7. 80x86 Mikroişlemci Programlama Teknikleri

```
          PAGE    60,132
TITLE     TABSRCH Table Search
;-------------------------------------------------
STACKSG   SEGMENT PARA STACK 'STACK'
          DW      32 DUP(?)
STACKSG   ENDS
;-------------------------------------------------
DATASG    SEGMENT PARA 'DATA'
STOKNIN   DW      '23'
STOKTAB   DB      '05','Excavators'
          DB      '08','Lifters   '
          DB      '09','Presses   '
          DB      '12','Valves    '
          DB      '23','Processors'
          DB      '27','Pumps     '

DESCRN    DB      10 DUP(?)
DATASG    ENDS
;-------------------------------------------------
CODESG    SEGMENT PARA 'CODE'
BEGIN     PROC    FAR
          ASSUME  CS:CODESG,DS:DATASG,SS:STACKSG,ES:DATASG
          PUSH    DS
          SUB     AX,AX
          PUSH    AX
          MOV     AX,DATASG
          MOV     DS,AX
          MOV     ES,AX
          CALL    C10SRCH
          RET
BEGIN     ENDP
; Search  routine:
C10SRCH   PROC
          MOV     CX,06
          MOV     AX,STOKNIN
          XCHG    AL,AH
          LEA     SI,STOKTAB
C20:
          CMP     AX,[SI]
          JE      C30
          ADD     SI,12
          LOOP    C20
          CALL    R10ERR
          RET
```

```
C30:
          MOV     CX,05
          LEA     DI,DESCRN
          INC     SI
          INC     SI
          REP MOVSW
          RET
C10SRCH   ENDP
;         Display error message:
R10ERR    PROC
          RET
R10ERR    ENDP
CODESG    ENDS
          END     BEGIN
```

```
          PAGE    60,132
TITLE     ASCADD  Adds ASCII numbers
;-------------------------------------------------
STACKSG   SEGMENT PARA STACK 'STACK'
          DW      32 DUP(?)
STACKSG   ENDS
;-------------------------------------------------
DATASG    SEGMENT PARA 'DATA'
ASC1      DB      '578'
ASC2      DB      '694'
ASC3      DB      '0000'
DATASG    ENDS
;-------------------------------------------------
CODESG    SEGMENT PARA 'CODE'
BEGIN     PROC    FAR
          ASSUME  CS:CODESG,DS:DATASG,SS:STACKSG
          PUSH    DS
          SUB     AX,AX
          PUSH    AX
          MOV     AX,DATASG
          MOV     DS,AX
          CALL    B10ADD
          RET
BEGIN     ENDP
;         Add ASCII numbers:
;         -----------------
B10ADD    PROC
          CLC
          LEA     SI,ASC1+2
          LEA     DI,ASC2+2
          LEA     BX,ASC3+3
          MOV     CX,03
B20:
          MOV     AH,00
          MOV     AL,[SI]
          ADC     AL,[DI]
          AAA
          MOV     [BX],AL
          DEC     SI
          DEC     DI
          DEC     BX
          LOOP    B20
          MOV     [BX],AH
          RET
B10ADD    ENDP
;-------------------------------------------------
CODESG    ENDS
          END     BEGIN
```

```
          PAGE    60,132
TITLE     ASCDIV  Divedes ASCII numbers
;-------------------------------------------------
STACKSG   SEGMENT PARA STACK 'STACK'
          DW      32 DUP(?)
