:EXIT',13,10,'$'
        JUMP_DEC                DB      ?
;-------------------------------------------------------------------------------------------------------------------
;DEC-BIN-----------------------------------------------------------------------------------------------------------
        DEC_BIN_VAR             DB      9 DUP (' '),'$'                 ;WILL CONTAIN THE
CHAIN OF BITS
        VAR1_DEC_BIN            DB      ?                               ;VAR1 WILL BE
USED TO CONVER NUMBER
        NUM_DEC_BIN             DB      ?                                       ;VARIABLE
FOR INPUT NUMBER
        AUX_DEC_BIN             DB      ?                               ;AUXILIARY
VARIABLE
        MESS1_DEC_BIN           DB      10,13,"ENTER DECIMAL NUMBER 0-99: $", 10, 13
```

```
        MESS2_DEC_BIN                   DB      10,13,'EQUIVALENT BINARY IS:$'
;-------------------------------------------------------------------------------------------------------------------------------
;DEC-OCT-----------------------------------------------------------------------------------------------------------------------
        MESS1_DEC_OCT                   DB      10,13,"ENTER A DECIMAL NUMBER: $"
        MESS2_DEC_OCT                   DB      10,13, "INVALID DECIMAL NUMBER...$"
        MESS3_DEC_OCT                   DB      10,13, "EQUIVALENT OCTAL IS: $"
        MESS4_DEC_OCT                   DB      6
                        DB      0
                        DB      6 DUP(0)
        MULTIPLIER_DEC_OCT  DB      10
;-------------------------------------------------------------------------------------------------------------------------------
;DEC-HEX-----------------------------------------------------------------------------------------------------------------------
        MESS1_DEC_HEX       DB      10,13,'ENTER A DECIMAL NUMBER:$'
        MESS2_DEC_HEX                   DB      10,13,'EQUIVALENT HEXADECIMAL IS:$'
;-------------------------------------------------------------------------------------------------------------------------------
;------------------HEXADECIMAL VARIABLES--------------------------------------------------------------------------------------
        MENU_HEX            DB      10,13,'PLEASE INPUT YOUR CHOICE:',13,10
                        DB      '1:HEXADECIMAL-->BINARY',13,10
                        DB      '2:HEXADECIMAL-->DECIMAL',13,10
                        DB      '0:EXIT',13,10,'$'
        JUMP_HEX            DB      ?
;-------------------------------------------------------------------------------------------------------------------------------
;HEX-BIN-----------------------------------------------------------------------------------------------------------------------
        MESS1_HEX_BIN                   DB      10,13,"YOUR HEXADECIMAL INPUT: $"
        MESS2_HEX_BIN                   DB      10,13,'EQUIVALENT BINARY IS:$'
;-------------------------------------------------------------------------------------------------------------------------------
;HEX-DEC-----------------------------------------------------------------------------------------------------------------------
```

```
        MESS1_HEX_DEC                   DB      10,13,'ENTER 1 TO 4 HEX DIGITS (CAPITAL
LETTERS):$'
        MESS2_HEX_DEC                   DB      10,13,'EQUIVALENT DECIMAL IS:$'
        MESS3_HEX_DEC                   DB      10,13,'INVALID HEXADECIMAL NUMBER!$'
        VAR_HEX_DEC             DB      5,?,5 DUP(?)                                    ;VARIABLE
WITH 3 SECTIONS.
        RESULT_HEX_DEC                  DB      6 DUP('$')
        ;RESULT COULD HAVE 5 DIGITS.

;-------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------

.CODE
MAIN PROC
MOV AX,@DATA
MOV DS,AX

START:

            PRINT MESSAGE_WELCOME
            MOV     AH, 0H
            INT     16H
                    MOV     JUMP,AL
            CMP     JUMP,'1'
            JZ      BINARY
                    JMP OPT2

;-------------------------BINARY CONVERSION-----------------------------------------------------------------------------
----------------
BINARY:
        START_BIN:
                PRINT MENU_BIN
                MOV AH,0H
                INT 16H
```

```
                MOV JUMP_BIN,AL
                CMP JUMP_BIN,'1'
                JZ  BIN_OCT
                JMP OPT2_BIN

;-------------------------------------------------------------------------------------------------------
BIN_OCT:
        CALL  INPBIN
    PRINT MESS2_BIN
    OCT1_BIN:
                ROL   BX,1              ;ROTATE LEFT BY 1
        MOV   AX,BX
        AND   AL,01H            ;TO EXTRACT THE LEAST SIGNIFICANT BIT FROM THE INPUT
ENTERED BY THE USER
        ADD   AL,30H            ;30H = 48 IN DECIMAL
        MOV   DL,AL
        MOV   AH,02H
        INT   21H
        MOV   CH,5
        OCT2_BIN:
            MOV   CL,3
        ROL   BX,CL
        MOV   AL,BL
        AND   AL,07H
        ADD   AL,30H
                MOV  DL,AL
        MOV   AH,2
        INT   21H
        DEC   CH
        JNZ   OCT2_BIN
        JMP EXIT_OCT
```

