



MICROPROCESSOR PRACTICAL FILE

3rd YEAR

SEMESTER V

SUBMITTED BY
SHAD JAMIL

SUBMITTED TO
Dr. SONAL LINDA

ROLL NO: CSC/21/45

EXAM ROLL NO: 21059570048

INDEX

Q1	Write an assembly language code to display the contents of AX register.	3
Q2	Write an assembly language code for bit 16 division.	3-5
Q3	Write an assembly language code for 16bit multiplication.	6-7
Q4	Write an assembly language code for 32bit addition	8-9
Q5	Write an assembly language code for 32-bit subtraction.	10-11
Q6	Write an assembly language code for 32bit division and multiplication.	11-18
Q7	Write an assembly language code for BCD addition and BCD subtraction.	19-24
Q8	Write an assembly language code for Linear Search and Binary Search.	25-29
Q9	Write a Program for binary to ascii conversion.	29-30
Q10	Write a program for ascii to binary conversion.	31-32

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

```
1  ;A program that displays the number in AL, loaded
2  ;with the first instruction (48H).
3  ;
4  .model tiny      ;selects tiny mode
5  .code            ;start code segment
6  .startup         ; start program
7      MOV AL,48H    ;load test data
8      MOV AH,0      ; clear AH
9      AAM           ; convert to BCD
10     ADD AX,3030H   ; convert to ASCII
11     MOV DL,AH      ; display most-significant digit
12     MOV AH,2
13     PUSH AX
14     INT 21H
15     POP AX
16     MOV DL,AL      ;display least-significant digit
17     INT 21H
18     MOV AX,4c00h
19     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
1  .model tiny
2  .code
3  mov ax,02H
4  mov bx,04H
5
6  div bl
7
8  DISP PROC
9  PUSH CX
10 MOV CL, 4
11 MOV CH, 4
12
13 D1:
14     ROL AX, CL
15     PUSH AX
16     AND AL, 0FH
17     ADD AL, 30H
18     CMP AL, '9'
19
20     JBE D2
21     ADD AL, 7H
22
23 D2:
24     MOV AH, 02H
25     MOV DL, AL
26     INT 21H
27     POP AX
28     DEC CH
29     JNZ D1
30     POP CX
31     RET
32     DISP ENDP
33
34 mov ah,4ch
35 int 21h
36 end
37

```

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
1  .model tiny
2  .code
3
4  mov ax,04H
5  mov bx,02H
6
7  mul bl
8
9  DISP PROC
10
11  PUSH CX
12  MOV CL, 4
13  MOV CH, 4
14  D1:
15      ROL AX, CL
16      PUSH AX
17      AND AL, 0FH
18      ADD AL, 30H
19      CMP AL, '9'
20
21      JBE D2
22      ADD AL, 7H
23  D2:
24      MOV AH, 02H
25      MOV DL, AL
26      INT 21H
27      POP AX
28      DEC CH
29      JNZ D1
30      POP CX
31      RET
32  DISP ENDP
33  mov ah,4ch
34  int 21h
35  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.

```

1  .model small
2  .data
3  op1 dd 1234567Fh
4  op2 dd 11111111h           ;Corel - 10
5  ans dd ?
6  .code
7      mov     ax, @data
8      mov     ds, ax
9      mov     ax, word ptr op1       ; lsb of number1 in ax
10     mov     bx, word ptr op1+2     ; msb of number1 in bx
11     mov     cx, word ptr op2       ; lsb of number2 in cx
12     mov     dx, word ptr op2+2     ; msb of number1 in dx
13     add     ax, cx                 ; add msb + msb + carry
14     mov     word ptr ans, ax        ; lsb answer
15     mov     word ptr ans+2, bx      ; msb answer
16     mov     bx, word ptr ans+2     ; Result in reg bx
17     mov     dh, 2
18 11:     mov     ch, 04h             ; Count of digits to be displayed
19         mov     cl, 04h             ; Count to roll by 4 bits
20 12:     rol     bx, cl               ; roll bx so that msb comes to lsb
21         mov     dl, bl               ; load dl with data to be displayed
22         and     dl, 0fh              ; get only lsb
23         cmp     dl, 09               ; check if digit is 0-9 or letter A-F
24         jbe     14
25         add     dl, 07                ; if letter add 37H else only add 30H
26 14:     add     dl, 30H
27         mov     ah, 02                ; INT 21H (Display character)
28         int     21h
29         dec     ch                    ; Decrement Count
29     dec     ch                    ; Decrement Count
30     jnz     12
31     dec     dh
32     cmp     dh, 0
33     mov     bx, word ptr ans        ; display lsb of answer
34     jnz     11
35     mov     ah, 4ch                 ; Terminate Program
36     int     21h
37     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.