STACKSG   ENDS
;-------------------------------------------------
DATASG    SEGMENT PARA 'DATA'
DIVDND    DB      '3698'
DIVSOR    DB      '4'
QUOTNT    DB      4 dup(0)
DATASG    ENDS
;-------------------------------------------------
CODESG    SEGMENT PARA 'CODE'
BEGIN     PROC    FAR
          ASSUME  CS:CODESG,DS:DATASG,SS:STACKSG
          PUSH    DS
          SUB     AX,AX
          PUSH    AX
          MOV     AX,DATASG
          MOV     DS,AX
          CALL    D10DIV
          RET
BEGIN     ENDP
;         Divide ASCII numbers:
;         --------------------
D10DIV    PROC
          MOV     CX,04
          SUB     AH,AH
          AND     DIVSOR,0FH
          LEA     SI,DIVDND
          LEA     DI,QUOTNT
D20:
          MOV     AL,[SI]
          AND     AL,0FH
          AAD
          DIV     DIVSOR
          MOV     [DI],AL
          INC     DI
          INC     SI
          LOOP    D20
          RET
D10DIV    ENDP
;-------------------------------------------------
CODESG    ENDS
          END     BEGIN
```

```
          PAGE    60,132
TITLE     ASCMUL  Multiplies ASCII numbers
;-------------------------------------------------
STACKSG   SEGMENT PARA STACK 'STACK'
          DW      32 DUP(?)
STACKSG   ENDS
;-------------------------------------------------
DATASG    SEGMENT PARA 'DATA'
MULTCND   DB      '3783'
MULTPLR   DB      '5'
PRODUCT   DB      5 dup(0)
DATASG    ENDS
;-------------------------------------------------
CODESG    SEGMENT PARA 'CODE'
BEGIN     PROC    FAR
          ASSUME  CS:CODESG,DS:DATASG,SS:STACKSG
          PUSH    DS
          SUB     AX,AX
          PUSH    AX
          MOV     AX,DATASG
          MOV     DS,AX
          CALL    C10MULT
          RET
BEGIN     ENDP
;         Multiply ASCII numbers:
;         ----------------------
C10MULT   PROC
          MOV     CX,04
          LEA     SI,MULTCND+3
          LEA     DI,PRODUCT+4
          AND     MULTPLR,0FH
C20:
          MOV     AL,[SI]
          AND     AL,0FH
          MUL     MULTPLR
          AAM
          ADD     AL,[DI]
          AAA
          MOV     [DI],AL
          DEC     DI
          MOV     [DI],AH
          DEC     SI
          LOOP    C20
          RET
C10MULT   ENDP
;-------------------------------------------------
CODESG    ENDS
          END     BEGIN
```

```
         PAGE    60,132
TITLE    BCDARITH Converts ASCII numbers to BCD and add
;-------------------------------------------------
STACKSG  SEGMENT PARA STACK 'STACK'
         DW      32 DUP(?)
STACKSG  ENDS
;-------------------------------------------------
DATASG   SEGMENT PARA 'DATA'
ASC1     DB      '057836'
ASC2     DB      '069427'
BCD1     DB      '000'
BCD2     DB      '000'
BCD3     DB      4 DUP(0)
DATASG   ENDS
;-------------------------------------------------
CODESG   SEGMENT PARA 'CODE'
BEGIN    PROC    FAR
         ASSUME  CS:CODESG,DS:DATASG,SS:STACKSG
         PUSH    DS
         SUB     AX,AX
         PUSH    AX
         MOV     AX,DATASG
         MOV     DS,AX
         LEA     SI,ASC1+4
         LEA     DI,BCD1+2
         CALL    B10CONV
         LEA     SI,ASC2+4
         LEA     DI,BCD2+2
         CALL    B10CONV
         CALL    C10ADD
         RET
BEGIN    ENDP
;        Convert ASCII to BCD:
;        --------------------
B10CONV  PROC
         MOV     CL,04
         MOV     DX,03
B20:
         MOV     AX,[SI]
         XCHG    AH,AL
         SHL     AL,CL
         SHL     AX,CL
         MOV     [DI],AH
         DEC     SI
         DEC     SI
         DEC     DI
         DEC     DX
         JNZ     B20
         RET
B10CONV  ENDP
;        Add BCD numbers:
```

```
;         -----------------
C10ADD    PROC
          XOR     AH,AH
          LEA     SI,BCD1+2
          LEA     DI,BCD2+2
          LEA     BX,BCD3+3
          MOV     CX,03
          CLC
C20:
          MOV     AL,[SI]
          ADC     AL,[DI]
          DAA
          MOV     [BX],AL
          DEC     SI
          DEC     DI
          DEC     BX
          LOOP    C20
          RET
C10ADD    ENDP
;-------------------------------------------------
CODESG    ENDS
          END     BEGIN
```

```
          PAGE    60,132
TITLE     CALLPROC  Calling Procedures
;-------------------------------------------------
STACKSG   SEGMENT PARA STACK 'STACK'
          DW      32 DUP(?)