```
;-------------------------------------------------------------------------------------------------------------------------------
OPT2_BIN:
        CMP JUMP_BIN,'2'
        JZ  BIN_DEC
        JMP OPT3_BIN
;-------------------------------------------------------------------------------------------------------------------------------
BIN_DEC:
        CALL  INPBIN
        PRINT MESS3_BIN                         ;THE ANSWER WILL BE UPTO 5 DIGITS
    MOV   CX,10000
    CALL  DECL1_BIN
    MOV   CX,1000
        CALL  DECL1_BIN
    MOV   CX,100
    CALL  DECL1_BIN
    MOV   CX,10
    CALL  DECL1_BIN
    MOV   CX,1
    CALL  DECL1_BIN
        JMP EXIT_OCT

;-------------------------------------------------------------------------------------------------------------------------------
OPT3_BIN:
        CMP JUMP_BIN,'3'
        JZ  BIN_HEX
        JMP OPT4_BIN
;-------------------------------------------------------------------------------------------------------------------------------
BIN_HEX:
        CALL  INPBIN
        PRINT MESS4_BIN
```

```
    MOV   CX,4
    MOV   AL,0
        HEXL1_BIN:                  ;SINCE HEXADECIMAL USES 4 BITS TO REPRESENT
                ROL   BX,1                  ;ROTATE LEFT BY 1
        RCL   AL,1                  ;ROTATE CARRY LEFT BY 1
        ROL   BX,1
        RCL   AL,1
        ROL   BX,1
        RCL   AL,1
        ROL   BX,1
        RCL   AL,1
        MOV   DL,AL
        AND   DL,0FH                ;EXTRACTING THE LEAST SIGNIFICANT 4 BITS WHICH
CORRESPOND TO THE CURRENT HEXADECIMAL DIGIT
        ADD   DL,30H
        CMP   DL,39H                ;39H = 9 IN DECIMAL
        JBE   HEXL2_BIN
        ADD   DL,7
        HEXL2_BIN:
                MOV   AH,02H
        INT   21H
        LOOP  HEXL1_BIN
        JMP EXIT_OCT

;-------------------------------------------------------------------------------------------------------------------------
-------------------

OPT4_BIN:
        CMP JUMP_BIN,'0'
        JZ EXIT_BIN
        JMP ERROR_BIN

ERROR_BIN:
```

```
        PRINT OPT_ERROR
        JMP START_BIN

EXIT_BIN:
        PRINT EXIT_KEY
        MOV AH,0
        INT 16H                                         ;TERMINATE PROGRAM.
        JMP MAIN_EXIT

;-------------------------------------------------------------------------------------------------------------------------------
OPT2:
        CMP    JUMP,'2'
        JE     OCTAL
        JMP  OPT3
;-------------------------------------------------------------------------------------------------------------------------------
;----------------------OCTAL CONVERSION-----------------------------------------------------------------------------------------
OCTAL:
        START_OCT:
                PRINT MENU_OCT
                MOV AH,0H
                INT 16H
                MOV JUMP_OCT,AL
                CMP JUMP_OCT,'1'
                JZ  OCT_BIN
                JMP OPT2_OCT

;-------------------------------------------------------------------------------------------------------
OCT_BIN:
        PRINT MESS1_OCT_BIN
```

```
        MOV SI,0
        MOV AH,1                                ;FOR INPUT
        MOV CX,0
        INPUT_OCT_BIN:
                INT 21H
        MOV VAR_OCT_BIN[SI],AL                  ;PLACE 1ST DIGIT
        INC CX
        INC SI
        CMP AL,13
        JNE INPUT_OCT_BIN

        PRINT MESS2_OCT_BIN
        MOV DI,0
        MOV AH,2
        DEC CX

        OUTPUT_OCT_BIN:
                MOV BL,VAR_OCT_BIN[DI]          ;MOVE 1ST DIGIT TO BL
        CALL CONVERTOCT_OCT_BIN                 ;PROCEDURE CALLING
        INC DI
        LOOP OUTPUT_OCT_BIN

        JMP EXIT_OCT

;-------------------------------------------------------------------------------------------------------------------------
--------------
OPT2_OCT:
        CMP JUMP_OCT,'2'
        JZ  OCT_DEC
        JMP OPT3_OCT
;-------------------------------------------------------------------------------------------------------
```

```
OCT_DEC:
        CALL CLEAR_RESULT_OCT_DEC                   ;CLEAR OURMEMORY(IN CASE IT HOLDS
PREVIOUS RESULT).
        PRINT MESS1_OCT_DEC
        MOV  AH, 10                         ;CAPTURE OCTAL NUMBER AS STRING
        LEA  DX, VAR_OCT_DEC
        INT  21H

                                    ;CONVERT OCTAL-STRING TO NUMBER.
        LEA  SI, VAR_OCT_DEC+2                      ;CHARS OF THE OCTAL-STRING.
        MOV  BH, [SI-1]             ;SECOND BYTE IS LENGTH.
        CALL OCT2NUMBER_OCT_DEC                     ;NUMBER RETURNS IN AX.

