MICROPROCESSOR PRACTICAL FILE

3rd YEAR

SEMESTER V

SUBMITTED BY
SHAD JAMIL

ROLL NO: CSC/21/45

EXAM ROLL NO: 21059570048

SUBMITTED TO
Dr. SONAL LINDA

INDEX

Q1: Write an assembly language code to display the contents of AX register.

```
;A program that displays the number in AL, loaded
;with the first instruction (48H).
;
.model tiny     ;selects tiny mode
.code           ;start code segment
.startup        ; start program
        MOV AL,48H     ;load test data
        MOV AH,0        ; clear AH
        AAM             ; convert to BCD
        ADD AX,3030H    ; convert to ASCII
        MOV DL,AH       ; display most-significant digit
        MOV AH,2
        PUSH AX
        INT 21H
        POP AX
        MOV DL,AL       ;display least-significant digit
        INT 21H
        MOV AX,4c00h
        INT 21h
```

Output:

```
C:\TASM>tasm ah.asm
Turbo Assembler  Version 2.51  Copyright (c) 1988, 1991 Borland International

Assembling file:   ah.asm
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  491k


C:\TASM>tlink ah.obj
Turbo Link  Version 4.0 Copyright (c) 1991 Borland International
Warning: No stack

C:\TASM>ah
72
```

Q2: Write an assembly language code for bit 16 division.

DIV_16.asm

```
.model tiny
.code
mov ax,02H
mov bx,04H

div bl

DISP PROC
PUSH CX
MOV CL, 4
MOv CH, 4

D1:
  ROL AX, CL
  PUSH AX
  AND AL, 0FH
  ADD AL , 30H
  CMP AL, '9'

  JBE D2
  ADD AL , 7H

D2:
  MOV AH, 02H
  MOV DL, AL
  INT 21H
  POP AX
  DEC CH
  JNZ D1
  POP CX
  RET
  DISP ENDP

mov ah,4ch
int 21h
end

```

Output:

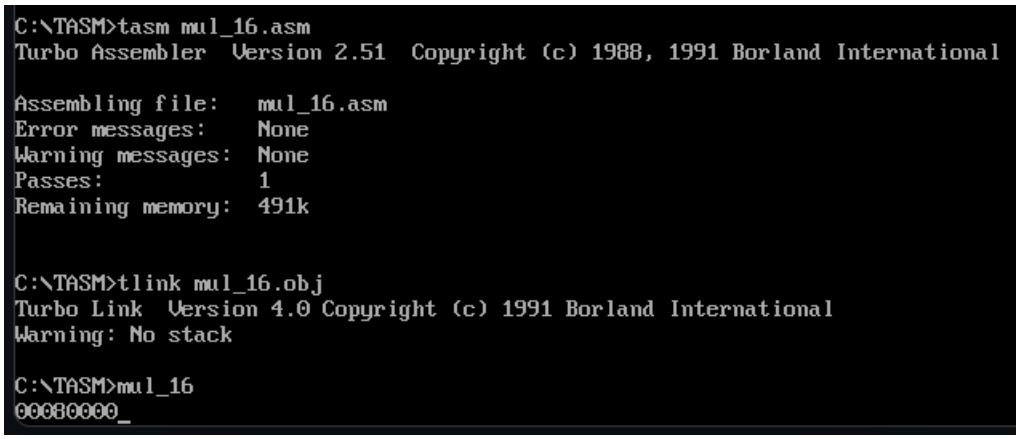

C:\TASM>tasm DIV_16
Turbo Assembler Version 2.51 Copyright (c) 1988, 1991 Borland International
Assembling file: DIV_16.ASM
Error messages: None
Warning messages: None
Passes: 1
Remaining memory: 491k
C:\TASM>tlink div_16.obj
Turbo Link Version 4.0 Copyright (c) 1991 Borland International
Warning: No stack
C:\TASM>div_16
0200

Q3: Write an assembly language code for 16bit multiplication.

mul_16.asm

```
.model tiny
.code

mov ax,04H
mov bx,02H

mul bl

DISP PROC

PUSH CX
MOV CL, 4
MOv CH, 4
D1:
  ROL AX, CL
  PUSH AX
  AND AL, 0FH
  ADD AL , 30H
  CMP AL, '9'

  JBE D2
  ADD AL , 7H
D2:
  MOV AH, 02H
  MOV DL, AL
  INT 21H
  POP AX
  DEC CH
  JNZ D1
  POP CX
  RET
  DISP ENDP
mov ah,4ch
int 21h
end
```

Output:


