```
; Q1: SUM OF ARRAY ELEMENTS
.MODEL SMALL
.STACK 100H
.DATA
      ARR DB 5,3,1,7,9,2,6,4,8,10   ;ARRAY ELEMENTS
      LEN DW $-ARR                  ;LENGTH OF ARRAY
      SUM DW ?                ;SUM
.CODE
START:      MOV AX, @DATA
      MOV DS, AX
      MOV SI, 0
      MOV AX, 0
      MOV CX, LEN
REPEAT: MOV BL, ARR[SI]
      MOV BH, 0
      ADD AX, BX
      INC SI
      DEC CX
      JNZ REPEAT
      MOV SUM, AX
      MOV AH, 4CH
      INT 21H
      END START
.END
```

```
; Q2. AVERAGE OF ARRAY ELEMENTS
.MODEL SMALL
.STACK 100H
.DATA
      ARR DB 7,8,6,5,7,3,6    ;ARRAY ELEMENTS
      LEN DW $-ARR            ;LENGTH OF ARRAY
      AVG DW ?          ;AVERAGE
.CODE
START:      MOV AX, @DATA
      MOV DS, AX
      MOV SI, 0         ;LOAD OFFSET TO SI
      MOV AX, 0         ;INITIALIZE SUM=0
      MOV CX, LEN       ;LOOP VARIABLE
REPEAT: MOV BL, ARR[SI]       ;8 BIT NUMBER
      MOV BH, 0         ;FIRST 8 BITS=0
      ADD AX, BX        ;ADDITION
      INC SI                  ;INCREMENT OF OFFSET ADDRESS
      DEC CX                  ;DECREMENT OF COUNT
      JNZ REPEAT        ;IF ZF=0 REPEAT
      MOV DX, LEN
      DIV DL                  ;DIVIDING FOR AVERAGE
      MOV AVG, AX       ;STORING AVERAGE
      MOV AH, 4CH
      INT 21H
      END START
.END
```

```
; Q3. MINIMUM AND MAXIMUM OF ARRAY ELEMENTS
.MODEL SMALL
.STACK 100H
.DATA
      ARR DB 7,8,6,5,7,3,6    ;ARRAY ELEMENTS
      LEN DW $-ARR            ;LENGTH OF ARRAY
      MIN DB ?
      MAX DB ?
.CODE
START:      MOV AX, @DATA
      MOV DS, AX
      MOV SI, 0         ;LOAD OFFSET TO SI
      MOV AL, ARR[SI]         ;INITIALIZE MIN=FIRST ELEMENT
      MOV MIN,AL        ;INITIALIZE MIN
      MOV MAX,AL        ;INITIALIZE MAX
      MOV CX, LEN       ;LOOP VARIABLE
      INC SI
      DEC CX
REPEAT: MOV BL, ARR[SI]       ;8 BIT NUMBER
      INC SI                  ;INCREMENT OF OFFSET ADDRESS
      DEC CX                  ;DECREMENT OF COUNT
      CMP MIN, AL       ;COMPARE MIN and CURRENT NO.
      JLE SKIP
      MOV MIN, AL       ;[MIN]=[AL]
SKIP:       CMP MAX, AL       ;COMPARE MAX AND CUURENT NO.
      JGE NEXT
      MOV MAX, AL       ;[MAX]=[AL]
NEXT: JNZ REPEAT        ;IF ZF=0 REPEAT
      MOV AH, 4CH
      INT 21H
      END START
.END
```

```
;SWAP 2 NUMBERS
.MODEL SMALL
.STACK 100H
.DATA
      DATA1 DB 52H            ;FIRST NUMBER
      DATA2 DB 29H            ;SECOND NUMBER
.CODE
START:      MOV AX, @DATA
      MOV DS, AX
      MOV AL, DATA1           ;COPYING FIRST NUMBER
      MOV AH, DATA2           ;COPYING SECOND NUMBER
      MOV DATA1, AH           ;COPYING 2ND TO 1ST LOCATION
      MOV DATA2, AL           ;COPYING 1ST TO 2ND LOCATION
      MOV AH, 4CH
      INT 21H
      END START
.END
```

```
; Converting BCD to Hexadecimal
.MODEL SMALL
.STACK 100H
.DATA
      DATA1 DB 99H            ;BCD NUMBER IN HEX FORM
      HEX   DB ?