                                    ;CONVERT NUMBER TO DECIMAL-STRING TO DISPLAY.
        LEA  SI, RESULT_OCT_DEC
        CALL NUMBER2STRING_OCT_DEC                          ;STRING RETURNS IN SI
(OURMEMORY).

                                    ;DISPLAY 'IN DECIMAL IS IT:$'
        PRINT MESS2_OCT_DEC
                                    ;DISPLAY NUMBER AS STRING.
        PRINT RESULT_OCT_DEC

        ILLEGAL_OCT_DEC:                            ;JUMP HERE WHEN INVALID CHARACTER
FOUND.
                JMP EXIT_OCT

;-------------------------------------------------------------------------------------------------------------------------
-------------
OPT3_OCT:
        CMP JUMP_OCT,'3'
        JZ  OCT_HEX
```

```
        JMP OPT4_OCT
;-------------------------------------------------------------------------------------------------------------
OCT_HEX:
        PRINT MESS1_OCT_HEX
                                        ; READ OCTAL INPUT NUMBER
        MOV  CL, 3                      ; CONST (TO SHIFT 3X TO THE LEFT) [8086]
        MOV  AH, 01H
        INT  21H
        SUB  AL, 30H                    ; CONVERT ASCII DIGIT TO BINARY
        MOV  BL, AL                     ; ADD FIRSTDIGIT

        MOV  AH, 01H
        INT  21H
        SUB  AL, 30H                    ; CONVERT ASCII DIGIT TO BINARY
        SHL  BL, CL                     ; MAKE ROOM TO ADD THE FOLLOWING DIGIT
                                        ; THIS IS THE 1ST TIME THAT FIRSTDIGIT GETS SHIFTED TO
THE LEFT, SO *8
        OR   BL, AL                     ; ADD SECONDDIGIT

        MOV  AH, 01H
        INT  21H
        SUB  AL, 30H                    ; CONVERT ASCII DIGIT TO BINARY
        SHL  BL, CL                     ; MAKE ROOM TO ADD THE FOLLOWING DIGIT
                                ; THIS IS THE 2ND TIME THAT FIRSTDIGIT GETS SHIFTED TO THE LEFT,
SO *64
                                ; THIS IS THE ONLY TIME THAT SECONDDIGIT GETS SHIFTED TO THE
LEFT, SO *8
        OR   BL, AL                     ; ADD THIRDDIGIT

                                        ; DISPLAY OUTPUT MESSAGE AND HEXADECIMAL NUMBER
        PRINT MESS2_OCT_HEX
```

```
        MOV  BH, 0
        MOV  SI, BX
        MOV  CL, 4                  ; CONST (TO SHIFT 4X TO THE RIGHT) [8086]
        SHR  BX, CL                 ; ONLY KEEP MOST SIGNIFICANT HEX DIGIT
        MOV  DL, HEX_TABLE[BX]      ; LOOKUP HEXADECIMAL DIGIT
        MOV  AH, 02H
        INT  21H
        AND  SI, 15                 ; ONLY KEEP LEAST SIGNIFICANT HEX DIGIT
        MOV  DL, HEX_TABLE[SI]      ; LOOKUP HEXADECIMAL DIGIT
        MOV  AH, 02H
        INT  21H
        JMP EXIT_OCT

;-------------------------------------------------------------------------------------------------------------------------------
OPT4_OCT:
        CMP JUMP_OCT,'0'
        JZ EXIT_OCT
        JMP ERROR_OCTAL

ERROR_OCTAL:
        PRINT OPT_ERROR
        JMP START_OCT

EXIT_OCT:
        PRINT EXIT_KEY
        MOV AH,0
        INT 16H                                 ;TERMINATE PROGRAM.
        JMP MAIN_EXIT

;-------------------------------------------------------------------------------------------------------------------------------
```

```
OPT3:
        CMP    JUMP,'3'
        JE     DECIMAL
        JMP  OPT4
;-------------------------------------------------------------------------------------------------------------------------------
;-----------------------DECIMAL CONVERSION-------------------------------------------------------------------------------------
DECIMAL:
        START_DEC:
                PRINT MENU_DEC
                MOV AH,0H
                INT 16H
                MOV JUMP_DEC,AL
                CMP JUMP_DEC,'1'
                JZ  DEC_BIN
                JMP OPT2_DEC