C:\TASM>tasm mul_16.asm
Turbo Assembler Version 2.51 Copyright (c) 1988, 1991 Borland International
Assembling file: mul_16.asm
Error messages: None
Warning messages: None
Passes: 1
Remaining memory: 491k
C:\TASM>tlink mul_16.obj
Turbo Link Version 4.0 Copyright (c) 1991 Borland International
Warning: No stack
C:\TASM>mul_16
00080000_

Q4: Write an assembly language code for 32bit addition.

```
.model small
.data
op1 dd 1234567Fh
op2 dd 11111111h                        ;Corel - 10
ans dd ?
.code
     mov      ax, @data
     mov      ds, ax
     mov      ax, word ptr op1          ; lsb of number1 in ax
     mov      bx, word ptr op1+2        ; msb of number1 in bx
     mov      cx, word ptr op2          ; lsb of number2 in cx
     mov      dx, word ptr op2+2        ; msb of number1 in dx
     add      ax, cx                    ; add msb + msb + carry
     mov      word ptr ans, ax          ; lsb answer
     mov      word ptr ans+2, bx        ; msb answer
     mov      bx, word ptr ans+2        ; Result in reg bx
     mov      dh, 2
l1:     mov      ch, 04h                ; Count of digits to be displayed
     mov      cl, 04h                   ; Count to roll by 4 bits
l2:     rol      bx, cl                 ; roll bl so that msb comes to lsb
     mov      dl, bl                    ; load dl with data to be displayed
     and      dl, 0fH                   ; get only lsb
     cmp      dl, 09                    ; check if digit is 0-9 or letter A-F
     jbe      l4
     add      dl, 07                    ; if letter add 37H else only add 30H
l4:  add      dl, 30H
     mov      ah, 02                    ; INT 21H (Display character)
     int      21H
     dec      ch                        ; Decrement Count
     dec      ch                        ; Decrement Count
     jnz      l2
     dec      dh
     cmp      dh, 0
     mov     bx, word ptr ans           ; display lsb of answer
     jnz      l1
     mov      ah, 4ch                   ; Terminate Program
     int      21h
     end
```

Output:


C:\TASM>tasm add_32.asm
Turbo Assembler Version 2.51 Copyright (c) 1988, 1991 Borland International
Assembling file: add_32.asm
Error messages: None
Warning messages: None
Passes: 1
Remaining memory: 491k
C:\TASM>tlink add_32.obj
Turbo Link Version 4.0 Copyright (c) 1991 Borland International
Warning: No stack
C:\TASM>add_32
12346790

Q5: Write an assembly language code for 32-bit subtraction.

```
.model small
.data
op1 dd 12345678h
op2 dd 11111111h
ans dd ?
.code
     mov      ax, @data
     mov      ds, ax
     mov      ax, word ptr op1          ; lsb of number1 in ax
     mov      bx, word ptr op1+2        ; msb of number1 in bx
     mov      cx, word ptr op2          ; lsb of number2 in cx
     mov      dx, word ptr op2+2        ; msb of number1 in dx
     sub      ax, cx                    ; subtract lsb + lsb
     mov      word ptr ans, ax          ; lsb answer
     mov      word ptr ans+2, bx        ; msb answer
     mov      bx, word ptr ans+2        ; Result in reg bx
     mov      dh, 2
l1:     mov      ch, 04h                ; Count of digits to be displayed
     mov      cl, 04h                   ; Count to roll by 4 bits
l2:     rol      bx, cl                 ; roll bl so that msb comes to lsb
     mov      dl, bl                    ; load dl with data to be displayed
     and      dl, 0fH                   ; get only lsb
     cmp      dl, 09                    ; check if digit is 0-9 or letter A-F
     jbe      l4
     add      dl, 07                    ; if letter add 37H else only add 30H
l4:     add      dl, 30H
     mov      ah, 02                    ; INT 21H (Display character)
     int      21H
     dec      ch                        ; Decrement Count
```

Output:

```
C:\TASM>tasm sub_32.asm
Turbo Assembler  Version 2.51  Copyright (c) 1988, 1991 Borland International

Assembling file:   sub_32.asm
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  491k


C:\TASM>tlink sub_32.obj
Turbo Link  Version 4.0 Copyright (c) 1991 Borland International
Warning: No stack

C:\TASM>sub_32
12344567
C:\TASM>S
```

Q6: Write an assembly language code for 32bit division and multiplication.