STACKSG   ENDS
;-------------------------------------------------
CODESG    SEGMENT PARA 'CODE'
BEGIN     PROC    FAR
          ASSUME  SS:STACKSG,CS:CODESG
          PUSH    DS              ;Store DS on stack.
          SUB     AX,AX           ;Set AX to zero.
          PUSH    AX              ;Store zero on stack.
          CALL    B10
          RET
BEGIN     ENDP
;-------------------------------------------------
B10       PROC
          CALL    C10
          RET
B10       ENDP
;-------------------------------------------------
C10       PROC
          RET
C10       ENDP
;-------------------------------------------------
CODESG    ENDS
          END     BEGIN
```

```
          PAGE    65,132
TITLE     CASE    Change lowercase to uppercase
;-------------------------------------------------
STACKSG   SEGMENT PARA STACK 'STACK'
          DW      32 DUP(?)
STACKSG   ENDS
;-------------------------------------------------
DATASG    SEGMENT PARA 'DATA'
TITLEX    DB      'Change to uppercase letters'
DATASG    ENDS
;-------------------------------------------------
CODESG    SEGMENT PARA 'CODE'
BEGIN     PROC    FAR
          ASSUME  CS:CODESG,DS:DATASG,SS:STACKSG
          PUSH    DS              ;Store DS on stack.
          SUB     AX,AX           ;Set AX to zero.
          PUSH    AX              ;Store zero on stack.
          MOV     AX,DATASG
          MOV     DS,AX
          CALL    B10CASE
          RET
BEGIN     ENDP
;-------------------------------------------------
B10CASE   PROC    NEAR
          MOV     CX,31
          LEA     BX,TITLEX+1
B20:
          MOV     AH,[BX]
          CMP     AH,61H
          JB      B30
          CMP     AH,7AH
          JA      B30
          AND     AH,11011111B
          MOV     [BX],AH
B30:
          INC     BX
          LOOP    B20
          RET
B10CASE   ENDP
;-------------------------------------------------
CODESG    ENDS
          END     BEGIN
```

```
          PAGE    60,132
TITLE     EXADD   Example ADD and SUB operations
;-------------------------------------------------
STACKSG   SEGMENT PARA STACK 'STACK'
          DW      32 DUP(?)
STACKSG   ENDS
;-------------------------------------------------
DATASG    SEGMENT PARA 'DATA'
BYTE1     DB      64H
BYTE2     DB      40H
BYTE3     DB      16H
WORD1     DW      4000H
WORD2     DW      2000H
WORD3     DW      1000H
DATASG    ENDS
;-------------------------------------------------
CODESG    SEGMENT PARA 'CODE'
BEGIN     PROC    FAR
          ASSUME  CS:CODESG,DS:DATASG,SS:STACKSG
          PUSH    DS
          SUB     AX,AX
          PUSH    AX
          MOV     AX,DATASG
          MOV     DS,AX
          CALL    B10ADD
          CALL    C10SUB
          RET
BEGIN     ENDP
;         Examples of ADD bytes:
;         ----------------------
B10ADD    PROC
          MOV     AL,BYTE1
          MOV     BL,BYTE2
          ADD     AL,BL
          ADD     AL,BYTE3
          ADD     BYTE1,BL
          ADD     BL,10H
          ADD     BYTE1,25H
          RET
B10ADD    ENDP
;         Examples of SUB words:
;         ----------------------
C10SUB    PROC
          MOV     AX,WORD1
          MOV     BX,WORD2
          SUB     AX,BX
          SUB     AX,WORD3
          SUB     WORD1,BX
          SUB     BX,1000H
          SUB     WORD1,256H
          RET
```

```
C10SUB   ENDP
;-------------------------------------------------
CODESG   ENDS
         END     BEGIN

         PAGE    60,132
TITLE    EXASM2  Move and Add operations.
