SECTION – 3

ASSEMBLY LANGUAGE PROGRAMMING

Session 3 & 4 – Simple Assembly Programs

Ex 1: Write a program to add two numbers present in two consecutive memory locations and store the result in next memory location.

Ans:

```
Prg(add2num.asm)
Title add two numbers in consecutive memory location
dosseg
.model small
.stack
.data
msg1 db 13,10,"Sum of two numbers stored in memory:$"
num1 db 20h
num2 db 15h
sum db ?
res db 20 DUP('$')
.code
main proc
mov ax,@data
mov ds,ax
mov al,num1
add al,num2
mov sum,al
lea dx,msg1
mov ah,09h
int 21h
mov dl,summov ah,02hint 21h
mov ax,4c00h
int 21h
main endp
end
```

Output:

```
Sum of two numbers stored in memory:5
```

Ex 2: Develop program to read a character from console and echo it.

Ans:

```
Prg(rdecho.asm)
Title read a character from console and echo it.
dosseg
.model small
.stack
.data
msg1 db 13,10,"Enter a character:$"
msg2 db 13,10,"Read a character from console and echo:$"
.code
main proc
mov ax,@data
mov ds,ax
lea dx,msg1
mov ah,09h
int 21h
mov ah,01h
int 21h
mov bl,al
lea dx,msg2
mov ah,09h
int 21h
mov dl,bl
mov ah,02h
int 21h
```

```
mov ax,4c00h
int 21h
main endp
end
```

Output:

```
Enter a character:w
Read a character from console and echo:w
```

Ex 3: Develop and execute a program to read 10 chars from console.

Ans:

```
Prg(rd10chr.asm)
Title read a 10 character from console.
dosseg
.model small
.stack
.data
msg1 db 13,10,"Enter a 10 character:$"
.code
main proc
mov ax,@data
mov ds,ax
lea dx,msg1
mov ah,09h
int 21h
mov cx,00
mov cl,10
rpt: mov ah,01h
int 21h
mov bl,al
loop rpt
mov ax,4c00h
int 21h
main endp
end
```

Output:

```
Enter a 10 character:1234567890
```

Ex 4: Write a program to exchange two memory variables using MOV and XCHG instruction. Can you do it with just XCHG?

Ans:

```
Prg(XCHGin.asm)
     Title to exchange two memory variables using MOV and XCHG instruction
dosseg
     .model small
     .stack
 .data
     msg1 db 13,10,"First value in memory:$"
     msg2 db 13,10,"Second value in memory:$"
     msg3 db 13,10,"After using XCHG instruction:$"
     msg4 db 13,10,"First value in memory:$"
     msg5 db 13,10,"Second value in memory:$"
     value1 db 35h
     value2 db 32h
     .code
     main proc
     mov ax,@data
     mov ds,ax
     lea dx,msg1
     mov ah,09h
     int 21h
     mov dl,value1
     mov ah,02h
     int 21h
```

```
lea dx,msg2
mov ah,09h
int 21h
mov dl,value2
mov ah,02h
int 21h
lea dx,msg3
mov ah,09h
int 21h
;exchanging the value
mov al,value1
XCHG value2,al
mov value1,al
lea dx,msg4
mov ah,09h
int 21h
mov dl,value1
mov ah,02h
int 21h
lea dx,msg5
mov ah,09h
int 21h
mov dl,value2
mov ah,02h
int 21h
main endp
end
```

Output:

```
First value in memory:5
Second value in memory:2
After using XCHG instruction:
First value in memory:2
Second value in memory:5
```

Ex 6: Write a program, which will read two decimal numbers, then multiply them together, and finally print out the result (in decimal).

Ans:

```
data segment
   ms1 db 13,10,"ENTER FIRST NO    :$"
   ms2 db 13,10,"ENTER SECOND NO   :$"
   ms3 db 13,10,"MULTIPLICATION IS :$"
data ends

code segment
      assume cs:code,ds:data
        start:
      mov ax,data
      mov ds,ax
      mov ah,09h
      mov dx,offset ms1
      int 21h

      mov ah,01h
      int 21h
      mov cl,al
      and cl,0fh

      mov ah,09h
      mov dx,offset ms2
      int 21h

      mov ah,01h
      int 21h
      and al,0fh

      mul cl
```

```
        aam

        mov bx,ax
        or bx,3030h

        mov ah,09h
        mov dx,offset ms3
        int 21h

        mov dl,bh
        mov ah,02h
        int 21h

        mov dl,bl
        mov ah,02h
        int 21h

        mov ah,4ch
        int 21h

    code ends
       end start
    output-
    multiplication upto 9 * 9 = 81
```

Ex 7: Write a program to convert the ASCII code to its BCD equivalent.

Ans:

```
    Prg(pkdbcd.asm)
    Title convert the ASCII code to bcd equivalent
dosseg
    .model small
    .stack
 .data
    msg1 db 13,10,"Enter the first number:$"
    msg3 db 13,10,"Result of packed bcd:$"
    bcd db ?
    first db ?
    sec db ?
    res db 20 DUP('$')
    .code
 main proc
    mov ax,@data
    mov ds,ax
    lea dx,msg1
    mov ah,09h
    int 21h
    mov ax,00
    mov ah,01h
    int 21h
    sub al,'0'
    mov bl,al
    mov ax,00
    mov ah,01h
    int 21h
    sub al,'0'
    and bl,0Fh
    and al,0Fh
    mov cl,04h
    rol bl,cl
    or al,bl
    mov bcd,al
    lea dx,msg3
    mov ah,09h
    int 21h
    mov dx,00
    mov dl,bcd
```

```
mov ah,02h
int 21h
mov ax,4c00h
int 21h
main endp
end
```

OUTPUT:

```
Enter first number:35
Result of packed bcd:05
```

Ex 8: Write a program, which will read in two decimal inputs and print out their sum, in decimal.