;-------------------------------------------------------------------------------------------------------------------------------
DEC_BIN:
        PRINT MESS1_DEC_BIN
        MOV VAR1_DEC_BIN,0                      ;INITIALIZES VAR1 VALUE TO 0
        MOV AH,01H                      ;INT TO OBTAIN INPUT
        INT 21H
        SUB AL,30H                      ; ASCII CODE VALUE TO REAL DECIMAL VALUE
CONVERSION (SUBTRACTS 48D)
        MOV NUM_DEC_BIN,AL                      ;INPUT NUMBER FROM AL IS MOVED TO
VARIABLE NUM

        MOV AL,NUM_DEC_BIN
        MOV BL,10                       ;10 IS STORED IN BL
        MUL BL                          ;NUMBER TO CONVERT IS MULTIPLIED BY 10
```

```
        MOV AUX_DEC_BIN,AL                  ;AUX VARIABLE IS ASSIGNED THE RESULT

        MOV VAR1_DEC_BIN,0                  ;WE OBTAIN THE SECOND USER NUMBER
INPUT
        MOV AH,01H
        INT 21H
        SUB AL,30H


        ADD AUX_DEC_BIN,AL                  ;WE ADD AUX TO THE PREVIOUS NUMBER
MULTIPLIED BY 10
        MOV BL,AUX_DEC_BIN                  ;DOESN'T NEED TO BE MULTIPLIED
        MOV NUM_DEC_BIN,BL                  ;RESULT IS STORED IN BL

        PRINT MESS2_DEC_BIN

        MOV SI,6                            ;CYCLES WHERE WE USE LONG DIVISION (DIVIDE BY
2)
        L1_DEC_BIN:                         ;L1  LABEL
                XOR AH,AH                     ;RESETS AH
                MOV AL,NUM_DEC_BIN
                MOV BL,2
                DIV BL
                MOV VAR1_DEC_BIN,AH
                MOV NUM_DEC_BIN,AL

                MOV DL,VAR1_DEC_BIN
                ADD DL,30H

                MOV DEC_BIN_VAR[SI],DL      ;CONCATENATES RESULTS

                CMP NUM_DEC_BIN,1           ;COMPARES NUM WITH 1
```

```
                DEC SI
                JNE L1_DEC_BIN                  ;L1 LOOPS UNTIL IT GOES THROUGH THE ALL
NUMBERS
                JE EXIT_DEC_BIN                 ;EXITS LOOP

                CMP NUM_DEC_BIN,0 ;COMPARES NUM WITH 0
                JNE L1_DEC_BIN
        JE EXIT_DEC_BIN

        EXIT_DEC_BIN:                   ;EXIT LABEL
                MOV DL,NUM_DEC_BIN              ;PRINTS THE CHAIN IN BINARY
                ADD DL,30H
                MOV DEC_BIN_VAR[SI],DL
                PRINT DEC_BIN_VAR
                JMP EXIT_DEC
;-------------------------------------------------------------------------------------------------------------------------------
OPT2_DEC:
        CMP JUMP_DEC,'2'
        JZ  DEC_OCT
        JMP OPT3_DEC
;-------------------------------------------------------------------------------------------------------------------------------
DEC_OCT:
        PRINT MESS1_DEC_OCT

    MOV AH, 10
    LEA DX, MESS4_DEC_OCT
    INT 21H

    MOV SI, OFFSET MESS4_DEC_OCT + 2
    MOV CL, BYTE PTR [SI-1]
    MOV CH, 00H
```

```
    SUBTRACT_DEC_OCT :
    MOV AL, BYTE PTR [SI]
    CMP AL, 30H
    JNB CONT1_DEC_OCT
    PRINT MESS2_DEC_OCT
    JMP EXIT_DEC

    CONT1_DEC_OCT :
    CMP AL, 3AH
    JB CONT2_DEC_OCT
    PRINT MESS2_DEC_OCT
    JMP EXIT_DEC

    CONT2_DEC_OCT :
    SUB AL, 30H
    MOV BYTE PTR [SI], AL
        INC SI
    LOOP SUBTRACT_DEC_OCT

MOV SI, OFFSET MESS4_DEC_OCT + 2
    MOV CL, BYTE PTR [SI-1]
    MOV CH, 00H
    MOV AX, 0000H

    CALC_DEC_OCT:
    MUL MULTIPLIER_DEC_OCT
    MOV BL, BYTE PTR [SI]
    MOV BH, 00H
    ADD AX, BX
    INC SI
    LOOP CALC_DEC_OCT
```

```
    MOV SI, OFFSET MESS4_DEC_OCT + 2
    MOV BX, AX
    MOV DX, 0000H
    MOV AX, 8000H
        CONVERT_DEC_OCT:
        MOV CX, 0000H
        CONV_DEC_OCT:
        CMP BX, AX
        JB CONT3_DEC_OCT
        SUB BX, AX
        INC CX
        JMP CONV_DEC_OCT
        CONT3_DEC_OCT :
        ADD CL, 30H
        MOV BYTE PTR [SI], CL
        INC SI
        MOV CX, 0008H
        DIV CX
        CMP AX, 0000H
        JNZ CONVERT_DEC_OCT