```
1  .model small
2  .data
3  op1 dd 12345678h
4  op2 dd 11111111h
5  ans dd ?
6  .code
7      mov     ax, @data
8      mov     ds, ax
9      mov     ax, word ptr op1      ; lsb of number1 in ax
10     mov     bx, word ptr op1+2    ; msb of number1 in bx
11     mov     cx, word ptr op2      ; lsb of number2 in cx
12     mov     dx, word ptr op2+2    ; msb of number2 in dx
13     sub     ax, cx                 ; subtract lsb + lsb
14     mov     word ptr ans, ax       ; lsb answer
15     mov     word ptr ans+2, bx     ; msb answer
16     mov     bx, word ptr ans+2     ; Result in reg bx
17     mov     dh, 2
18 11:     mov     ch, 04h            ; Count of digits to be displayed
19     mov     cl, 04h              ; Count to roll by 4 bits
20 12:     rol     bx, cl             ; roll bx so that msb comes to lsb
21     mov     dl, bl               ; load dl with data to be displayed
22     and     dl, 0fh              ; get only lsb
23     cmp     dl, 09               ; check if digit is 0-9 or letter A-F
24     jbe     14                   ; if letter add 37H else only add 30H
25     add     dl, 07
26 14:     add     dl, 30H
27     mov     ah, 02               ; INT 21H (Display character)
28     int     21h
29     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>
```

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

Division:

```
1  .MODEL SMALL
2  .STACK
3  .486
4  .DATA
5      ad dd ?
6      ad1 dd ?
7      msg db 13, 10, "Enter the divisor(32 Bit): $"
8      msg1 db 13, 10, "Enter the dividend(64 Bit): $"
9      msg2 db 13, 10, "Remainder: $"
10     msg3 db 13, 10, "Quotient: $"
11 .CODE
12 .STARTUP
13     ; Input of divisor(32 bit)
14     MOV DX, offset msg
15     MOV AH, 09
16     INT 21h
17     MOV EBX, 0
18     MOV CX, 4
19     abc: SHL EBX, 8
20     ; 1st PART OF DIVISOR
21     MOV AH, 01
22     INT 21h
23     CMP AL, 39h
24     JBE ab1
25     SUB AL, 37h
26     ab1:
27     AND AL, 00fh
28     SHL AL, 4
29     MOV BL, AL
30     ; 2nd PART OF DIVISOR
31     MOV AH, 01
32     INT 21h
33     CMP AL, 39h
34     JBE ab2
35     SUB AL, 37h
36     ab2:
37     AND AL, 00fh
38     ADD BL, AL
39     LOOP abc
40     MOV ad, EBX
41     ; Input of dividend (64 bit)
42     MOV DX, offset msg1
43     MOV AH, 09
44     INT 21h
45     MOV EBX, 0
46     MOV CX, 4
47     abc1: SHL EBX, 8
48     ; 1st PART OF DIVIDEND
49     MOV AH, 01
50     INT 21h
51     CMP AL, 39h
52     JBE ab3
53     SUB AL, 37h
54     ab3:
55     AND AL, 00fh
```

```

55     AND AL, 00fh
56     SHL AL, 4
57     MOV BL, AL
58     ; 2nd PART OF DIVIDEND
59     MOV AH, 01
60     INT 21h
61     CMP AL, 39h
62     JBE ab4
63     SUB AL, 37h
64     ab4:
65     AND AL, 00fh
66     ADD BL, AL
67     LOOP abc1
68     MOV EDX, EBX
69     MOV EBX, 0
70     MOV CX, 4
71     abc11: SHL EBX, 8
72     ;1st DIGIT OF SECOND NO.
73     MOV AH, 01
74     INT 21h
75     CMP AL, 39h
76     JBE ab31
77     SUB AL, 37h
78     ab31:
79     AND AL, 00fh
80     SHL AL, 4
81     MOV BL, AL
82     ;2nd DIGIT OF SECOND NO.
83     MOV AH, 01
84     INT 21h
85     CMP AL, 39h
86     JBE ab41
87     SUB AL, 37h
88     ab41:
89     AND AL, 00fh
90     ADD BL, AL
91     LOOP abc11
92     MOV EAX, EBX
93     MOV EBX, ad
94     div EBX
95     MOV ad,EAX
96     MOV ad1,EDX
97     ; Printing
98     MOV DX, offset msg2
99     MOV AH, 09
100    INT 21h
101    MOV EBX, ad1
102    MOV CX, 4
103    abc3 :rol EBX, 8
104    MOV AL, BL
105    AND AL, 0f0h
106    SHR AL, 4
107    ADD AL, 30h
108    CMP AL, 39h
109    JBE ab5

```