Ans:

```
Prg(desum.asm)
Title read 2 decimal number and print there sum
dosseg
.model small
.stack
.data
msg1 db 13,10,"Enter first number:$"
msg2 db 13,10,"Enter second number:$"
msg3 db 13,10,"Sum in decimal number:$"
num1 db ?
sum db ?
res db 20 DUP('$')
.code
main proc
mov ax,@data
mov ds,ax
lea dx,msg1
mov ah,09h
int 21h
mov ah,01h
int 21h
sub al,'0'
mov num1,al
lea dx,msg2
mov ah,09h
int 21h
mov ah,01h
int 21h
sub al,'0'
add al,num1
mov sum,al
lea dx,msg3
mov ah,09h
int 21h
mov si,offset res
mov ax,00
mov al,sum
call hex2asc
lea dx,res
mov ah,09h
int 21h
mov ax,4c00h
int 21h
main endp
hex2asc proc near
push ax
push bx
push cx
push dx
push si
```

```
        mov cx,00h
        mov bx,0Ah
        rpt1: mov dx,00
        div bx
        add dl,'0'
        push dx
        inc cx
        cmp ax,0Ah
        jge rpt1
        add al,'0'
        mov [si],al
    rpt2: pop ax
        inc si
        mov [si],al
        loop rpt2
        inc si
        mov al,'$'
        mov [si],al
        pop si
        pop dx
        pop cx
        pop bx
        pop ax
        ret
        hex2asc endp
        end
```

OUTPUT:

```
Enter first number:2
Enter second number:3
Sum in decimal number:05
Enter first number:5
Enter second number:6
Sum in decimal number:11
```

Ex 9: Write a program, which will read in two decimal inputs and print out the smaller of the two, in decimal.

Ans:

```
Prg(desmall.asm)
    Title read in two decimal inputs and print out the smaller of the two, in decimal
dosseg
    .model small
    .stack
 .data
    msg1 db 13,10,"Enter the first number:$"
    msg2 db 13,10,"Enter the second number:$"
    msg3 db 13,10,"Smaller of two in decimal:$"
    num1 db ?
    small db ?
    res db 20 DUP('$')
    .code
main proc
    mov ax,@data
    mov ds,ax
    lea dx,msg1
    mov ah,09h
    int 21h
    mov ah,01h
    int 21h
    sub al,'0'
    mov num1,al
    lea dx,msg2
    mov ah,09h
    int 21h
    mov ah,01h
```

```
        int 21h
        sub al,'0'
        cmp al,num1
        jb sma
        mov bl,num1
        mov small,bl
        jmp prin
        sma :mov small,al
        prin:lea dx,msg3
        mov ah,09h
        int 21h
        mov si,offset res
        mov ax,00
        mov al,small
        call hex2asc
        lea dx,res
        mov ah,09h
        int 21h
        mov ax,4c00h
        int 21h
        main endp
        hex2asc proc near
        push ax
        push bx
        push cx
        push dx
        push si
        mov cx,00h
        mov bx,0Ah
        rpt1: mov dx,00
        div bx
        add dl,'0'
        push dx
        inc cx
        cmp ax,0Ah
        jge rpt1
        add al,'0'
        mov [si],al
    rpt2: pop ax
        inc si
        mov [si],al
        loop rpt2
        inc si
        mov al,'$'
        mov [si],al
        pop si
        pop dx
        pop cx
        pop bx
        pop ax
        ret
        hex2asc endp
        end
        OUTPUT:
        Enter the first number:5
        Enter the second number:2
        Smaller of two in decimal:02
        Enter the first number:8
        Enter the second number:9
        Smaller of two in decimal:08
```

Ex 10: Write a program to calculate the average of three given numbers stored in memory.

Ans: **Prg(avgthree.asm)**

```
        Title calculate average of three given numbers stored in memory
    dosseg
        .model small
        .stack
    .data
        msg1 db 13,10,"Sum of three numbers stored in memory:$"
        msg2 db 13,10,"Average of three numbers stored in memory:$"
        num1 db 10h
        num2 db 10h
        num3 db 10h
        sum db ?
        avg db ?
        res db 20 DUP('$')
        .code
    main proc
        mov ax,@data
        mov ds,ax
        mov al,num1
        add al,num2
        add al,num3
        mov sum,al
        lea dx,msg1
        mov ah,09h
        int 21h
        mov dl,summov ah,02hint 21h
        mov al,sum
        mov ah,00h
        mov bl,03
        div bl
        mov avg,al
        lea dx,msg2
        mov ah,09h
        int 21h
        mov dl,avg
        mov ah,02h
        int 21h
        mov ax,4c00h
        int 21h
        main endp
        end
        OUTPUT:
        Sum of three numbers stored in memory:0
        Average of three numbers stored in memory:►
```

Ex 11: Write a program in 8086 assembly language to find the volume of sphere using following formula:

$V = 4/3\pi r^3$

Ans:

```
Prg(volsph.asm)
        Title volume of sphere:
    dosseg
        .model small
        .stack
    .data
        msg1 db 13,10,"Enter the radius:$"
        msg2 db 13,10,"Volume of sphere is:$"
        num db ?
        rad dw ?
        pi dw ?
        result dw ?
        res db 10 DUP('$')
        .code
    main proc
        mov ax,@data
        mov ds,ax
```

```
        lea dx,msg1
        mov ah,09h
        int 21h
        call readnum
        mov cx,2
        mov ax,00
        mov al,num
        mov bx,00
        mov bl,num
  rpt:  mov dx,00
        mul bl
        loop rpt
        mov rad,ax
        mov ax,00
        mov ax,22
        mov bx,00
        mov bx,7
        cwd
        mov dx,00
        div bx
        mov pi,ax
        mov ax,00
        mov ax,rad
        mov bx,00
        mov bx,4
        mov dx,00
        mul bx
        mov result,ax
        mov ax,00
        mov ax,result
        mov bx,pi
        mov dx,00
        mul bx
        mov result,ax
        mov bx,00
        mov bx,3
        cwd
        mov ax,00
        mov ax,result
        mov dx,00
        div bx
        mov result,ax
        mov si,offset res
        call hex2asc
        lea dx,msg2
        mov ah,09h
        int 21h
        lea dx,res
        mov ah,09h
        int 21h
        mov ax,4c00h
        int 21h
        main endp
readnum proc near
        mov ah,01h
        int 21h
        sub al,'0'
        mov bh,0Ah
        mul bh
        mov num,al
        mov ah,01h
        int 21h
        sub al,'0'
```

```
        add num,al
        ret
        readnum endp
        hex2asc proc near
        push ax
        push bx
        push cx
        push dx
        push si
        mov cx,00h
        mov bx,0Ah
        rpt1: mov dx,00
        div bx
        add dl,'0'
        push dx
        inc cx
        cmp ax,0Ah
        jge rpt1
        add al,'0'
        mov [si],al
        rpt2: pop ax
        inc si
        mov [si],al
        loop rpt2
        inc si
        mov al,'$'
        mov [si],al
        pop si
        pop dx
        pop cx
        pop bx
        pop ax
        ret
        hex2asc endp
        end
```

Output:

```
Enter the radius:02
Volume of sphere is:32
Enter the radius:04
Volume of sphere is:256
```

Ex 13: Write a program to convert Centigrade (Celsius) to Fahrenheit temperature measuring scales. Using formula: Celsius = (Fahrenheit - 32) * 5 / 9

Ans:

```
Prg(farcel.asm)
    Title convert temperature celsius to Farenheit:
dosseg
    .model small
    .stack
.data
    msg1 db 13,10,"Enter a number to find fahrenheit temperature:$"
    msg2 db 13,10,"Fahrenheit Temperature is:$"
    num db ?
    res db 10 DUP('$')
    .code
 main proc
    mov ax,@data
    mov ds,ax
    lea dx,msg1
    mov ah,09h
    int 21h
    call readnum
    mov bx,00
    mov bx,9
```

```
        mov ax,00
        mov al,num
        mov dx,00
        mul bx
        mov bx,5
        cwd
        div bx
        add ax,32
        mov si,offset res
        call hex2asc
        lea dx,msg2
        mov ah,09h
        int 21h
        lea dx,res
        mov ah,09h
        int 21h
        mov ax,4c00h
        int 21h
        main endp
readnum proc near
        mov ah,01h
        int 21h
        sub al,'0'
        mov bh,0Ah
        mul bh
        mov num,al
        mov ah,01h
        int 21h
        sub al,'0'
        add num,al
        ret
        readnum endp
        hex2asc proc near
        push ax
        push bx
        push cx
        push dx
        push si
        mov cx,00h
        mov bx,0Ah
        rpt1: mov dx,00
        div bx
        add dl,'0'
        push dx
        inc cx
        cmp ax,0Ah
        jge rpt1
        add al,'0'
        mov [si],al
        rpt2: pop ax
        inc si
        mov [si],al
        loop rpt2
        inc si
        mov al,'$'
        mov [si],al
        pop si
        pop dx
        pop cx
        pop bx
        pop ax
        ret
        hex2asc endp
        end
```

Output:

```
Enter a number to find fahrenheit temperature:28
Fahrenheit Temperature is:82
Enter a number to find fahrenheit temperature:40
Fahrenheit Temperature is:104
```

Ex 14: Write a Program which adds the sales tax in the Price list of items and replace the Price list with a new list.

Ans:

```
Prg(saltax.asm)
     Title adds the sales tax in the price list of items and replace price list with a new list:
dosseg
     .model small
     .stack
 .data
     msg1 db 13,10,"How many numbers:$"
     msg2 db 13,10,"Enter number between 1 to 99:$"
     msg3 db 13,10,"Enter Price:$"
     msg4 db 13,10,"Sales tax 2 rupes for less then 100 rupees:$"
     msg5 db 13,10,"After add sales tax price list is:$"
     msg6 db 13,10,"Price number is:$"
     ntable db 100 DUP(0)
     num db ?
     temp db ?
     res db 20 DUP('$')
     .code
     main proc
     mov ax,@data
     mov ds,ax
     lea dx,msg1
     mov ah,09h
     int 21h
     call readnum
     lea dx,msg2
     mov ah,09h
     int 21h
     ;read all numbers
     mov si,offset ntable
     mov ch,00
     mov cl,num
nread:lea dx,msg3
     mov ah,09h
     int 21h
     call readnum1
     mov al,temp
     mov [si],al
     inc si
     loop nread
     mov si,offset ntable
     mov cx,00
     mov cl,num
  sl: mov ax,00
     mov al,[si]
     add al,2
     mov [si],al
     inc si
     loop sl
     lea dx,msg4
     mov ah,09h
     int 21h
     lea dx,msg5
     mov ah,09h
     int 21h
```

```
        mov cx,00
        mov cl,num
        mov si,offset res
        mov di,offset ntable
rpt:    mov ax,00
        mov al,[di]
        call hex2asc
        lea dx,msg6
        mov ah,09h
        int 21h
        lea dx,res
        mov ah,09h
        int 21h
        inc di
        loop rpt
        mov ax,4c00h
        int 21h
        main endp
        readnum proc near
        mov ah,01h
        int 21h
        sub al,'0'
        mov bh,0Ah
        mul bh
        mov num,al
        mov ah,01h
        int 21h
        sub al,'0'
        add num,al
        ret
        readnum endp
readnum1 proc near
        mov ah,01h
        int 21h
        sub al,'0'
        mov bh,10
        mul bh
        mov temp,al
        mov ah,01h
        int 21h
        sub al,'0'
        add temp,al
        ret
        readnum1 endp
        hex2asc proc near
        push ax
        push bx
        push cx
        push dx
        push si
        mov cx,00h
        mov bx,0Ah
        rpt1: mov dx,00
        div bx
        add dl,'0'
        push dx
        inc cx
        cmp ax,0Ah
        jge rpt1
        add al,'0'
        mov [si],al
rpt2:   pop ax
        inc si
```

```
        mov [si],al
        loop rpt2
        inc si
        mov al,'$'
        mov [si],al
        pop si
        pop dx
        pop cx
        pop bx
        pop ax
        ret
        hex2asc endp
        end
```