    MOV BYTE PTR [SI], '$'
    PRINT MESS3_DEC_OCT
    PRINT MESS4_DEC_OCT+2
        JMP EXIT_DEC
;-------------------------------------------------------------------------------------------------------------------------
----------------------
OPT3_DEC:
        CMP JUMP_DEC,'3'
        JZ  DEC_HEX
```

```
        JMP OPT4_DEC
;-------------------------------------------------------------------------------------------------------------
DEC_HEX:
        PRINT MESS1_DEC_HEX
        MOV CX, 10

        INPUT_DEC_HEX:
                MOV AH, 1
                INT 21H
                CMP AL, 13
                JE INPUT_END_DEC_HEX
                SUB AL, 48
                MOV AH, 0
                PUSH AX
                MOV AX, BX
                MUL CX
                MOV BX, AX
                POP AX
                ADD BX, AX
                JMP INPUT_DEC_HEX

        INPUT_END_DEC_HEX:
                MOV AX, BX
                MOV CX, 16
                MOV BX, 0
        CONVERSION_DEC_HEX:
                DIV CX
                PUSH DX
                MOV DX, 0
                INC BL
                CMP AX, 0
```

```
        JNE CONVERSION_DEC_HEX
        PRINT MESS2_DEC_HEX

OUTPUT_START_DEC_HEX:
        POP AX
        CMP AL, 9
        JG OUTPUT_HEX_DEC_HEX

OUTPUT_DEC_HEX:
        ADD AL, 48
        MOV AH, 2
        MOV DL, AL
        INT 21H
        INC BH
        CMP BH, BL
        JNE OUTPUT_START_DEC_HEX
        JMP OUTPUT_END_DEC_HEX

OUTPUT_HEX_DEC_HEX:
        ADD AL, 55
        MOV AH, 2
        MOV DL, AL
        INT 21H
        INC BH
        CMP BH, BL
        JNE OUTPUT_START_DEC_HEX

OUTPUT_END_DEC_HEX:
        JMP EXIT_DEC
```

```
;-------------------------------------------------------------------------------------------------------------------------------
OPT4_DEC:
        CMP JUMP_DEC,'0'
        JZ EXIT_DEC
        JMP ERROR_DECIMAL

ERROR_DECIMAL:
        PRINT OPT_ERROR
        JMP START_DEC

EXIT_DEC:
        PRINT EXIT_KEY
        MOV AH,0
        INT 16H                                 ;TERMINATE PROGRAM.
        JMP MAIN_EXIT

;-------------------------------------------------------------------------------------------------------------------------------
OPT4:
        CMP     JUMP,'4'
        JE      HEXADECIMAL
        JMP  MAIN_EXIT
;-------------------------------------------------------------------------------------------------------------------------------
;--------------------HEXADECIMAL CONVERSION-----------------------------------------------------------------------------------
HEXADECIMAL:
        START_HEX:
                PRINT MENU_HEX
                MOV AH,0H
                INT 16H
```

```
                MOV JUMP_HEX,AL
                CMP JUMP_HEX,'1'
                JZ  HEX_BIN
                JMP OPT2_HEX
;-------------------------------------------------------------------------------------------------------------
HEX_BIN:
        MOV AX,0
        MOV BX,0
        MOV CX,0
        MOV DX,0

        PRINT MESS1_HEX_BIN
        PRINT MESS2_HEX_BIN

        MOV CX,-1                       ; ASSIGN -1 INTO CX TO ACT AS COUNTER
        INPUT_HEX_BIN:
                MOV AH, 00H
                INT 16H
                CMP AH, 1CH
                JE EXIT_HEX_BIN


        NUMBER_HEX_BIN:
                CMP AL, '0'
                JB INPUT_HEX_BIN
                CMP AL, '9'
                JA UPPERCASE_HEX_BIN
                SUB AL, 30H
                CALL PROCESS_HEX_BIN
                JMP INPUT_HEX_BIN
```

```
UPPERCASE_HEX_BIN:
        CMP AL, 'A'
        JB INPUT_HEX_BIN
        CMP AL, 'F'
        JA LOWERCASE_HEX_BIN
        SUB AL, 37H
        CALL PROCESS_HEX_BIN
        JMP INPUT_HEX_BIN

LOWERCASE_HEX_BIN:
        CMP AL, 'a'
        JB INPUT_HEX_BIN
        CMP AL, 'f'
        JA INPUT_HEX_BIN
        SUB AL, 57H
        CALL PROCESS_HEX_BIN
        JMP INPUT_HEX_BIN
        LOOP INPUT_HEX_BIN