.CODE
START:      MOV AX, @DATA
      MOV DS, AX
      MOV AL, DATA1           ;COPYING NUMBER TO AL
      MOV BL, DATA1           ;COPYING NUMBER TO BL
      AND AL, 0F0H            ;MASKING LAST 4 BITS
      AND BL, 0FH       ;MASKING FIRST 4 BITS
      MOV CL, 04H       ;COUNT ROTATION
      ROR AL, CL        ;ROTATING RIGHT BY 4
      MOV DL, 0AH       ;STORING DL=10 IN DECIMAL
      MUL DL                  ;MULTIPLICATION OF AL AND DL
      ADD AL, BL        ;ADDING AL AND BL
      MOV HEX, AL       ;STORING HEXADECIMAL VALUE
      MOV AH, 4CH
      INT 21H
      END START
.END
```

```
; Adding 2 4 digits BCD numbers
.MODEL SMALL
.STACK 100H
.DATA
      DATA1 DB 45H            ;FIRST NUMBER
      DATA2 DB 56H            ;SECOND NUMBER
      DATA3 DB ?        ;NEW NUMBER AFTER ADDITION
      CARRY DB ?        ;CARRY AFTER ADDITION
.CODE
START:      MOV AX, @DATA
      MOV DS, AX
      MOV AL, DATA1           ;STORING FIRST NUMBER
      MOV BL, DATA2           ;STORING SECOND NUMBER
      ADD AL, BL        ;ADDITION
      DAA               ;DECIMAL ADJUSTMENT
      MOV DATA3, AL           ;STORING ANSWER
      MOV AL, 00H       ;AL=0
      ADC AL, AL        ;ADDING CARRY TO AL
      MOV CARRY, AL           ;STORING CARRY
      MOV AH, 4CH
      INT 21H
      END START
.END
```

```
; Sum of 2 digit Hexadecimal No.
.MODEL SMALL
.STACK 100H
.DATA
      DATA1 DB 99H            ;STORING THE HEX NUMBER
      SUM   DB ?        ;SUM OF DIGITS
.CODE
START:      MOV AX, @DATA
      MOV DS, AX
      MOV AL, DATA1           ;COPYING THE NUMBER
      MOV AH, DATA1           ;COPYING THE NUMBER
      AND AL, 0FH       ;MASKING FIRST 4 BITS
      AND AH, 0F0H            ;MASKING LAST 4 BITS
      MOV CL, 04H       ;COUNT FOR ROTATION
      ROR AH, CL        ;ROTATING RIGHT BY 4 BITS
      ADD AL, AH        ;ADDING BOTH THE DIGITS
      MOV SUM,AL        ;STORING SUM
      MOV AH, 4CH
      INT 21H
      END START
.END
```

```
; Binary to Gray Code Convertor
.MODEL SMALL
.STACK 100H
.DATA
      BIN  DB 07H       ;BINARY NUMBER
      GRAY DB ?         ;GRAY CODE
.CODE
START:      MOV AX, @DATA
      MOV DS, AX
      MOV AL, BIN       ;COPYING BINARY NUMBER
      MOV BL, BIN       ;COPYING BINARY NUMBER
      SHR BL, 01        ;SHIFTING BY 1 AND MSB=0
      XOR AL, BL        ;XOR TO GET GRAY CODE
      MOV GRAY,AL       ;STORING GRAY CODE
      MOV AH, 4CH
      INT 21H
      END START
.END
```

```
;Count the number of set bits
.MODEL SMALL
.STACK 100H
.DATA
      DATA1 DB 99H            ;ORIGINAL NUMBER
      SET   DB ?        ;NUMBER OF SET BITS
.CODE
START:      MOV AX, @DATA
      MOV DS, AX
      MOV AL, DATA1           ;COOYING FIRST DATA
      MOV BL, 00        ;INITIALIZING TO 0
      MOV CX, 0008H           ;COUNT=8 FOR LOOP
REPEAT:     RCR AL, 01        ;ROTATING RIGHT THROUGH CARRY
      JNC SKIP          ;IF CARRY=0 SKIP NEXT STEP
      INC BL                  ;INCREMENT COUNT
SKIP: LOOP REPEAT       ;WHILE CX>0 LOOP CONTINUES
      MOV SET, BL       ;STORING NO. OF SET BIT
      MOV AH, 4CH
      INT 21H
      END START
.END
```