Output:

```
How many numbers:04
Enter number between 1 to 99:
Enter Price:11
Enter Price:22
Enter Price:33
Enter Price:44
Sales tax 2 rupes for less then 100 rupees:
After add sales tax price list is:
Price number is:13
Price number is:24
Price number is:35
Price number is:46
```

Session 5, 6 & 7 – Loop And Comparisons

Ex 1: Write a program to find the factorial of decimal number given by user.

Ans:

```
        Prg(fact.asm)
        Title factorial of a given number
    dosseg
        .model small
        .stack
     .data
        msg1 db 13,10,"Enter a number to find factorial:$"
        msg2 db 13,10,"Factorial of given number is:$"
        num db ?
        res db 10 DUP('$')
        .code
    main proc
        mov ax,@data
        mov ds,ax
        lea dx,msg1
        mov ah,09h
        int 21h
        call readnum
        mov ax,01
        mov ch,00
        mov cl,num
        cmp cx,00
        je skip
    rpt: mov dx,00
        mul cx
        loop rpt
        skip:mov si,offset res
        call hex2asc
        lea dx,msg2
        mov ah,09h
        int 21h
```

```
        lea dx,res
        mov ah,09h
        int 21h
        mov ax,4c00h
        int 21h
        main endp
        readnum proc near
        mov ah,01h
        int 21h
        sub al,'0'
        mov bh,0Ah
        mul bh
        mov num,al
        mov ah,01h
        int 21h
        sub al,'0'
        add num,al
        ret
        readnum endp
        hex2asc proc near
        push ax
        push bx
        push cx
        push dx
        push si
        mov cx,00h
        mov bx,0Ah
        rpt1: mov dx,00
        div bx
        add dl,'0'
        push dx
        inc cx
        cmp ax,0Ah
        jge rpt1
        add al,'0'
        mov [si],al
    rpt2: pop ax
        inc si
        mov [si],al
        loop rpt2
        ---------
        inc si
        mov al,'$'
        mov [si],al
        pop si
        pop dx
        pop cx
        pop bx
        pop ax
        ret
        hex2asc endp
        end
```

Output:

```
Enter a number to find factorial:03
Factorial of given number is:06
Enter a number to find factorial:05
Factorial of given number is:120
```

Ex 4: Write a program, which will read in decimal inputs repeatedly until a zero value is read; at this point, it should print out the sum of the numbers read in so far.

Ans: **Prg(sum0.asm)**

```
        Title read decimal inputs repeatedly until a zero value is read and print sum of the numbers
        read in so far:
        dosseg
        .model small
        .stack
    .data
        msg1 db 13,10,"Enter number and get the sum untill 00 is read:$"
        msg2 db 13,10,"Enter number:$"
        msg3 db 13,10,"Sum is:$"
        num db ?
        temp db ?
        res db 10 DUP('$')
        .code
        main proc
        mov ax,@data
        mov ds,ax
        lea dx,msg1
        mov ah,09h
        int 21h
        ;read numbers
        mov ax,00
        mov temp,al
   read: lea dx,msg2
        mov ah,09h
        int 21h
        call readnum
        mov al,num
        cmp al,00
        je ou
        mov ax,00
        mov al,temp
        add al,num
        mov temp,al
        mov ax,00
        mov al,temp
        mov si,offset res
        call hex2asc
        lea dx,msg3
        mov ah,09h
        int 21h
        lea dx,res
        mov ah,09h
        int 21h
        mov ax,00
        mov al,temp
        jmp read
        ou: mov ax,4c00h
        int 21h
        main endp
        readnum proc near
        mov ah,01h
        int 21h
        sub al,'0'
        mov bh,0Ah
        mul bh
        mov num,al
        mov ah,01h
        int 21h
        sub al,'0'
        add num,al
        ret
        readnum endp
        hex2asc proc near
```

```
        push ax
        push bx
        push cx
        push dx
        push si
        mov cx,00h
        mov bx,0Ah
        rpt1: mov dx,00
        div bx
        add dl,'0'
        push dx
        inc cx
        cmp ax,0Ah
        jge rpt1
        add al,'0'
        mov [si],al
    rpt2: pop ax
        inc si
        mov [si],al
        loop rpt2
        inc si
        mov al,'$'
        mov [si],al
        pop si
        pop dx
        pop cx
        pop bx
        pop ax
        ret
        hex2asc endp
        end
        Output:
        Enter number and get the sum untill 00 is read:
        Enter number:11
        Sum is:11
        Enter number:22
        Sum is:33
        Enter number:33
        Sum is:66
        Enter number:44
        Sum is:110
        Enter number:00
```

Ex 5: Develop and execute an assembly language program to find the LCM of two 16-bit unsigned integers.