;-------------------------------------------------
STACKSG  SEGMENT PARA STACK 'STACK'
         DB      32 DUP(?)
STACKSG  ENDS
;-------------------------------------------------
DATASG   SEGMENT PARA 'DATA'
FLDA     DW      250
FLDB     DW      125
FLDC     DW      ?
DATASG   ENDS
;-------------------------------------------------
CODESG   SEGMENT PARA 'CODE'
BEGIN    PROC    FAR
         ASSUME  CS:CODESG,DS:DATASG,SS:STACKSG,ES:NOTHING
         PUSH    DS              ;Store DS on stack.
         SUB     AX,AX           ;Set AX to zero.
         PUSH    AX              ;Store zero on stack.
         MOV     AX,DATASG       ;Store adress of
         MOV     DS,AX           ;DATASG in DS.

         MOV     AX,FLDA         ;Move 0250 to AX
         ADD     AX,FLDB         ;Add 0125 to AX
         MOV     FLDC,AX         ;Store sum in FLDC
         RET                     ;Return to DOS.
BEGIN    ENDP
CODESG   ENDS
         END     BEGIN

         PAGE    60,132
TITLE    EXCONV  Conversion of ASCII & Binary formats
;-------------------------------------------------
STACKSG  SEGMENT PARA STACK 'STACK'
         DW      32 DUP(?)
STACKSG  ENDS
;-------------------------------------------------
DATASG   SEGMENT PARA 'DATA'
ASCVAL   DB      '1234'
BINVAL   DW      0
ASCLEN   DW      4
MULT10   DW      1
DATASG   ENDS
;-------------------------------------------------
CODESG   SEGMENT PARA 'CODE'
BEGIN    PROC    FAR
```

```
         ASSUME  CS:CODESG,DS:DATASG,SS:STACKSG
         PUSH    DS
         SUB     AX,AX
         PUSH    AX
         MOV     AX,DATASG
         MOV     DS,AX
         CALL    B10ASBI
         CALL    C10BIAS
         RET
BEGIN    ENDP
;        Convert ASCII to Binary:
;        ------------------------
B10ASBI  PROC
         MOV     CX,10
         LEA     SI,ASCVAL-1
         MOV     BX,ASCLEN
B20:
         MOV     AL,[SI+BX]
         AND     AX,000FH
         MUL     MULT10
         ADD     BINVAL,AX
         MOV     AX,MULT10
         MUL     CX
         MOV     MULT10,AX
         DEC     BX
         JNZ     B20
         RET
B10ASBI  ENDP
;        Convert Binary to ASCII:
;        ------------------------
C10BIAS  PROC
         MOV     CX,0010
         LEA     SI,ASCVAL+3
         MOV     AX,BINVAL
C20:
         CMP     AX,0010
         JB      C30
         XOR     DX,DX
         DIV     CX
         OR      DL,30H
         MOV     [SI],DL
         DEC     SI
         JMP     C20
C30:
         OR      AL,30H
         MOV     [SI],AL
         RET
C10BIAS  ENDP
;-------------------------------------------------
CODESG   ENDS
         END     BEGIN
```

```
        PAGE    60,132
TITLE   EXDEF   Assembler Define Pseudo-ops
;-------------------------------------------------
DATASG  SEGMENT PARA 'DATA'
;               Define Byte - DB:
;               ----------------
FLD1DB  DB      ?               ;ilk kosulsuz
FLD2DB  DB      'Personal Computer' ;Karakter dizisi
FLD3DB  DB      25
FLD4DB  DB      19H
FLD5DB  DB      11001B
FLD6DB  DB      01,'JAN',02,'FEB',03,'MAR' ;Tablo
FLD7DB  DB      '12345'
FLD8DB  DB      10 DUP(0)       ;On tane sifir

;               Define Word - DW:
;               -----------------
FLD1DW  DW      0FFF0H
FLD2DW  DW      11001B
FLD3DW  DW      FLD7DB
FLD4DW  DW      1,2,3,4,5
FLD5DW  DW      5 DUP(0)

;               Define Doubleword - DD:

;               ----------------------
FLD1DD  DD      ?