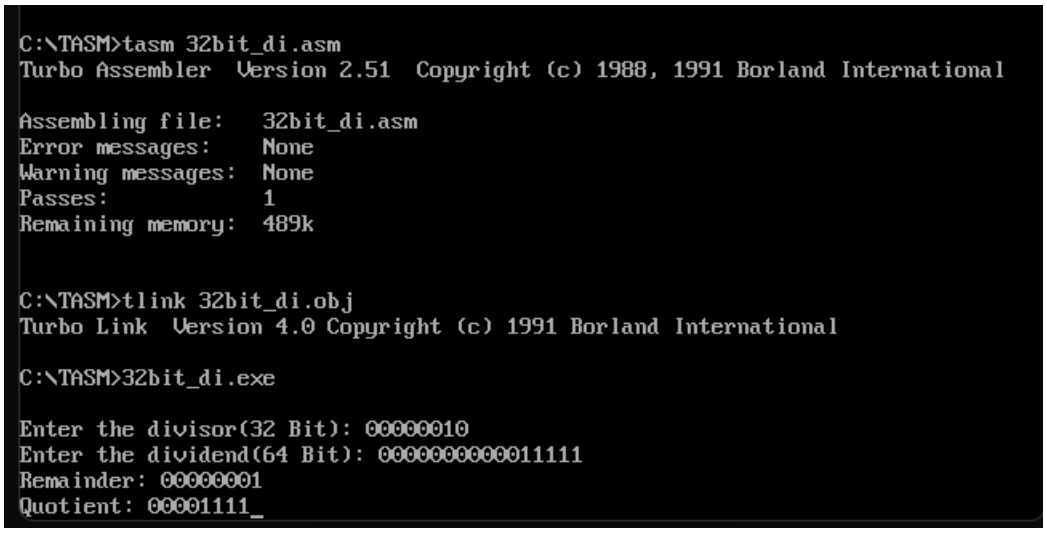
Division:

```
.MODEL SMALL
.STACK
.486
.DATA
    ad  dd ?
    ad1 dd ?
    msg db 13, 10, "Enter the divisor(32 Bit): $"
    msg1 db 13, 10, "Enter the dividend(64 Bit): $"
    msg2 db 13, 10, "Remainder: $"
    msg3 db 13, 10, "Quotient: $"
.CODE
.STARTUP
    ; Input of divisor(32 bit)
    MOV DX, offset msg
    MOV AH, 09
    INT 21h
    MOV EBX, 0
    MOV CX, 4
    abc: SHL EBX, 8
    ; 1st PART OF DIVISOR
    MOV AH, 01
    INT 21h
    CMP AL, 39h
    JBE ab1
    SUB AL, 37h
    ab1:
    AND AL, 00fh
    SHL AL, 4
    MOV BL, AL
    ; 2nd PART OF DIVISOR
    MOV AH, 01
    INT 21h
    CMP AL, 39h
    JBE ab2
    SUB AL, 37h
    ab2:
    AND AL, 00fh
    ADD BL, AL
    LOOP abc
    MOV ad, EBX
    ; Input of  dividend (64 bit)
    MOV DX, offset msg1
    MOV AH, 09
    INT 21h
    MOV EBX, 0
    MOV CX, 4
    abc1: SHL EBX, 8
    ; 1st PART OF DIVIDEND
    MOV AH, 01
    INT 21h
    CMP AL, 39h
    JBE ab3
    SUB AL, 37h
    ab3:
    AND AL, 00fh
```

```
    AND AL, 00fh
    SHL AL, 4
    MOV BL, AL
    ; 2nd PART OF DIVIDEND
    MOV AH, 01
    INT 21h
    CMP AL, 39h
    JBE ab4
    SUB AL, 37h
    ab4:
    AND AL, 00fh
    ADD BL, AL
    LOOP abc1
    MOV EDX, EBX
    MOV EBX, 0
    MOV CX, 4
    abc11: SHL EBX, 8
    ;1st DIGIT OF SECOND NO.
    MOV AH, 01
    INT 21h
    CMP AL, 39h
    JBE ab31
    SUB AL, 37h
    ab31:
    AND AL, 00fh
    SHL AL, 4
    MOV BL, AL
    ;2nd  DIGIT OF SECOND NO.
    MOV AH, 01
    INT 21h
    CMP AL, 39h
    JBE ab41
    SUB AL, 37h
    ab41:
    AND AL, 00fh
    ADD BL, AL
    LOOP abc11
    MOV EAX, EBX
    MOV EBX, ad
    div EBX
    MOV ad,EAX
    MOV ad1,EDX
    ; Printing
    MOV DX, offset msg2
    MOV AH, 09
    INT 21h
    MOV EBX, ad1
    MOV CX, 4
    abc3 :rol EBX, 8
    MOV AL, BL
    AND AL, 0f0h
    SHR AL, 4
    ADD AL, 30h
    CMP AL, 39h
    JBE ab5
```

```
    CMP AL, 39h
    JBE ab5
    ADD AL, 07h
    ab5:
    MOV DL,  AL
    MOV AH, 02
    INT 21h
    MOV AL, BL
    AND AL, 00fh
    ADD AL, 30h
    CMP AL, 39h
    JBE ab6
    ADD AL, 07h
    ab6:
    MOV DL, AL
    MOV AH, 02
    INT 21h
    LOOP abc3
    ; Printing
    MOV DX, offset msg3
    MOV AH, 09
    INT 21h
    MOV EBX, ad
    MOV CX, 4
    abc4 :rol EBX, 8
    MOV AL, BL
    AND AL, 0f0h
    SHR AL, 4
    ADD AL, 30h
    CMP AL, 39h
    JBE ab7
    ADD AL, 07h
    ab7:
    MOV DL, AL
    MOV AH, 02
    INT 21h
    MOV AL, BL
    AND AL, 00fh
    ADD AL, 30h
    CMP AL, 39h
    JBE ab8
    ADD AL, 07h
    ab8:
    MOV DL, AL
    MOV AH, 02
    INT 21h
    LOOP abc4

END
```