Ans:

```
        Prg(lcm16.asm)
        Title program to find lcm of two 16 bit unsigned integers.
    dosseg
        .model small
        .stack
    .data
        cr equ 0dh
        lf equ 0ah
        msg db cr,lf,"Program for LCM of two positive Integers..:$"
        msg1 db cr,lf,"Enter numbe1:$"
        msg2 db cr,lf,"Enter number2:$"
        msg3 db cr,lf,"LCM=:$"
        num1 dw ?
        num2 dw ?
        gcd dw ?
        num3 dw ?
        lcm dw ?
        res db 10 DUP(0)
```

```
        buff db 80
             db 0
             db 80 DUP(?)
        .code
        main proc
        mov ax,@data
        mov ds,ax
        mov ah,09h
        mov dx,offset msg
        int 21h
        ;Read number1
        mov ah,09h
        mov dx,offset msg1
        int 21h
        call readinteger
        ;Read number2
        mov ah,09h
        mov dx,offset msg2
        int 21h
        call readinteger1
        ;push num1 and num2 into stack
        mov ax,num1
        push ax
        mov ax,num2
        push ax
        call findgcd
        add sp,4
        ;adjust stack pointer
        mov gcd,ax
        ;gcd = findgcd(num[i],num[i+1])
        ;LCM = (num1*num2)/gcd(num1,num2)
        mov ax,num1
        mov dx,00
        mul num2
        div gcd
        mov lcm,ax
        ;print LCM
        mov ah,09h
        mov dx,offset msg3
        int 21h
        mov ax,lcm
        mov si,offset res
        call hex2asc
        mov ah,09h
        mov dx,offset res
        int 21h
        mov ax,4c00h
        int 21h
        main endp
readinteger proc near
        push dx
        push bx
        push ax
        mov ah,0ah
        mov dx,offset buff
        int 21h
        mov bx,offset buff
        add bx,2
        push bx
        call atoi
        pop bx
        mov num1,ax
        pop ax
        pop bx
```

```
        pop dx
        ret
        readinteger endp
        readinteger1 proc near
        push dxpush bxpush ax
        mov ah,0ah
        mov dx,offset buff
        int 21h
        mov bx,offset buff
        add bx,2
        push bx
        call atoi
        pop bx
        mov num2,ax
        pop ax
        pop bx
        pop dx
        ret
        readinteger1 endp
        findgcd proc near
        push bp
        mov bp,sp
        push dx
        push bx
        rpt: mov ax,[bp+4]
        mov bx,[bp+6]
        cmp ax,bx
        jl skip
        mov [bp+6],ax
        mov [bp+6],bx
  skip: mov dx,00
        mov ax,[bp+6]
        div word ptr[bp+4]
                                ;num2/num1
        mov [bp+6],dx
        cmp dx,00
        jne rpt
        mov ax,[bp+4]
        pop bx
        pop dx
        pop bp
        ret
        findgcd endp
        atoi proc near
        push bp
        mov bp,sp
        push si
        push dx
        push cx
        push bx
        mov si,[bp+4]
        ;finding the length of the string
        mov bx,00
  nxtch: mov al,[si]
        inc bx
        inc si
        cmp al,cr
        jne nxtch
        ;cx=length of the string
        mov cx,bx
        dec cx
        ;si is pointing outside the string so adjust
        dec si
```

```
        mov dx,00
        mov bx,01
        nxt: dec si
        push dx
        ;dx:ax=digit
        xor dx,dx
        mov ah,00
        mov al,[si]
        sub al,'0'
        mul bx
        pop dx
        add dx,ax
        ;generate multiples bx=10,100,1000....
        push dx
        push cx
        xor dx,dx
        mov cx,10
        mov ax,bx
        mul cx
        mov bx,ax
        pop cx
        pop dx
        loop nxt
        mov ax,dx
        pop bx
        pop cx
        pop dx
        pop si
        pop bp
        ret
        atoi endp
        hex2asc proc near
        push ax
        push bx
        push cx
        push dx
        push si
        mov cx,00h
        mov bx,0Ah
        rpt1: mov dx,00
        div bx
        add dl,'0'
        push dx
        inc cx
        cmp ax,0Ah
        jge rpt1
        add al,'0'
        mov [si],al
    rpt2: pop ax
        inc si
        mov [si],al
        loop rpt2
        inc si
        mov al,'$'
        mov [si],al
        pop si
        pop dx
        pop cx
        pop bx
        pop ax
        ret
        hex2asc endp
        end
```

Output:

```
Program for LCM of two positive Integers..:
Enter numbe1:150
Enter number2:75
LCM=:150
```

Ex 7: Develop and execute a program to sort a given set of 8-bit unsigned integers into ascending order.

Ans:

```
Prg(ascor.asm)
Title sort(bubble sort) an given array element in ascending order
dosseg
    .model small
    .stack
 .data
    msg1 db 13,10,"How many numbers:$"
    msg2 db 13,10,"Enter number:$"
    msg3 db 13,10,"Sorted elements in ascending order are:$"
    msg4 db 13,10,"Element:$"
    ntable db 100 DUP(0)
    num db ?
    temp db ?
    count db ?
    res db 10 DUP('$')
    .code
    main proc
    mov ax,@data
    mov ds,ax
    lea dx,msg1
    mov ah,09h
    int 21h
    call readnum
    ;read all numbers
    mov si,offset ntable
    mov ch,00
    mov cl,num
nread:lea dx,msg2
    mov ah,09h
    int 21h
    call readnum1
    mov al,temp
    mov [si],al
    inc si
    loop nread
    ;sorting an array elements
    mov cx,00
    mov cl,num
    cmp cx,01
    ;if(num=01)then print array elements
    je pnxt1
    nxtps:mov dx,00
    ;flag =false
    mov bx,00
    ;j=1
nxtj: mov al,ntable[bx]
    mov ah,ntable[bx+1]
    cmp ah,0
    je skip
    cmp al,ah
    jle skip
    mov ntable[bx],ah
    mov ntable[bx+1],al
    mov dl,01
```

```
    skip: inc bx
          cmp bx,cx
          jl nxtj
          dec cx
          jz pnxt1
          cmp dl,01h
          je nxtps
          ;print array elements
    pnxt1:mov ch,00
          mov cl,num
          mov di,offset ntable
          mov si,offset res
          lea dx,msg3
          mov ah,09h
          int 21h
    pnxt: lea dx,msg4
          mov ah,09h
          int 21h
          mov ah,00
          mov al,[di]
          call hex2asc
          lea dx,res
          mov ah,09h
          int 21h
          inc di
          loop pnxt
          mov ax,4c00h
          int 21h
          main endp
  readnum proc near
          mov ah,01h
          int 21h
          sub al,'0'
          mov bh,0Ah
          mul bh
          mov num,al
          mov ah,01h
          int 21h
          sub al,'0'
          add num,al
          ret
          readnum endp
 readnum1 proc near
          mov ah,01h
          int 21h
          sub al,'0'
          mov bh,0Ah
          mul bh
          mov temp,al
          mov ah,01h
          int 21h
          sub al,'0'
          add temp,al
          ret
          readnum1 endp
          hex2asc proc near
          push ax
          push bx
          push cx
          push dx
          push si
          mov cx,00h
          mov bx,0Ah
          rpt1: mov dx,00
```

```
        div bx
        add dl,'0'
        push dx
        inc cx
        cmp ax,0Ah
        jge rpt1
        add al,'0'
        mov [si],al
    rpt2: pop ax
        inc si
        mov [si],al
        loop rpt2
        inc si
        mov al,'$'
        mov [si],al
        pop si
        pop dx
        pop cx
        pop bx
        pop ax
        ret
        hex2asc endp
        end
```