FLD2DD  DD      'PC'
FLD3DD  DD      12345
FLD4DD  DD      FLD3DB - FLD2DB
FLD5DD  DD      12,73

;               Define Quadword - DQ:
;               ---------------------
FLD1DQ  DQ      ?
FLD2DQ  DQ      03C32H
FLD3DQ  DQ      12345

;               Define Tenbytes - DT:
;               ---------------------
FLD1DT  DT      ?
FLD2DT  DT      'PC'

DATASG  ENDS
        END
```

```
         PAGE    60,132
TITLE    EXDIV   Examples of DIV & IDIV operations
;-------------------------------------------------
STACKSG  SEGMENT PARA STACK 'STACK'
         DW      32 DUP(?)
STACKSG  ENDS
;-------------------------------------------------
DATASG   SEGMENT PARA 'DATA'
BYTE1    DB      80H
BYTE3    DB      16H
WORD1    DW      2000H
WORD2    DW      0010H
WORD3    DW      1000H
DATASG   ENDS
;-------------------------------------------------
CODESG   SEGMENT PARA 'CODE'
BEGIN    PROC    FAR
         ASSUME  CS:CODESG,DS:DATASG,SS:STACKSG
         PUSH    DS
         SUB     AX,AX
         PUSH    AX
         MOV     AX,DATASG
         MOV     DS,AX
         CALL    D10DIV
         CALL    E10IDIV
         RET
BEGIN    ENDP
;        Examples of DIV :
;        ----------------
D10DIV   PROC
         MOV     AX,WORD1
         DIV     BYTE1
         MOV     AL,BYTE1
         SUB     AH,AH
         DIV     BYTE3
         MOV     DX,WORD2
         MOV     AX,WORD3
         DIV     WORD1
         MOV     AX,WORD1
         SUB     DX,DX
         DIV     WORD3
         MOV     BYTE1,01
         MOV     AX,1234H
         IDIV    BYTE1
         RET
D10DIV   ENDP
;        Examples of IDIV words:
;        -----------------------
E10IDIV  PROC
         MOV     AX,WORD1
         IDIV    BYTE1
```

```
          MOV     AL,BYTE1
          CBW
          IDIV    BYTE3
          MOV     DX,WORD2
          MOV     AX,WORD3
          IDIV    WORD1
          MOV     AX,WORD1

          CWD
          IDIV    WORD3
          RET
E10IDIV   ENDP
;-------------------------------------------------
CODESG    ENDS
          END     BEGIN
```

```
          PAGE    60,132
TITLE     EXIMM   Example Immediate Operands
;-------------------------------------------------
STACKSG   SEGMENT PARA STACK 'STACK'
          DB      32 DUP(?)
STACKSG   ENDS
;-------------------------------------------------
DATASG    SEGMENT PARA 'DATA'
FLD1      DB      ?
FLD2      DW      ?
DATASG    ENDS
;-------------------------------------------------
CODESG    SEGMENT PARA 'CODE'
BEGIN     PROC    FAR
          ASSUME  CS:CODESG,DS:DATASG,SS:STACKSG

;                 Move & Compare Operations:
;                 -------------------------
          MOV     BX,275
          CMP     AL,19H

;                 Arithmetic Operations:
;                 ---------------------
          ADC     AL,5
          ADD     BH,12

          SBB     AL,5
          SUB     FLD1,5

;                 Rotate & Shift (1 bit only):
;                 ---------------------------
          RCL     BL,1
          RCR     AH,1
```

```
          ROL     FLD2,1
          ROR     AL,1
          SAL     CX,1
          SAR     BX,1
          SHR     FLD1,1

;                 Logical Operations:
;                 -------------------
          AND     AL,00101100B
          OR      BH,2AH
          TEST    BL,7AH
          XOR     FLD1,23H

          RET                       ;Return to DOS.
