**3. Addition, Subtraction, Division, Multiplication**

data segment

data1 db 04h

data2 db 02h

data3 db -6h

add1 db ?

sub1 db ?

mul1 dw ?

div1 dw ?

sigmul dw ?

sigdiv dw ?

ends

code segment

start:

mov ax,data

mov ds,ax

mov ax,0

mov al,data1

mov bl,data2

add al,bl

mov add1,al

mov al,data1

mov bl,data2

sub al,bl

mov sub1,al

mov al,data1

mov bl,data2

mul bl

mov mul1,ax

mov al,data1

mov bl,data2

div bl

mov div1,ax

mov al,data3

mov bl,data2

imul bl

mov sigmul,ax

mov al,data3

mov bl,data2

idiv bl

mov sigdiv,ax

mov dl,add1

mov dl,sub1

mov dx,mul1

mov dx,div1

mov dx,sigmul

mov dx,sigdiv

mov ax, 4c00h

int 21h

ends

end start

**4. 1’s and 2’s Complement.**

code segment

start:

mov al,23h

neg al

mov bl,57h

not bl

mov ax, 4c00h

int 21h

ends

end start

**5. Storing, Retrieving and Exchange of content in Memory**

code segment

start:

mov [500],20h

mov [600],30h

mov bl,[500]

mov bh,[600]

xchg bl,[600]

xchg bh,[500]

mov cl,[500]

mov ch,[600]

mov ax, 4c00h

int 21h

ends

end start

**6. Logical and shift Microoperation**

data segment

data1 dw 1973h

data2 dw 6735h

ends

code segment

start:

mov ax,data

mov ds,ax

mov ax,data1

mov bx,data2

and ax,bx

mov ax,data1

mov bx,data2

or ax,bx

mov ax,data1

mov bx,data2

xor ax,bx

mov ax,data1

mov bx,data2

rcl ax,1

rcr ax,1

rol ax,1

ror ax,1

sal ax,1

shl ax,1

sar ax,1

shr ax,1

mov ax, 4c00h

int 21h

ends

end start

**7. Square and Cube of Number**

data segment

data1 db 2h

sqr db ?

cub db ?

ends

code segment

start:

mov ax,data

mov ds,ax

mov ax,0

mov al,data1

mov bl,data1

mul bl

mov sqr,al

mul bl

mov cub,al

mov ax, 4c00h

int 21h

ends

end start

**8. Copy Array**

data segment

data1 db 12h,23h,42h

len db $-data1

data2 db ?

ends

code segment

start:

mov ax,data

mov ds,ax

mov ax,0

mov cl,len

mov ch,0

lea si,data1

lea di,data2

next:

mov bl,[si]

mov [di],bl

inc si

inc di

loop next

mov ax, 4c00h

int 21h

ends

end start

**9. Average of 3 Numbers**

data segment

data1 db 12h

data2 db 23h

data3 db 40h

ends

code segment

start:

mov ax,data

mov ds,ax

mov ax,0

mov al,data1

mov bl,data2

mov cl,data3

add al,bl

add al,cl

mov bl,03H

div bl

mov ax, 4c00h

int 21h

ends

end start

**10. Average of N Numbers**

data segment

data1 db 12h,23h,40h

len db $-data1

data2 db ?

ends

code segment

start:

mov ax,data

mov ds,ax

mov ax,0

mov cl,len

mov ch,0

lea si,data1

next:

mov bl,[si]

add al,bl

inc si

loop next

mov cl,len

div cl

mov ax, 4c00h

int 21h

ends

end start

**11. Largest and Smallest Number**

**Largest**

data segment

data1 db 23h,12h,56h,33h,72h,70h

len db $-data1

str db "Largest number is $"

ends

code segment

start:

mov ax,data

mov ds,ax

mov ax,0

mov cl,len

mov ch,0

lea si,data1

mov bl,0

next:

mov bh,[si]

cmp bl,bh

js next1

inc si

;dec cl

loop next

jmp next2

next1:

mov bl,bh

inc si

;dec cl

loop next

next2:

mov dx, offset str

mov ah,09

int 21h

mov al,bl

mov ah,0

mov bh,10h

div bh

mov bx,ax

add bx,3030h

mov dl,bl

mov ah,02h

int 21h

mov dl,bh

mov ah,02h

int 21h

mov ax, 4c00h

int 21h

ends

end start

**Smallest Number**

data segment

data1 db 23h,02h,56h,33h,72h,89h,08h

len db $-data1

str db "Smallest number is $"

ends

code segment

start:

mov ax,data

mov ds,ax

mov ax,0

mov cl,len

mov ch,0

lea si,data1

mov bl,99h

next:

mov bh,[si]

cmp bh,bl

jb next1

inc si

;dec cl

loop next

jmp next2:

next1:

mov bl,bh

inc si

;dec cl

loop next

next2:

mov dx, offset str

mov ah,09

int 21h

mov al,bl

mov ah,0

mov bh,10h

div bh

mov bx,ax

add bx,3030h

mov dl,bl

mov ah,02h

int 21h

mov dl,bh

mov ah,02h

int 21h

mov ax, 4c00h

int 21h

ends

end start

**12 Sort of an Array**

DATA SEGMENT

DATA1 DB 99H,12H,56H,45H,36H

LEN DB $-DATA1

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE,DS:DATA

START: MOV AX,DATA

MOV DS,AX

MOV CH,LEN

DEC CH

UP2:

MOV CL,LEN

DEC CL

LEA SI,DATA1

UP1: MOV AL,[SI]

MOV BL,[SI+1]

CMP AL,BL

JC DOWN

MOV DL,[SI+1]

XCHG [SI],DL

MOV [SI+1],DL

DOWN: INC SI

DEC CL

JNZ UP1

DEC CH

JNZ UP2

MOV CL,05H

MOV CH,0

LEA SI,DATA1

NEXT:

MOV BL,[SI]

INC SI

LOOP NEXT

HLT

CODE ENDS

END START

**13 Fibonacci Series**

data segment

len db 7 ;Change this value for different value

data1 db ?

ends

code segment

start:

mov ax,data

mov ds,ax

mov bl,00h

mov bh,01h

mov cl,len

mov ch,00h

dec cl

dec cl

lea si,data1

mov [si],bl

inc si

mov [si],bh

inc si

next:

mov al,bl

add al,bh

mov [si],al

inc si

mov bl,bh

mov bh,al

loop next

lea si,data1

mov cl,len

mov ch,00

next2:

mov al,[si]

mov ah,00h

mov bl,10h

div bl

mov bx,ax

mov dl,bl

add dl,30h

mov ah,02h

int 21h

mov dl,bh

add dl,30h

mov ah,02h

int 21h

mov dl,2ch

mov ah,02h

int 21h

inc si

loop next2

mov ax, 4c00h

int 21h

ends

end start

**14: Factorial of Number**

data segment

a db 5

data ends

code segment

assume ds:data,cs:code

start:

mov ax,data

mov ds,ax

mov ah,00

mov al,a

l1: dec a

mul a

mov cl,a

cmp cl,01

jnz l1

mov ah,4ch

int 21h

code ends

end start

**15. Search of Number**

data segment

data1 db 12h,23h,67h,34h,86h

len db $-data1

data2 db 23h

str1 db "Number found on Position:$"

str2 db "Number not Found$"

ends

code segment

start:

mov ax,data

mov ds,ax

mov ax,0

mov cl,len

mov ch,0

mov bl,data2

mov bh,01h

lea si,data1

next:

cmp bl,[si]

je eql

inc bh

inc si

loop next

lea dx,str2

mov ah,09h

int 21h

jmp ext

eql:

lea dx,str1

mov ah,09h

int 21h

mov dl,bh

mov dh,00h

add dl,30h

mov ah,02h

int 21h

jmp ext

ext:

mov ax, 4c00h

int 21h

ends

end start

**16. Hello World print**

data segment

str1 db "Hello World$"

ends

code segment

start:

mov ax,data

mov ds,ax

mov ax,0

lea dx,str1

;mov dl,

mov ah,09h

int 21h

mov ax, 4c00h

int 21h

ends

end start

**17. Copy string to other string**

data segment

str1 db "Hello"

len db $-str1

str2 db ?

ends

code segment

start:

mov ax,data

mov ds,ax

mov ax,0

mov cl,len

mov ch,0

mov SI, offset str1

mov DI, offset str2

next:

mov bl,[SI]

mov [DI],bl

inc SI

inc DI

loop next

mov [DI],24h ;to copy $ at the end to run intrrupt

lea dx,str2

;mov dl,

mov ah,09h

int 21h

mov ax, 4c00h

int 21h

ends

end start

**18. Reverse String**

data segment

str1 db "Hello World$"

len db $-str1

rev db ?

ends

code segment

start:

mov ax,data

mov ds,ax

mov ax,0

mov bl,len

lea si,str1

add si,bx

dec si

dec si

mov cl,len

mov ch,0

lea di,rev

next:

mov bl,[si]

mov [di],bl

inc di

dec si

loop next

mov [di],24h

lea dx,rev

mov ah,09h

int 21h

mov ax, 4c00h

int 21h

ends

end start

**19 Concat String**

data segment

str1 db "Hello"

len1 db $-str1

str2 db "World"

len2 db $-str2

con db ?

ends

code segment

start:

mov ax,data

mov ds,ax

mov ax,0

mov cl,len1

lea si,str1

mov ch,0

lea di,con

next:

mov bl,[si]

mov [di],bl

inc di

inc si

loop next

mov [di],20h

inc di

lea si,str2

mov cl,len2

mov ch,0

next2:

mov bl,[si]

mov [di],bl

inc di

inc si

loop next2

mov [di],24h

lea dx,con

mov ah,09h

int 21h

mov ax, 4c00h

int 21h

ends

end start

**20 Writing two string on two different lines**

.MODEL SMALL

.STACK

.DATA

STRING1 DB "HELLO WORLD $" ; declaring string

STRING2 DB 10, 13, "GOOD MORNING $" ; declaring string

STRING3 DB 10, 13, "HAVE GOOD DAY $" ; declaring string

.CODE

MAIN PROC ; main procedure

MOV AX, @DATA ; initialize the data segment

MOV DS, AX

LEA DX, STRING1 ; loading the effective address

MOV AH, 09H ; for string display

INT 21H ; dos interrupt function

LEA DX, STRING2 ; loading the effective address

MOV AH, 09H ; for string display

INT 21H ; dos interrupt function

LEA DX, STRING3 ; loading the effective address

MOV AH, 09H ; for string display

INT 21H ; dos interrupt function

MOV AX, 4C00H ; end request

INT 21H

MAIN ENDP ; end procedure

END MAIN ; end program