PROCESS_HEX_BIN:
        MOV CH, 4
        MOV CL, 3
        MOV BL, AL

CONVERT_HEX_BIN:
        MOV AL, BL
        ROR AL, CL
        AND AL, 01
        ADD AL, 30H

        MOV AH, 02H
```

```
                MOV DL, AL
                INT 21H

                DEC CL
                DEC CH
                JNZ CONVERT_HEX_BIN

                MOV DL, 20H
                INT 21H
        RET

        EXIT_HEX_BIN:
                JMP EXIT_HEX
;-------------------------------------------------------------------------------------------------------------------------------
OPT2_HEX:
        CMP JUMP_HEX,'2'
        JZ  HEX_DEC
        JMP OPT3_HEX
;-------------------------------------------------------------------------------------------------------------
HEX_DEC:
        CALL CLEAR_RESULT_HEX_DEC
        PRINT MESS1_HEX_DEC
        MOV  AH, 10                             ;CAPTURE HEX NUMBER AS STRING
        LEA  DX, VAR_HEX_DEC
        INT  21H

                                        ;CONVERT HEX-STRING TO NUMBER.
        LEA  SI, VAR_HEX_DEC+2                  ;CHARS OF THE HEX-STRING.
        MOV  BH, [SI-1]                 ;SECOND BYTE IS LENGTH.
        CALL HEX2NUMBER_HEX_DEC                 ;NUMBER RETURNS IN AX.
```

```
                                        ;CONVERT NUMBER TO DECIMAL-STRING TO DISPLAY.
        LEA  SI, RESULT_HEX_DEC
        CALL NUMBER2STRING_HEX_DEC                  ;STRING RETURNS IN SI
(OURMEMORY).

                                        ;DISPLAY 'IN DECIMAL IS IT:$'
        PRINT MESS2_HEX_DEC

                                        ;DISPLAY NUMBER AS STRING.
        PRINT RESULT_HEX_DEC

        ILLEGAL_HEX_DEC:                    ;JUMP HERE WHEN INVALID CHARACTER
FOUND.
                JMP EXIT_HEX

;-------------------------------------------------------------------------------------------------------------------
---------
OPT3_HEX:
        CMP JUMP_HEX,'0'
        JZ EXIT_HEX
        JMP ERROR_HEXADECIMAL

ERROR_HEXADECIMAL:
        PRINT OPT_ERROR
        JMP START_HEX

EXIT_HEX:
        PRINT EXIT_KEY
        MOV AH,0
        INT 16H                             ;TERMINATE PROGRAM.
        JMP MAIN_EXIT
```

```
;-------------------------------------------------------------------------------------------------------------------------------------------------------------

;----------------------------MAIN EXIT-----------------------------------------------------------------------------------------------------------------------

MAIN_EXIT:
        MOV AH, 4CH
        INT 21H
        MAIN ENDP

;-------------------------------------------------------------------------------------------------------------------------------------------------------------

;--------------------------PROCEDURES-----------------------------------------------------------------------------------------------------------------------

;------BINARY PROCEDURES-----------------------------------------------------------------------------------------------------------------------------------

INPBIN PROC
      PRINT MESS1_BIN                          ;FOR INPUT OF BINARY NUMBER
      MOV   BX,0
         BINL1_BIN:
              MOV   AH,01H                     ;FIRST DIGIT INPUT
        INT   21H
        CMP   AL,13                            ;COMPARE AND IF ZERO JUMP TO BINL2_BIN
        JZ    BINL2_BIN
        AND   AL,01H                           ;TO EXTRACT THE LEAST SIGNIFICANT BIT FROM THE INPUT ENTERED BY THE USER
        SHL   BX,1
        OR    BL,AL                            ;SETS THE LEAST SIGNIFICANT BIT OF THE BX REGISTER TO THE VALUE IN THE AL REGISTER
        JMP   BINL1_BIN
         BINL2_BIN:
              RET
INPBIN  ENDP
```

```
;-------------------------------------------------


DECL1_BIN PROC
        MOV   AX,BX
        MOV   DX,0
        DIV   CX
        MOV   BX,DX
        MOV   DL,AL
        ADD   DL,30H
        MOV   AH,2
        INT   21H
        RET
DECL1_BIN ENDP


;-------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------

;------OCTAL PROCEDURES-------------------------------------------------------------------------------------------------------
----------------

CONVERTOCT_OCT_BIN PROC
          SUB  BL,48                          ;FOR NUMBER
          SHL  BL, 1                          ;MOVES THE BITS OF THE OCTAL DIGIT TO THE
LEAST SIGNIFICANT BITS OF THE TWO REGISTERS,
          RCL  BH, 1                          ;WITH THE LEAST SIGNIFICANT BIT OF BL BECOMING
THE LEAST SIGNIFICANT BIT OF BH
          SHL  BL, 1
          RCL  BH, 1
          SHL  BL, 1
          RCL  BH, 1
          SHL  BL, 1
          RCL  BH, 1
          SHL  BL, 1
```