Output:


C:\TASM>tasm 32bit_di.asm
Turbo Assembler Version 2.51 Copyright (c) 1988, 1991 Borland International
Assembling file: 32bit_di.asm
Error messages: None
Warning messages: None
Passes: 1
Remaining memory: 489k
C:\TASM>tlink 32bit_di.obj
Turbo Link Version 4.0 Copyright (c) 1991 Borland International
C:\TASM>32bit_di.exe
Enter the divisor(32 Bit): 00000010
Enter the dividend(64 Bit): 0000000000011111
Remainder: 00000001
Quotient: 00001111_

Multiplication:

```
.MODEL SMALL
.STACK
.486
.DATA
first dd ?
second dd ?
INMSG1 db 13, 10, "Enter the first number(32 Bit): $"
INMSG2 db 13, 10, "Enter the second number(32 Bit): $"
OUTMSG db 13, 10, "Product: $"
.CODE
.STARTUP
; INPUT - FIRST 32-bit NUMBER
    ; Print 'INMSG1'
    MOV DX, OFFSET INMSG1 ; DX <- Address of msg
    MOV AH, 09 ; 09 - Print string to stdout w/ echo
    INT 21H ; Interrupt - Application specific
    ; Initialisations
    MOV EBX, 0 ; Clear EBX
    MOV CX, 4 ; Counter for loop
    L1: SHL EBX, 8 ; Shift 8 bits to left
    ; 1st DIGIT of FIRST Part
    MOV AH, 01 ; 01 - single byte input
    INT 21H ; Interrupt - Application specific
    ; Compare input byte with 39H(ASCII for decimal 9)
    CMP AL, 39H
    JBE isDigit1 ; if input is below(less) or equal (0-9)
    SUB AL, 37H ; if input is a letter (out of A,B,C,D,E,F)
    isDigit1: AND AL, 0FH ; Masking
    ; (Now, AL will contain Hex value for ASCII input.)
    SHL AL, 4 ; Shift 4 bits to left i.e. append 0H
    MOV BL, AL ; Store it in BL
    ; 2nd  DIGIT of FIRST Part (Same as FIRST Part)
    MOV AH, 01
    INT 21H
    CMP AL, 39H
    JBE isDigit2
    SUB AL, 37H
    isDigit2: AND AL, 0FH
    ADD BL, AL ; Add new input(in AL) to BL
    LOOP L1
    MOV first, EBX
    ; INPUT - SECOND 32-bit NUMBER [Same as FIRST NUMBER]
    MOV DX, OFFSET INMSG2
    MOV AH, 09
    INT 21H
    MOV EBX, 0
    MOV CX, 4
    L2: SHL EBX, 8
    ; 1st DIGIT of FIRST Part
    MOV AH, 01
    INT 21H
    CMP AL, 39H
    JBE isDigit3
    SUB AL, 37H
    isDigit3: AND AL, 0FH
```

```
    isDigit3: AND AL, 0FH
    SHL AL, 4
    MOV BL, AL
    ; 2nd DIGIT of SECOND Part
    MOV AH, 01
    INT 21H
    CMP AL, 39H
    JBE isDigit4
    SUB AL, 37h
    isDigit4: AND AL, 00FH
    ADD BL, AL
    LOOP  L2
    ; MULTIPLICATION
    MOV EAX, first ; Copy first num to EAX for MUL
    MUL EBX ; Multiplication with contents of EBX is second num
    ; Now, the product (64-bit) is stored as,
    MOV first, EAX ; least significant 32 bits
    MOV second, EDX ; most significant 32 bits
    ; OUTPUT - PRINTING RESULTS
    ; Print 'OUTMSG'
    MOV DX, OFFSET OUTMSG
    MOV AH, 09
    INT 21H
    ; Print most significant 32 bits in ASCII
    MOV EBX, second
    MOV CX, 4
    L3: ROL EBX, 8
    MOV AL, BL
    AND AL, 0F0H
    SHR AL, 4
    ADD AL, 30H
    CMP AL, 39H
    JBE ab5
    ADD AL, 07H
    ab5:
    MOV DL, AL
    MOV AH, 02
    INT 21H
    MOV AL, BL
    AND AL, 00FH
    ADD AL, 30H
    CMP AL, 39H
    JBE ab6
    ADD AL, 07H
    ab6:
    MOV DL, AL
    MOV AH, 02
    INT 21H
    LOOP L3
    ; Print least significant 32 bits in ASCII
    MOV EBX, first
    MOV CX, 4
    L4: ROL EBX,8
```

```
106     MOV CX, 4
107     L4: ROL EBX,8
108     MOV AL, BL
109     AND AL, 0F0H
110     SHR AL, 4
111     ADD AL, 30H
112     CMP AL, 39H
113     JBE ab7
114     ADD AL, 07H
115     ab7:
116     MOV DL, AL
117     MOV AH, 02
118     INT 21H
119     MOV AL, BL
120     AND AL, 00FH
121     ADD AL, 30H
122     CMP AL, 39H
123     JBE ab8
124     ADD AL, 07H
125     ab8:
126     MOV DL, AL
127     MOV AH, 02
128     INT 21H
129     LOOP L4
130 mov ah,4cH
131 int 21H
132 End
```