Output:

```
How many numbers:04
Enter number:04
Enter number:03
Enter number:02
Enter number:01
Sorted elements in ascending order are:
Element:01
Element:02
Element:03
Element:04
```

Ex 11: Write a program to Convert ASCII number into decimal digit.

Ans:

```
        Prg(ascdec.asm)
        Title convert ASCII to decimal digit
        dosseg
        .model small
        .stack
    .data
        msg1 db 13,10,"Enter a number:$"
        msg2 db 13,10,"Decimal number is:$"
        num db ?
        res db 10 DUP('$')
        .code
    main proc
        mov ax,@data
        mov ds,ax
        lea dx,msg1
        mov ah,09h
        int 21h
        call readnum
    skip:mov si,offset res
        mov ax,00
        mov al,num
        call hex2asc
        lea dx,msg2
        mov ah,09h
        int 21h
```

```
        lea dx,res
        mov ah,09h
        int 21h
        mov ax,4c00h
        int 21h
        main endp
readnum proc near
        mov ah,01h
        int 21h
        sub al,'0'
        mov bh,0Ah
        mul bh
        mov num,al
        mov ah,01h
        int 21h
        sub al,'0'
        add num,al
        ret
        readnum endp
        hex2asc proc near
        push ax
        push bx
        push cx
        push dx
        push si
        mov cx,00h
        mov bx,0Ah
        rpt1: mov dx,00
        div bx
        add dl,'0'
        push dx
        inc cx
        cmp ax,0Ah
        jge rpt1
        add al,'0'
        mov [si],al
 rpt2: pop ax
        inc si
        mov [si],al
        loop rpt2
        inc si
        mov al,'$'
        mov [si],al
        pop si
        pop dx
        pop cx
        pop bx
        pop ax
        ret
        hex2asc endp
        end
```

Output:

```
Enter a number:12
Decimal number is:12
```

Ex 16: Write a Program, which should adds two 5-byte numbers (numbers are stored in array- NUM1 & NUM2), and stores the sum in another array named RESULT.

Ans:

```
Prg(ad5bnm.asm)
    Title add 5 byte numbers(num1 and num2 array) and stores the sum array named
    RESULT
dosseg
    .model small
    .stack
```

```
.data
    len equ 05h
    msg db 13,10,"To calculate sum of 5 byte number stored in memory.....$"
    msg1 db 13,10,"Element in first array.....$"
    msg2 db 13,10,"Element is:$"
    msg3 db 13,10,"Element in second array.....$"
    msg4 db 13,10,"Sum is:$"
    num1 db 31h, 32h, 33h, 34h, 35h
    num2 db 31h, 32h, 33h, 34h, 35h
    sum db 6 DUP(0)
    res db 10 DUP(0)
    .code
    main proc
    mov ax,@data
    mov ds,ax
    lea dx,msg
    mov ah,09h
    int 21h
    ;print first array element
    lea dx,msg1
    mov ah,09h
    int 21h
    mov cx,00
    mov cl,05
    mov di,00
nxt: lea dx,msg2
    mov ah,09h
    int 21h
    mov dl,num1[di]
    mov ah,02h
    int 21h
    inc di
    loop nxt
    ;print second array element
    lea dx,msg3
    mov ah,09h
    int 21h
    mov cx,00
    mov cl,05
    mov si,00
nxt1:lea dx,msg2
    mov ah,09h
    int 21h
    mov dl,num2[si]
    mov ah,02h
    int 21h
    inc si
    loop nxt1
    ;adding 2 array element
    mov si,00
    mov cx,00
    mov cl,05
    clc
again:mov al,num1[si]
    adc al,num2[si]
    mov sum[si],al
    inc si
    loop again
    rcl al,01h
    and al,01h
    mov sum[si],al
    ;printing array sum
    mov cx,00
    mov cl,06
```

```
        mov si,00
        lea dx,msg4
        mov ah,09h
        int 21h
    pnxt:mov dl,sum[si]
        mov ah,02h
        int 21h
        inc si
        loop pnxt
        mov ax,4c00h
        int 21h
        main endp
        end
```

Output:

```
To calculate sum of 5 byte number stored in memory.....
Element in first array.....
Element is:1
Element is:2
Element is:3
Element is:4
Element is:5
Element in second array.....
Element is:1
Element is:2
Element is:3
Element is:4
Element is:5
Sum is:bdfhj
```

Ex 17: Write a program which should convert 4 digits BCD number into its binary equivalent.