BEGIN     ENDP
CODESG    ENDS
          END
```

```
          PAGE    60,132
TITLE     EXJUMP  Illustration of JMP for looping
;-------------------------------------------------
STACKSG   SEGMENT PARA STACK 'STACK'
          DB      32 DUP(?)
STACKSG   ENDS
;-------------------------------------------------
CODESG    SEGMENT PARA 'CODE'
BEGIN     PROC    FAR
          ASSUME  SS:STACKSG,CS:CODESG
          PUSH    DS                ;Store DS on stack.
          SUB     AX,AX             ;Set AX to zero.
          PUSH    AX                ;Store zero on stack.
          MOV     AX,01
          MOV     BX,01
          MOV     CX,01
A20:
          ADD     AX,01
          ADD     BX,AX
          SHL     CX,1
          JMP     A20
BEGIN     ENDP
CODESG    ENDS
          END     BEGIN
```

```
          PAGE    60,132
TITLE     EXJUMP  Illustration of LOOP for looping
;-------------------------------------------------
STACKSG   SEGMENT PARA STACK 'STACK'
          DB      32 DUP(?)
STACKSG   ENDS
;-------------------------------------------------
CODESG    SEGMENT PARA 'CODE'
BEGIN     PROC    FAR
          ASSUME  SS:STACKSG,CS:CODESG
          PUSH    DS              ;Store DS on stack.
          SUB     AX,AX           ;Set AX to zero.
          PUSH    AX              ;Store zero on stack.


          MOV     AX,01
          MOV     BX,01
          MOV     DX,01
          MOV     CX,10
A20:
          INC     AX
          ADD     BX,AX
          SHL     DX,1
          LOOP    A20
          RET
BEGIN     ENDP
CODESG    ENDS
          END     BEGIN
```

```
          PAGE    60,132
TITLE     EXMOVE  Extended Move operations
;-------------------------------------------------
STACKSG   SEGMENT PARA STACK 'STACK'
          DW      32 DUP(?)
STACKSG   ENDS
;-------------------------------------------------
DATASG    SEGMENT PARA 'DATA'
NAME1     DB      'ABCDEFGHI'
NAME2     DB      'JKLMNOPQR'
NAME3     DB      'STUVWXYZ*'
DATASG    ENDS
;-------------------------------------------------
CODESG    SEGMENT PARA 'CODE'
BEGIN     PROC    FAR
          ASSUME  CS:CODESG,DS:DATASG,SS:STACKSG,ES:DATASG
          PUSH    DS              ;Store DS on stack.
          SUB     AX,AX           ;Set AX to zero.
          PUSH    AX              ;Store zero on stack.
          MOV     AX,DATASG
```

```
          MOV     DS,AX
          MOV     ES,AX
          CALL    B10MOVE
          CALL    C10MOVE
          RET
BEGIN     ENDP
;         Extended Move using Jump-on-Condition:
;         -------------------------------------
B10MOVE   PROC
          LEA     SI,NAME1
          LEA     DI,NAME2
          MOV     CX,09
B20:
          MOV     AL,[SI]
          MOV     [DI],AL
          INC     SI
          INC     DI
          DEC     CX
          JNZ     B20
          RET
B10MOVE   ENDP
;         Extended Move using LOOP:
;         ------------------------
C10MOVE   PROC
          LEA     SI,NAME2
          LEA     DI,NAME3
          MOV     CX,09
C20:
          MOV     AL,[SI]
          MOV     [DI],AL
          INC     DI
          INC     SI
          LOOP    C20
          RET
C10MOVE   ENDP
;-------------------------------------------------
CODESG    ENDS
          END     BEGIN
```

```
          PAGE    60,132
TITLE     EXADD   Example of MUL & IMUL operations
;-------------------------------------------------
STACKSG   SEGMENT PARA STACK 'STACK'
          DW      32 DUP(?)