Output:

```
C:\TASM>tasm 32bit_m.asm
Turbo Assembler  Version 2.51  Copyright (c) 1988, 1991 Borland International

Assembling file:   32bit_m.asm
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  489k


C:\TASM>tlink 32bit_m.obj
Turbo Link  Version 4.0 Copyright (c) 1991 Borland International

C:\TASM>32bit_m.exe

Enter the first number(32 Bit): 00000010
Enter the second number(32 Bit): 00000100
Product: 0000000000001000
```

Q7: Write an assembly language code for BCD addition and BCD subtraction.

Addition:

bcda.asm

```
.MODEL SMALL ;assembler memory model
.STACK 100H
.DATA
MSG1 DB "4 DIGIT BCD SUM IS = $"
BCDSUML DB ?
BCDSUMH DB ?
.CODE
MOV AX,@DATA
MOV DS,AX
XOR AX, AX ; clear register AX
MOV AL,34H
MOV BL, 98H
ADD AL,BL
DAA; DECIMAL ADJUST AFTER ADDITION
MOV BCDSUML, AL
MOV AL,12H
MOV BL, 23H
ADC AL,BL
DAA; OPERATES ONLY ON AL
MOV BCDSUMH,AL
MOV AH,BCDSUMH
MOV AL,BCDSUML
PUSH AX
MOV DX, OFFSET MSG1
MOV AH, 09H
INT 21H ; INT STANDS FOR INTERRUPT INSTRUCTION 21H IS INTERRUPT NO FOR DOS SERVICES
POP AX
CALL DISPLAY
MOV AH, 4CH
INT 21H
DISPLAY PROC NEAR ; PROC IS KEYWORD FOR PROCEDURE
MOV CH, 04H
MOV CL, 04H
DISP1:
ROL AX, CL ; ROTATE LEFT 4 TIMES
PUSH AX ; SAVING ON STACK
AND AL, 0FH
ADD AL, 30H ; 48 IN DECIMAL
CMP AL, '9' ; COMPARE WITH ASCII VALUE OF 9
JBE DISP2
ADD AL, 7
DISP2: MOV DL, AL
MOV AH, 02H
INT 21H
POP AX
DEC CH
JNZ DISP1
RET
DISPLAY ENDP
END
```

Output:


C:\TASM>tasm bcda.asm
Turbo Assembler Version 2.51 Copyright (c) 1988, 1991 Borland International
Assembling file: bcda.asm
Warning bcda.asm(31) Reserved word used as symbol: DISPLAY
Error messages: None
Warning messages: 1
Passes: 1
Remaining memory: 490k
C:\TASM>tlink bcda.obj
Turbo Link Version 4.0 Copyright (c) 1991 Borland International
C:\TASM>bcda
4 DIGIT BCD SUM IS = 3632

Subtraction:

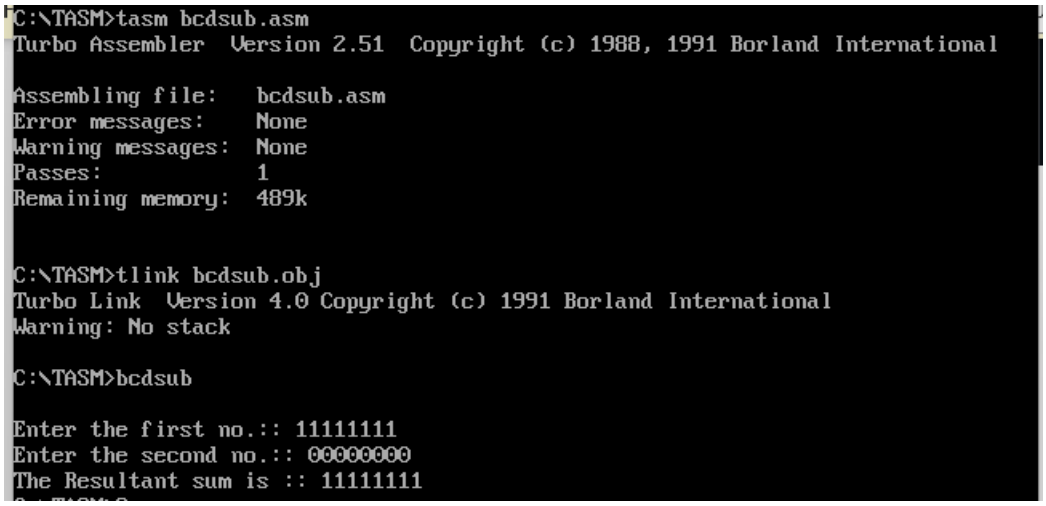
bcdsub.asm

```
.model small
.386
.data
num1 DD 00000000H
num2 DD 00000000H
num3 DD 00000000H
msg db 10,13,"Enter the first no.:: $"
msg1 db 10,13,"Enter the second no.:: $"
msg2 db 10,13,"The Resultant sum is :: $"

.code
.startup

MOV AH,09
MOV DX,OFFSET msg
INT 21H

MOV EBX,0
MOV CX,8

AGAIN: MOV AH,01 ;1ST NO. ENTERED
INT 21H
CMP AL,'A'
JGE L2
SUB AL,30H
SHL EBX,4
ADD BL,AL
LOOP AGAIN

MOV num1,EBX

MOV AH,09
MOV DX,OFFSET msg1
INT 21H

MOV EBX,0
MOV CX,8
AGAIN1:MOV AH,01 ;2nd NO. ENTERED
INT 21H
CMP AL,'A'
JGE L2
SUB AL,30H
SHL EBX,4
ADD BL,AL
LOOP AGAIN1
```

bcdsub.asm

```
43    SHL EBX,4
44    ADD BL,AL
45    LOOP AGAIN1
46
47    MOV num2, EBX
48
49    mov ax, word ptr num1
50    mov dx, word ptr num2
51    add al,dl
52    daa
53    mov bl,al
54
55    mov al,ah
56    adc al,dh
57    daa
58    mov bh,al
59
60    mov word ptr num3, bx
61
62    mov ax, word ptr num1+2
63    mov dx, word ptr num2+2
64    adc al,dl
65    daa
66    mov bl,al
67
68    mov al,ah
69    adc al,dh
70    daa
71    mov bh,al
72
73    mov word ptr num3+2,bx
74    mov ebx,num3
75
76
77
78    mov ah, 09h
79    mov dx, offset msg2
80    int 21h
81    jnc 16
82    mov ah, 02h
83    mov dl, "1"
84    int 21h
85    16: MOV CX,8
86    AGAIN2: ROL EBX,4
87    MOV DL,BL
88    AND DL,0FH
89    ADD DL,30H
90    MOV AH,02
91    INT 21H
92    LOOP AGAIN2
93    L2:
94    mov ah,4CH
95    int 21h
96    END
```

Output:


C:\TASM>tasm bcdsub.asm
Turbo Assembler Version 2.51 Copyright (c) 1988, 1991 Borland International
Assembling file: bcdsub.asm
Error messages: None
Warning messages: None
Passes: 1
Remaining memory: 489k
C:\TASM>tlink bcdsub.obj
Turbo Link Version 4.0 Copyright (c) 1991 Borland International
Warning: No stack
C:\TASM>bcdsub
Enter the first no.:: 11111111
Enter the second no.:: 00000000
The Resultant sum is :: 11111111

Q8: Write an assembly language code for Linear Search and Binary Search.