```
108     CMP AL, 39h
109     JBE ab5
110     ADD AL, 07h
111     ab5:
112     MOV DL, AL
113     MOV AH, 02
114     INT 21h
115     MOV AL, BL
116     AND AL, 00fh
117     ADD AL, 30h
118     CMP AL, 39h
119     JBE ab6
120     ADD AL, 07h
121     ab6:
122     MOV DL, AL
123     MOV AH, 02
124     INT 21h
125     LOOP abc3
126     ; Printing
127     MOV DX, offset msg3
128     MOV AH, 09
129     INT 21h
130     MOV EBX, ad
131     MOV CX, 4
132     abc4 :rol EBX, 8
133     MOV AL, BL
134     AND AL, 0f0h
135     SHR AL, 4
136     ADD AL, 30h
137     CMP AL, 39h
138     JBE ab7
139     ADD AL, 07h
140     ab7:
141     MOV DL, AL
142     MOV AH, 02
143     INT 21h
144     MOV AL, BL
145     AND AL, 00fh
146     ADD AL, 30h
147     CMP AL, 39h
148     JBE ab8
149     ADD AL, 07h
150     ab8:
151     MOV DL, AL
152     MOV AH, 02
153     INT 21h
154     LOOP abc4
155
156     END
157
```

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): 0000000000001111
Remainder: 00000001
Quotient: 00001111_
```

Multiplication:

```
1  .MODEL SMALL
2  .STACK
3  .486
4  .DATA
5  first dd ?
6  second dd ?
7  INMSG1 db 13, 10, "Enter the first number(32 Bit): $"
8  INMSG2 db 13, 10, "Enter the second number(32 Bit): $"
9  OUTMSG db 13, 10, "Product: $"
10 .CODE
11 .STARTUP
12 ; INPUT - FIRST 32-bit NUMBER
13     ; Print 'INMSG1'
14     MOV DX, OFFSET INMSG1 ; DX <- Address of msg
15     MOV AH, 09 ; 09 - Print string to stdout w/ echo
16     INT 21H ; Interrupt - Application specific
17     ; Initialisations
18     MOV EBX, 0 ; Clear EBX
19     MOV CX, 4 ; Counter for loop
20     L1: SHL EBX, 8 ; Shift 8 bits to left
21     ; 1st DIGIT of FIRST Part
22     MOV AH, 01 ; 01 - single byte input
23     INT 21H ; Interrupt - Application specific
24     ; Compare input byte with 39H(ASCII for decimal 9)
25     CMP AL, 39H
26     JBE isDigit1 ; if input is below(less) or equal (0-9)
27     SUB AL, 37H ; if input is a letter (out of A,B,C,D,E,F)
28     isDigit1: AND AL, 0FH ; Masking
29     ; (Now, AL will contain Hex value for ASCII input.)
30     SHL AL, 4 ; Shift 4 bits to left i.e. append 0H
31     MOV BL, AL ; Store it in BL
32     ; 2nd DIGIT of FIRST Part (Same as FIRST Part)
33     MOV AH, 01
34     INT 21H
35     CMP AL, 39H
36     JBE isDigit2
37     SUB AL, 37H
38     isDigit2: AND AL, 0FH
39     ADD BL, AL ; Add new input(in AL) to BL
40     LOOP L1
41     MOV first, EBX
42     ; INPUT - SECOND 32-bit NUMBER [Same as FIRST NUMBER]
43     MOV DX, OFFSET INMSG2
44     MOV AH, 09
45     INT 21H
46     MOV EBX, 0
47     MOV CX, 4
48     L2: SHL EBX, 8
49     ; 1st DIGIT of FIRST Part
50     MOV AH, 01
51     INT 21H
52     CMP AL, 39H
53     JBE isDigit3
54     SUB AL, 37H
55     isDigit3: AND AL, 0FH
```

```

55     isDigit3: AND AL, 0FH
56     SHL AL, 4
57     MOV BL, AL
58     ; 2nd DIGIT of SECOND Part
59     MOV AH, 01
60     INT 21H
61     CMP AL, 39H
62     JBE isDigit4
63     SUB AL, 37h
64     isDigit4: AND AL, 00FH
65     ADD BL, AL
66     LOOP L2
67     ; MULTIPLICATION
68     MOV EAX, first ; Copy first num to EAX for MUL
69     MUL EBX ; Multiplication with contents of EBX is second num
70     ; Now, the product (64-bit) is stored as,
71     MOV first, EAX ; least significant 32 bits
72     MOV second, EDX ; most significant 32 bits
73     ; OUTPUT - PRINTING RESULTS
74     ; Print 'OUTMSG'
75     MOV DX, OFFSET OUTMSG
76     MOV AH, 09
77     INT 21H
78     ; Print most significant 32 bits in ASCII
79     MOV EBX, second
80     MOV CX, 4
81     L3: ROL EBX, 8
82     MOV AL, BL
83     AND AL, 0F0H
84     SHR AL, 4
85     ADD AL, 30H
86     CMP AL, 39H
87     JBE ab5
88     ADD AL, 07H
89     ab5:
90     MOV DL, AL
91     MOV AH, 02
92     INT 21H
93     MOV AL, BL
94     AND AL, 00FH
95     ADD AL, 30H
96     CMP AL, 39H
97     JBE ab6
98     ADD AL, 07H
99     ab6:
100    MOV DL, AL
101    MOV AH, 02
102    INT 21H
103    LOOP L3
104    ; Print least significant 32 bits in ASCII
105    MOV EBX, first
106    MOV CX, 4
107    L4: ROL EBX, 8