Ans:

```
Prg(bcdbin.asm)
        Title convert 4 digit bcd number into its binary equivalent
dosseg
    .model small
    .stack
    .datathou equ 3E8h
                                ;1000 =3E8h
    msg db 13,10,"To convert bcd number of 4 digit:$"
    msg1 db 13,10,"Stored in memory to binary equivalent:$"
    msg2 db 13,10,"Hex number for 10 is 0Ah:$"
    msg3 db 13,10,"Hex number for 100 is 64h:$"
    msg4 db 13,10,"Hex number for 1000 is 3E8h:$"
    msg5 db 13,10,"The number stored in memory is 4567h:$"
    msg6 db 13,10,"Its Hex number is 11D7h:$"
    msg7 db 13,10,"After converting bcd number to binary number:$"
    msg8 db 13,10,"Binary number is:$"
    bcd dw 4567h
    hex dw ?
    res db 40 DUP('$')
    .code
    main proc
    mov ax,@data
    mov ds,ax
    lea dx,msg
    mov ah,09h
    int 21h
    lea dx,msg1
    mov ah,09h
    int 21h
    lea dx,msg2
    mov ah,09h
    int 21h
```

```
lea dx,msg3
mov ah,09h
int 21h
lea dx,msg4
mov ah,09h
int 21h
lea dx,msg5
mov ah,09h
int 21h
lea dx,msg6
mov ah,09h
int 21h
;converting bcd to binary
mov ax,bcd
mov bx,ax
mov al,ah
mov bh,bl
mov cl,04
ror ah,cl
ror bh,cl
and ax,0F0Fh
and bx,0F0Fh
mov cx,ax
;multiplying the number by 10,100,1000 to set to there place value
mov ax,0000h
mov al,ch
mov di,thou
mul di
mov dh,00h
mov dl,bl
add dx,ax
mov ax,0064h
mul cl
add dx,ax
mov ax,000Ah
mul bh
add dx,ax
mov hex,dx
;printing the binary number
;its hex value is stored in memory
lea dx,msg7
mov ah,09h
int 21h
lea dx,msg8
mov ah,09h
int 21h
mov ax,00
mov si,offset res
mov ax,hex
call hex2asc
mov dx,offset res
mov ah,09h
int 21h
mov ax,4c00h
int 21h
main endp
hex2asc proc near
push ax
push bx
push cx
push dx
push si
mov cx,00h
mov bx,0Ah
```

```
        rpt1: mov dx,00
              div bx
              add dl,'0'
              push dx
              inc cx
              cmp ax,0Ah
              jge rpt1
              add al,'0'
              mov [si],al
        rpt2: pop ax
              inc si
              mov [si],al
              loop rpt2
              inc si
              mov al,'$'
              mov [si],al
              pop si
              pop dx
              pop cx
              pop bx
              pop ax
              ret
              hex2asc endp
              end
```

Output:

```
To convert bcd number of 4 digit:
Stored in memory to binary equivalent:
Hex number for 10 is 0Ah:
Hex number for 100 is 64h:
Hex number for 1000 is 3E8h:
The number stored in memory is 4567h:
Its Hex number is 11D7h:
After converting bcd number to binary number:
Binary number is:4567
```

Session 8 - Strings

Ex 1: Write a program, which takes two inputs as strings and display the Concatenated string.

Ans:

```
        Prg(strcon.asm)
                Title string concat
        dosseg
        .model small
        .stack
        .data
        msg1 db 13,10,"Enter a string with dolar symbol as a break:$"
        msg2 db 13,10,"Enter second string with dolar symbol as a break:$"
        msg3 db 13,10,"Concated string is:$"
        strg db 20 DUP(0)
        .code
    main proc
        mov ax,@data
        mov ds,ax
```

```
        lea di,strg
        lea dx,msg1
        mov ah,09h
        int 21h
 first:mov ah,01h
        int 21h
        cmp al,24h
        je next
       ; inc di
        mov [di],al
        inc di
        jmp first
 next: lea dx,msg2
        mov ah,09h
        int 21h
second:mov ah,01h
        int 21h
        cmp al,24h
        je con
        ; inc di
        mov [di],al
        inc di
        jmp second


 con : lea dx,msg3
        mov ah,09h
        int 21h
        lea di,strg
        dis: mov al,[di]
        cmp al,0
        je ou
        mov dl,al
        mov ah,02h
        int 21h
        inc di
        jmp dis
        ou: mov ax,4c00h
        int 21h
        main endp
        end
```

Output:

```
Enter a string with dolar symbol as a break:saint$
Enter second string with dolar symbol as a break:alosius$
Concated string is:saintalosius
```

Ex 2: Write a program, which converts string lower case characters to upper case characters and upper case characters to lower case characters.

Ans:

```
        Prg(strul.asm)
        Title convert string upper case to lower case and lower case to upper case
        dosseg
        .model small
        .stack
        .data
        msg1 db 13,10,"Enter a string with dolar symbol as a break:$"
        msg2 db 13,10,"Modified string is:$"
        buf db 80 DUP(0)
        revbuf db 80 DUP(0)
        strlen db ?
        .code
 main proc
        mov ax,@data
        mov ds,ax
```

```
        lea dx,msg1
        mov ah,09h
        int 21h
        lea si,buf
read:   mov ah,01h
        int 21h
        mov [si],al
        inc si
        cmp al,24h
        je check
        jmp read
        check:lea si,buf
        lea di,revbuf
start:  mov al,[si]
        cmp al,'$'
        je dis
        cmp al,60h
        jb lower
        cmp al,7Ah
        jb upper
        jmp start
lower:  cmp al,40h
        jb skip
        cmp al,5Ah
        jb up
  up:   add al,20h
        mov [di],al
        inc di
        inc si
        jmp start
        upper:cmp al,60h
        ja lo
        lo: sub al,20h
        mov [di],al
        inc di
        inc si
        jmp start
skip:   mov [di],al
        inc si
        inc di
        jmp start
        dis:mov al,'$'
        mov [di],al
       lea dx,msg2
        mov ah,09h
        int 21h
        lea dx,revbuf
        mov ah,09h
        int 21h
        ou:mov ax,4c00h
        int 21h
        main endp
        end
```

Output:

```
Enter a string with dolar symbol as a break:SaiNt$
Modified string is:sAInT
```

Ex 3: Write a program for reversing a given string.