Linear Search:

lins.asm

```
;Linear Search
.model small
.386
.data
ARRAY DW 20 DUP (?)
DATA1 dw 0000H
success db 10,13,"Element is present in the array $"
fail db 10,13,"Element is not present in the arary $"
msg db 10,13,"Enter the size of the array :: $"
msg2 db 10,13,"Enter the array :: $"
msg3 db 10,13,"Enter the element to be searched :: $"
.code
.startup
MOV AH,09
MOV DX,OFFSET msg
INT 21H

MOV AH,01
INT 21H
SUB AL,30H
MOV AH,0
MOV CX,AX

MOV DATA1,AX

MOV AH,09
MOV DX,OFFSET msg2
INT 21H
MOV AH,0
MOV SI, 0
```

```
31   MOV BX, OFFSET ARRAY
32   L1: MOV DL, 0AH ; jump onto next line
33   MOV AH, 02H
34   INT 21H
35   MOV DX, SI ; input element of the array
36   MOV AH, 01H
37   INT 21H
38
39   SUB AL,30H
40   ;MOV SI, DX
41   MOV [BX + SI], AX
42   INC SI
43   LOOP L1
44
45   MOV CX,DATA1
46
47   MOV AH,09
48   MOV DX,OFFSET msg3
49   INT 21H
50
51   MOV AH,01 ; Enter element to be searched
52   INT 21H
53   SUB AL,30H
54   MOV SI, 0
55   MOV BX, OFFSET ARRAY
56   L2: CMP [BX + SI], AL ; linear search loop
57   JZ L3 ; jump if element is found
58   INC SI
59   LOOP L2
60
61   MOV AH, 09H
62   MOV DX,OFFSET fail ; if the element is not found
63   INT 21H
64   MOV AH, 4CH ; to forcefully terminate the program
65   INT 21H
66   L3: MOV AH, 09H
67   MOV DX,OFFSET success ; if the element is found
68   INT 21H
69   MOV AH, 4CH
70   INT 21H
71   END
```

Output:

```
C:\TASM>lins

Enter the size of the array :: 4
Enter the array ::
5
7
6
1
Enter the element to be searched :: 7
Element is present in the array
```

Binary Search:

```
.model small
.386
.data
ARRAY DW 20 DUP (?)
DATA1 dw 0000H
DATA2 dw 0000H
success db 10,13,"Element is present in the array $"
fail db 10,13,"Element is not present in the arary $"
msg db 10,13,"Enter the size of the array :: $"
msg2 db 10,13,"Enter the array :: $"
msg3 db 10,13,"Enter the element to be searched :: $"
.code
.startup
MOV AH,09
MOV DX,OFFSET msg
INT 21H
MOV AH,01
INT 21H
SUB AL,30H
MOV AH,0
MOV CX,AX
MOV DATA1,AX

MOV AH,09
MOV DX,OFFSET msg2
INT 21H
MOV AH,0
MOV SI, 0
MOV BX, OFFSET ARRAY
L1: MOV DL, 0AH ; jump onto next line
MOV AH, 02H
INT 21H
MOV DX, SI ; input element of the array
MOV AH, 01H
INT 21H
SUB AL,30H
MOV SI, DX
MOV [BX + SI], AX
INC SI
LOOP L1

MOV AH,09
MOV DX,OFFSET msg3
INT 21H
MOV AH,01 ; Enter element to be searched
```

```
44    INT 21H
45    MOV AH,01 ; Enter element to be searched
46    INT 21H
47    SUB AL,30H
48    MOV DATA2,AX
49    MOV CX,DATA1
50    MOV SI,0
51    MOV DI, DATA1
52    MOV BP, 0
53    MOV BX, OFFSET ARRAY
54    MOV AX, DATA1
55    L2: MOV SI, DI
56    ADD SI, BP
57    MOV AX, SI
58    MOV DL, 2
59    DIV DL
60    MOV AH,0
61    MOV DX,0
62    MOV SI,AX
63    MOV DX,DATA2
64    CMP [BX + SI],DL
65    JZ L3
66    CALL L4
67    LOOP L2
68    MOV AH, 09H
69    MOV DX,OFFSET fail ; if the element is not found
70    INT 21H
71    MOV AH, 4CH ; to forcefully terminate the program
72    INT 21H
73    L3: MOV AH, 09H
74    MOV DX,OFFSET success ; if the element is found
75    INT 21H
76    MOV AH, 4CH
77    INT 21H
78    L4 PROC NEAR
79    CMP [BX+SI], DL
80    JL L6
81    MOV DI, SI
82    RET
83    L6: MOV BP,SI
84    RET
85    L4 ENDP
86    mov ah,4ch
87    int 21h
88    END
```