```



```

105     MOV EBX, 11730
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: 00000000000001000

```

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

Addition:

```
1  .MODEL SMALL ;assembler memory model
2  .STACK 100H
3  .DATA
4  MSG1 DB "4 DIGIT BCD SUM IS = $"
5  BCDSUML DB ?
6  BCDSUMH DB ?
7  .CODE
8  MOV AX,@DATA
9  MOV DS,AX
10 XOR AX, AX ; clear register AX
11 MOV AL,34H
12 MOV BL, 98H
13 ADD AL,BL
14 DAA; DECIMAL ADJUST AFTER ADDITION
15 MOV BCDSUML, AL
16 MOV AL,12H
17 MOV BL, 23H
18 ADC AL,BL
19 DAA; OPERATES ONLY ON AL
20 MOV BCDSUMH,AL
21 MOV AH,BCDSUMH
22 MOV AL,BCDSUML
23 PUSH AX
24 MOV DX, OFFSET MSG1
25 MOV AH, 09H
26 INT 21H ; INT STANDS FOR INTERRUPT INSTRUCTION 21H IS INTERRUPT NO FOR DOS SERVICES
27 POP AX
28 CALL DISPLAY
29 MOV AH, 4CH
30 INT 21H
31 DISPLAY PROC NEAR ; PROC IS KEYWORD FOR PROCEDURE
32 MOV CH, 04H
33 MOV CL, 04H
34 DISP1:
35 ROL AX, CL ; ROTATE LEFT 4 TIMES
36 PUSH AX ; SAVING ON STACK
37 AND AL, 0FH
38 ADD AL, 30H ; 48 IN DECIMAL
39 CMP AL, '9' ; COMPARE WITH ASCII VALUE OF 9
40 JBE DISP2
41 ADD AL, 7
42 DISP2: MOV DL, AL
43 MOV AH, 02H
44 INT 21H
45 POP AX
46 DEC CH
47 JNZ DISP1
48 RET
49 DISPLAY ENDP
50 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:

```
bcsub.asm
1  .model small
2  .386
3  .data
4  num1 DD 00000000H
5  num2 DD 00000000H
6  num3 DD 00000000H
7  msg db 10,13,"Enter the first no.:: $"
8  msg1 db 10,13,"Enter the second no.:: $"
9  msg2 db 10,13,"The Resultant sum is :: $"
10
11  .code
12  .startup
13
14  MOV AH,09
15  MOV DX,OFFSET msg
16  INT 21H
17
18  MOV EBX,0
19  MOV CX,8
20
21  AGAIN: MOV AH,01 ;1ST NO. ENTERED
22  INT 21H
23  CMP AL,'A'
24  JGE L2
25  SUB AL,30H
26  SHL EBX,4
27  ADD BL,AL
28  LOOP AGAIN
29
30  MOV num1,EBX
31
32  MOV AH,09
33  MOV DX,OFFSET msg1
34  INT 21H
35
36  MOV EBX,0
37  MOV CX,8
38  AGAIN1:MOV AH,01 ;2nd NO. ENTERED
39  INT 21H
40  CMP AL,'A'
41  JGE L2
42  SUB AL,30H
43  SHL EBX,4
44  ADD BL,AL
45  LOOP AGAIN1
```