```
        RCL  BH, 1

        MOV DH,0
        CONV_OCT_BIN:
                SHL BL,1
                JC PRINT1_OCT_BIN
                JMP PRINT0_OCT_BIN

        PRINT0_OCT_BIN:
                MOV DL,'0'
                INT 21H
                INC DH
                CMP DH,3
                JE EXIT_OCT_BIN
                JMP CONV_OCT_BIN

        PRINT1_OCT_BIN:
                MOV DL,'1'
                INT 21H
                INC DH
                CMP DH,3
                JE EXIT_OCT_BIN
                JMP CONV_OCT_BIN

        EXIT_OCT_BIN:
                RET

CONVERTOCT_OCT_BIN ENDP

;-------------------------------------------------------
CLEAR_RESULT_OCT_DEC PROC
```

```
        LEA  SI, RESULT_OCT_DEC
        MOV  AL, '$'
        MOV  CX, 10
CLEARING_OCT_DEC:
        MOV  [SI], AL
        INC  SI
        LOOP CLEARING_OCT_DEC

        RET
CLEAR_RESULT_OCT_DEC ENDP

OCT2NUMBER_OCT_DEC PROC
        MOV  AX, 0      ;THE NUMBER.                ;INPUT  : BH = STRING LENGTH (1..6).
                                                    ;SI = OFFSET OCTAL-STRING
                                                    ;OUTPUT : AX = NUMBER.
        CICLO_OCT_DEC:
                SHL  AL, 1                  ;SHL  AX, 3     ;SHIFT LEFT LOWER 3 BITS.
                RCL  AH, 1
                SHL  AL, 1
                RCL  AH, 1
                SHL  AL, 1
                RCL  AH, 1

        MOV  BL, [ SI ]             ;GET ONE HEX CHAR FROM STRING.
        CALL VALIDATE_OCT_DEC
        SUB  BL, 48                 ;CONVERT DIGIT TO NUMBER.
        JMP  CONTINUE_OCT_DEC

        CONTINUE_OCT_DEC:
                OR   AL, BL                 ;CLEAR UPPER 4 BITS.
                INC  SI                 ;NEXT HEX CHAR.
```

```
            DEC  BH                 ;BH == 0 : FINISH.
            JNZ  CICLO_OCT_DEC          ;BH != 0 : REPEAT.
        FIN_OCT_DEC:
            RET
OCT2NUMBER_OCT_DEC ENDP

;INPUT : BL = HEX CHAR TO VALIDATE.

VALIDATE_OCT_DEC PROC
        CMP BL, '0'
        JB  ERROR_OCT_DEC           ;IF BL < '0'
        CMP BL, '7'
        JA  ERROR_OCT_DEC           ;IF BL > 'F'
        CMP BL, '0'
        JAE OK_OCT_DEC              ;IF BL <= '9'
        CMP BL, '7'
        JBE OK_OCT_DEC              ;IF BL >= 'A'
        ERROR_OCT_DEC:
            POP  AX                 ;REMOVE CALL VALIDATE.
            POP  AX                  ;REMOVE CALL HEX2NUMBER.
                                    ;DISPLAY 'ILLEGAL CHARACTER- ENTER 0-9 OR A-F$'
        PRINT MESS3_OCT_DEC
        JMP  ILLEGAL_OCT_DEC            ;GO TO 'DO YOU WANT TO DO IT AGAIN (Y/N)?$'
        OK_OCT_DEC:
            RET
VALIDATE_OCT_DEC ENDP


NUMBER2STRING_OCT_DEC PROC
        MOV  BX, 10                 ;DIGITS ARE EXTRACTED DIVIDING BY 10.
        MOV  CX, 0                  ;COUNTER FOR EXTRACTED DIGITS.
```

```
        CYCLE1_OCT_DEC:
                MOV  DX, 0                      ;NECESSARY TO DIVIDE BY BX.
                DIV  BX                 ;DX:AX / 10 = AX:QUOTIENT DX:REMAINDER.
                PUSH DX                         ;PRESERVE DIGIT EXTRACTED FOR LATER.
                INC  CX                 ;INCREASE COUNTER FOR EVERY DIGIT EXTRACTED.
                CMP  AX, 0                      ;IF NUMBER IS
                JNE  CYCLE1_OCT_DEC                     ;NOT ZERO, LOOP.
                                                ;NOW RETRIEVE PUSHED DIGITS.
        LEA  SI, RESULT_OCT_DEC
        CYCLE2_OCT_DEC:
                POP  DX
                ADD  DL, 48                     ;CONVERT DIGIT TO CHARACTER.
                MOV  [ SI ], DL
                INC  SI
                LOOP CYCLE2_OCT_DEC