Output:

```
C:\TASM>bins

Enter the size of the array :: 4
Enter the array ::
2
4
6
7
Enter the element to be searched :: 4
Element is present in the array
```

Q9: Write a Program for binary to ascii conversion.

btoa.asm

```
.model small
.data
      array db 8 dup(?)
      msg db 0dh,0ah,'Program for conversion of binary to ascii:$'
      msg1 db 0dh,0ah,'Enter the element to array:$'

.code
.startup

mov dx,offset msg
mov ah,09h
int 21h
mov dx,offset msg1
mov si,0
mov cx,8
again:
    mov ah,01h
    int 21h
    sub al,30h
    mov array[si],al
    inc si
loop again

mov cx,8
mov al,01h
mov sp,0h
mov si,07h
mov bl,02h
again1:
```

```
29    again1:
30
31        mov dl,array[si]
32        cmp dl,01h
33        jz  l2
34    here:
35    dec si
36    mul bl
37    loop again1
38
39    jmp ext
40
41    l2:
42       add sp, ax
43       jmp here
44
45    ext:
46    mov dx,sp
47    mov ah,02h
48    int 21h
49
50    mov ax,4ch;
51    int 21h
52    end
```

Output:

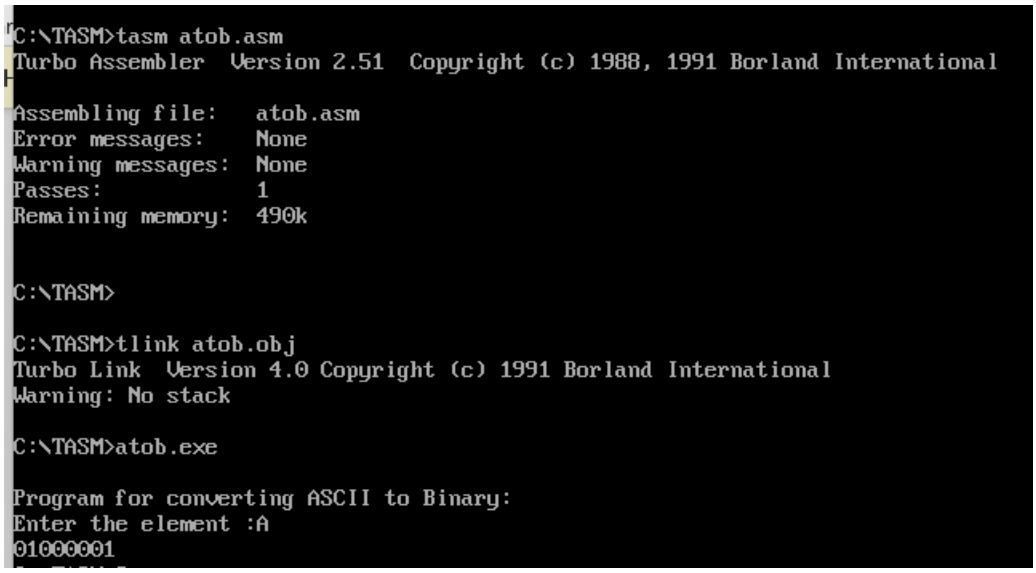
```
C:\TASM>btoa

Program for conversion of binary to ascii:01000000@
```

Q10: Write a program for ascii to binary conversion.

atob.asm

```
;6. Write a program for ascii to binary conversion.
.model small
.data
    msg db 0dh,0ah,'Program for converting ASCII to Binary:$'
    msg1 db 0dh,0ah,'Enter the element :$'

.code
 .startup

 mov dx,offset msg
 mov ah,09h
 int 21h
 mov dx,offset msg1
 mov ah,09h
 int 21h
 mov ah,01h
 int 21h
 mov bl,al
 mov dl,0Ah
 mov ah,02h
 int 21h
 mov cx,8
 again:
   shl bl,1
   jc l2
   jnc l3
   loop again
 l2:
   mov dl,31h
   mov ah,02h
   int 21h
   jmp l4
 l3:
   mov dl,30h
   mov ah,02h
   int 21h
   jmp l4
 l4:
   loop again

mov ah,4ch
int 21h
end
```

Output:


C:\TASM>tasm atob.asm
Turbo Assembler Version 2.51 Copyright (c) 1988, 1991 Borland International
Assembling file: atob.asm
Error messages: None
Warning messages: None
Passes: 1
Remaining memory: 490k
C:\TASM>
C:\TASM>tlink atob.obj
Turbo Link Version 4.0 Copyright (c) 1991 Borland International
Warning: No stack
C:\TASM>atob.exe
Program for converting ASCII to Binary:
Enter the element :A
01000001