```

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 bcsub.asm
Turbo Assembler Version 2.51 Copyright (c) 1988, 1991 Borland International

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

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

C:\TASM>bcsub

Enter the first no.: 11111111
Enter the second no.: 00000000
The Resultant sum is :: 11111111
C:\TASM>
```

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

Linear Search:

```
lins.asm
1  ;Linear Search
2  .model small
3  .386
4  .data
5  ARRAY DW 20 DUP (?)
6  DATA1 dw 0000H
7  success db 10,13,"Element is present in the array $"
8  fail db 10,13,"Element is not present in the array $"
9  msg db 10,13,"Enter the size of the array :: $"
10 msg2 db 10,13,"Enter the array :: $"
11 msg3 db 10,13,"Enter the element to be searched :: $"
12 .code
13 .startup
14 MOV AH,09
15 MOV DX,OFFSET msg
16 INT 21H
17
18 MOV AH,01
19 INT 21H
20 SUB AL,30H
21 MOV AH,0
22 MOV CX,AX
23
24 MOV DATA1,AX
25
26 MOV AH,09
27 MOV DX,OFFSET msg2
28 INT 21H
29 MOV AH,0
30 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
72

```

Output:

```

C:\TASM>link
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:

```
1  .model small
2  .386
3  .data
4  ARRAY DW 20 DUP (?)
5  DATA1 dw 0000H
6  DATA2 dw 0000H
7  success db 10,13,"Element is present in the array $"
8  fail db 10,13,"Element is not present in the array $"
9  msg db 10,13,"Enter the size of the array :: $"
10 msg2 db 10,13,"Enter the array :: $"
11 msg3 db 10,13,"Enter the element to be searched :: $"
12 .code
13 .startup
14 MOV AH,09
15 MOV DX,OFFSET msg
16 INT 21H
17 MOV AH,01
18 INT 21H
19 SUB AL,30H
20 MOV AH,0
21 MOV CX,AX
22 MOV DATA1,AX
23
24 MOV AH,09
25 MOV DX,OFFSET msg2
26 INT 21H
27 MOV AH,0
28 MOV SI, 0
29 MOV BX, OFFSET ARRAY
30 L1: MOV DL, 0AH ; jump onto next line
31 MOV AH, 02H
32 INT 21H
33 MOV DX, SI ; input element of the array
34 MOV AH, 01H
35 INT 21H
36 SUB AL,30H
37 MOV SI, DX
38 MOV [BX + SI], AX
39 INC SI
40 LOOP L1
41
42 MOV AH,09
43 MOV DX,OFFSET msg3
44 INT 21H
45 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
1  .model small
2  .data
3      array db 8 dup(?)
4      msg db 0dh,0ah,'Program for conversion of binary to ascii:$'
5      msg1 db 0dh,0ah,'Enter the element to array:$'
6
7  .code
8  .startup
9
10 mov dx,offset msg
11 mov ah,09h
12 int 21h
13 mov dx,offset msg1
14 mov si,0
15 mov cx,8
16 again:
17     mov ah,01h
18     int 21h
19     sub al,30h
20     mov array[si],al
21     inc si
22 loop again
23
24 mov cx,8
25 mov al,01h
26 mov sp,0h
27 mov si,07h
28 mov bl,02h
29 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:010000000

```

Q10: Write a program for ascii to binary conversion.

```
atob.asm
1  ;6. Write a program for ascii to binary conversion.
2  .model small
3  .data
4      msg db 0dh,0ah,'Program for converting ASCII to Binary:$'
5      msg1 db 0dh,0ah,'Enter the element :$'
6
7  .code
8  .startup
9
10     mov dx,offset msg
11     mov ah,09h
12     int 21h
13     mov dx,offset msg1
14     mov ah,09h
15     int 21h
16     mov ah,01h
17     int 21h
18     mov bl,al
19     mov dl,0Ah
20     mov ah,02h
21     int 21h
22     mov cx,8
23     again:
24         shl bl,1
25         jc 12
26         jnc 13
27         loop again
28     12:
29         mov dl,31h
30         mov ah,02h
31         int 21h
32         jmp 14
33     13:
34         mov dl,30h
35         mov ah,02h
36         int 21h
37         jmp 14
38     14:
39         loop again
40
41     mov ah,4ch
42     int 21h
43     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
```