Ans:

```
Prg(strrev.asm)
Title reversing a string
dosseg
.model small
```

```
        .stack
    .data
        msg1 db 13,10,"Enter a string with dolar symbol as a break:$"
        msg2 db 13,10,"Reverse of a string is:$"
        strg db 20 DUP(0)
        restr db 20 DUP(0)
        .code
    main proc
        mov ax,@data
        mov ds,ax
        mov es,ax
        mov di,00
        lea dx,msg1
        mov ah,09h
        int 21h
    read:mov ah,01h
        int 21h
        cmp al,24h
        je next
        inc di
        mov strg[di],al
        jmp read
        next: mov si,00
   start:cmp di,0
        je dmsg2
        mov al,strg[di]
        mov restr[si],al
        inc si
        dec di
        jmp start
   dmsg2:lea dx,msg2
        mov ah,09h
        int 21h
    dis:mov al,restr[di]
        cmp al,0
        je ou
        mov dl,al
        mov ah,02h
        int 21h
        inc di
        jmp dis
        ou: mov ax,4c00h
        int 21h
        main endp
        end
```

Output:

```
Enter a string with dolar symbol as a break:saint$
Reverse of a string is:tnias
```

Ex 6: Write a program to determine a given string is a palindrome. If 'Yes' output the message “The given string is a palindrome”. If 'No' output the message “No, it is not a palindrome”.

Ans:

```
        Prg(strpal.asm)
        Title string palindrome
        dosseg
        .model small
        .stack
    .data
        msg1 db 13,10,"Enter a string with dolar symbol as a break:$"
        msg2 db 13,10,"Reverse of a given string is:$"
        msg3 db 13,10,"String length is:$"
        msg4 db 13,10,"Is Palindrome:$"
        msg5 db 13,10,"Not a Palindrome:$"
        buf db 80 DUP(0)
```

```
        revbuf db 80 DUP(0)
        strlen db ?
        .code
    main proc
        mov ax,@data
        mov ds,ax
        lea dx,msg1
        mov ah,09h
        int 21h
        lea si,buf
    read: mov ah,01h
        int 21h
        mov [si],al
        inc si
        cmp al,24h
        je cou
        jmp read
        cou: lea si,buf
        mov bx,00
    count:mov al,[si]
        inc si
        ;inc bl
        cmp al,24h
        je rev
        inc bx
        jmp count
        rev: lea di,revbuf
        lea si,buf
        add si,bx
        mov cx,00
        mov cx,bx
        dec si
    revst:mov al,[si]
        mov [di],al
        dec si
        inc di
        loop revst
        lea di,revbuf
        lea si,buf
        add di,bx
        add si,bx
        mov al,[si]
        mov [di],al
     dis:lea dx,msg2
        mov ah,09h
        int 21h
        lea dx,revbuf
        mov ah,09h
        int 21h
        lea si,buf
        lea di,revbuf
        mov cx,bx
    check:mov al,[si]
        cmp [di],al
        jne pal
        inc di
        inc si
        loop check
        lea dx,msg4
        mov ah,09h
        int 21h
        jmp ou
        pal:lea dx,msg5
        mov ah,09h
```

```
int 21h
ou:mov ax,4c00h
int 21h
main endp
end
```

Output:

```
Enter a string with dolar symbol as a break:srrs$
Reverse of a given string is:srrs
Is Palindrome:
```

Ex 7: Write a program to search for a character in a given string and calculate the number of occurrences of the character in the given string.

Ans:

```
Prg(strchr.asm)
    Title count character occourence in a string
    dosseg
    .model small
    .stack
 .data
    msg1 db 13,10,"Enter a string with dolar symbol as a break:$"
    msg2 db 13,10,"Enter a character to count:$"
    msg3 db 13,10,"Number of times occoured in a given string:$"
    buf db 80 DUP(0)
    chr db 10 DUP('$')
    strlen db ?
    res db 10 DUP('$')
    .code
 main proc
    mov ax,@data
    mov ds,ax
    lea dx,msg1
    mov ah,09h
    int 21h
    mov si,offset buf
read: mov ah,01h
    int 21h
    mov [si],al
    inc si
    cmp al,24h
    je next
    jmp read
next: lea dx,msg2
    mov ah,09h
    int 21h
read1:mov si,offset chr
    mov ah,01h
    int 21h
    mov [si],al
    inc si
    mov al,24h
    mov [si],al
    mov bx,00
    mov si,offset buf
    mov ax,00
    mov di,offset chr
check:mov al,[si]
    cmp al,[di]
    je count
    cmp al,'$'
    je dis
    inc si
    jmp check
```

```
count:inc bl
    inc si
    jmp check
    dis:mov strlen,bl
    lea si,res
    mov ax,00
    mov al,strlen
    call hex2asc
    lea dx,msg3
    mov ah,09h
    int 21h
    lea dx,res
    mov ah,09h
    int 21h
    ou:mov ax,4c00h
    int 21h
    main endp
    hex2asc proc near
    push ax
    push bx
    push cx
    push dx
    push si
    mov cx,00h
    mov bx,0Ah
    rpt1: mov dx,00
    div bx
    add dl,'0'
    push dx
    inc cx
    cmp ax,0Ah
    jge rpt1
    add al,'0'
    mov [si],al
rpt2: pop ax
    inc si
    mov [si],al
    loop rpt2
    inc si
    mov al,'$'
    mov [si],al
    pop si
    pop dx
    pop cx
    pop bx
    pop ax
    ret
    hex2asc endp
    end
```

Output:

```
Enter a string with dolar symbol as a break:saintalosius$
Enter a character to count:a
Number of times occoured in a given string:02
```