        RET
NUMBER2STRING_OCT_DEC ENDP

;-------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------

;-----HEXADECIMAL PROCEDURES-----------------------------------------------------------------------------------------
-------------------

CLEAR_RESULT_HEX_DEC PROC               ;CLEAR OURMEMORY VALUE
        LEA  SI, RESULT_HEX_DEC
        MOV  AL, '$'
        MOV  CX, 5
        CLEARING_HEX_DEC:
                MOV  [SI], AL
                INC  SI
                LOOP CLEARING_HEX_DEC
```

```
        RET
CLEAR_RESULT_HEX_DEC ENDP
                                        ;INPUT  : BH = STRING LENGTH (1..4).
                                        ;SI = OFFSET HEX-STRING.
                                        ;OUTPUT : AX = NUMBER.
HEX2NUMBER_HEX_DEC PROC
        MOV  AX, 0      ;THE NUMBER.
        CICLO_HEX_DEC:
                SHL  AL, 1                      ;     SHL  AX, 4      ;SHIFT LEFT LOWER 4 BITS.
                RCL  AH, 1                      ;SHIFT LEFT AL AND AH MANUALLY 4 TIMES TO
SIMULATE SHL AX,4.
                SHL  AL, 1
                RCL  AH, 1
                SHL  AL, 1
                RCL  AH, 1
                SHL  AL, 1
                RCL  AH, 1

        MOV  BL, [ SI ]                 ;GET ONE HEX CHAR FROM STRING.
        CALL VALIDATE_HEX_DEC

        CMP  BL, 'A'                    ;BL = 'A'..'F' : LETTER.
        JAE  LETTERAF_HEX_DEC                   ;BL = '0'..'9' : DIGIT.
                                                ;CHARISDIGIT09.
        SUB  BL, 48                     ;CONVERT DIGIT TO NUMBER.
        JMP  CONTINUE_HEX_DEC
        LETTERAF_HEX_DEC:
                SUB  BL, 55             ;CONVERT LETTER TO NUMBER.
        CONTINUE_HEX_DEC:
                OR   AL, BL                     ;CLEAR UPPER 4 BITS.
                INC  SI                         ;NEXT HEX CHAR.
```

```
                DEC  BH                        ;BH == 0 : FINISH.
                JNZ  CICLO_HEX_DEC                      ;BH != 0 : REPEAT.
        FIN_HEX_DEC:
                RET
HEX2NUMBER_HEX_DEC ENDP

;INPUT : BL = HEX CHAR TO VALIDATE.

VALIDATE_HEX_DEC PROC
        CMP BL, '0'
        JB  ERROR_HEX_DEC           ;IF BL < '0'
        CMP BL, 'F'
        JA  ERROR_HEX_DEC           ;IF BL > 'F'
        CMP BL, '9'
        JBE OK_HEX_DEC              ;IF BL <= '9'
        CMP BL, 'A'
        JAE OK_HEX_DEC              ;IF BL >= 'A'
        ERROR_HEX_DEC:
                POP  AX                     ;REMOVE CALL VALIDATE.
                POP  AX                     ;REMOVE CALL HEX2NUMBER.
                                            ;DISPLAY 'ILLEGAL CHARACTER- ENTER 0-9 OR A-F$'
        PRINT MESS3_HEX_DEC
                JMP  ILLEGAL_HEX_DEC                ;GO TO 'DO YOU WANT TO DO IT AGAIN
(Y/N)?$'
        OK_HEX_DEC:
                RET
VALIDATE_HEX_DEC ENDP



NUMBER2STRING_HEX_DEC PROC
        MOV  BX, 10                                 ;DIGITS ARE EXTRACTED DIVIDING BY 10.
```

```
        MOV  CX, 0                          ;COUNTER FOR EXTRACTED DIGITS.
        CYCLE1_HEX_DEC:
                MOV  DX, 0                  ;NECESSARY TO DIVIDE BY BX.
                DIV  BX                 ;DX:AX / 10 = AX:QUOTIENT DX:REMAINDER.
                PUSH DX                     ;PRESERVE DIGIT EXTRACTED FOR LATER.
                INC  CX                 ;INCREASE COUNTER FOR EVERY DIGIT EXTRACTED.
                CMP  AX, 0                  ;IF NUMBER IS
                JNE  CYCLE1_HEX_DEC                 ;NOT ZERO, LOOP.
                                            ;NOW RETRIEVE PUSHED DIGITS.
                LEA  SI, RESULT_HEX_DEC
        CYCLE2_HEX_DEC:
                POP  DX
                ADD  DL, 48             ;CONVERT DIGIT TO CHARACTER.
                MOV  [ SI ], DL
                INC  SI
                LOOP CYCLE2_HEX_DEC

        RET
NUMBER2STRING_HEX_DEC ENDP

;-------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------

END MAIN
```