STACKSG   ENDS
;-------------------------------------------------
DATASG    SEGMENT PARA 'DATA'
BYTE1     DB      80H
BYTE2     DB      40H
WORD1     DW      8000H
WORD2     DW      2000H
DATASG    ENDS
;-------------------------------------------------
CODESG    SEGMENT PARA 'CODE'
BEGIN     PROC    FAR
          ASSUME  CS:CODESG,DS:DATASG,SS:STACKSG
          PUSH    DS
          SUB     AX,AX
          PUSH    AX
          MOV     AX,DATASG
          MOV     DS,AX
          CALL    C10MUL
          CALL    D10IMUL
          RET
BEGIN     ENDP
;         Examples of MUL:
C10MUL    PROC
          MOV     AL,BYTE1
          MUL     BYTE2
          MOV     AX,WORD1
          MUL     WORD2
          MOV     AL,BYTE1
          SUB     AH,AH
          MUL     WORD1
          RET
C10MUL    ENDP
;         Examples of IMUL:
D10IMUL   PROC
          MOV     AL,BYTE1
          IMUL    BYTE2
          MOV     AX,WORD1
          IMUL    WORD2
          MOV     AL,BYTE1
          CBW
          IMUL    WORD1
          RET
D10IMUL   ENDP
;-------------------------------------------------
CODESG    ENDS
          END     BEGIN
```

```
         PAGE    65,132
TITLE    EXSTRG  Test of string operations
;-------------------------------------------------
STACKSG  SEGMENT PARA STACK 'STACK'
         DW      32 DUP(?)
STACKSG  ENDS
;-------------------------------------------------

DATASG   SEGMENT PARA 'DATA'
NAME1    DB      'Assemblers'
NAME2    DB      10 DUP(' ')
NAME3    DB      10 DUP(' ')
DATASG   ENDS
;-------------------------------------------------
CODESG   SEGMENT PARA 'CODE'
BEGIN    PROC    FAR
         ASSUME  CS:CODESG,DS:DATASG,SS:STACKSG,ES:DATASG
         PUSH    DS
         SUB     AX,AX
         PUSH    AX
         MOV     AX,DATASG
         MOV     DS,AX
         MOV     ES,AX
         CALL    C10MVSB
         CALL    D10MVSW
         CALL    E10LODS
         CALL    F10STOS
         CALL    G10CMPS
         CALL    H10SCAS
         RET
BEGIN    ENDP
;        Use of MVSB:
;        -----------
C10MVSB  PROC
         CLD
         LEA     SI,NAME1
         LEA     DI,NAME2
         MOV     CX,10
         REP     MOVSB
         RET
C10MVSB  ENDP
;        Use of MVSW:
;        -----------
D10MVSW  PROC
         CLD
         LEA     SI,NAME2
         LEA     DI,NAME3
         MOV     CX,05
         REP     MOVSW
         RET
D10MVSW  ENDP
```

```
;         Use of LODSW:
;         ------------
E10LODS   PROC
          CLD
          LEA     SI,NAME1
          LODSW
          RET
E10LODS   ENDP
;         Use of STOSW:
;         ------------
F10STOS   PROC
          CLD
          LEA     DI,NAME3
          MOV     CX,05
          MOV     AX,2020H
          REP     STOSW
          RET
F10STOS   ENDP
;         Use of CMPSB:
;         ------------
G10CMPS   PROC
          CLD
          MOV     CX,10
          LEA     SI,NAME1
          LEA     DI,NAME2
          REPE    CMPSB
          JNE     G20
          MOV     BH,01
G20:
          MOV     CX,10
          LEA     SI,NAME2
          LEA     DI,NAME3
          REPE    CMPSB
          JE      G30
          MOV     BL,02
G30:      RET
G10CMPS   ENDP
;         Use of SCASB:
;         ------------
H10SCAS   PROC
          CLD
          MOV     CX,10
          LEA     DI,NAME1
          MOV     AL,'m'
          REPNE   SCASB
          JNE     H20
          MOV     AH,03
H20:      RET
H10SCAS   ENDP
CODESG    ENDS
          END     BEGIN
```