Add of two 8-bit numbers

Data segment

Num1 db 84h

Num2 db 55h

Result db?

Data ends

Code segment

Start: assume ds: data, cs:code

Mov ax,data

Mov ds,ax

Mov ah,00h

Mov al,num1

Mov bl,num2

Add al,bl

Mov result,al

Mov ax,4c00h

Int 21h

Code ends

End start

Add of two 16-bit numbers

Data segment

Num1 dw 84h

Num2 dw 55h

Result db?

Data ends

Code segment

Start: assume ds: data, cs:code

Mov ax,data

Mov ds,ax

Mov ax,num1

Mov bx,num2

Add ax,bx

Mov result,ax

Mov ax,4c00h

Int 21h

Code ends

End start

Subtraction of two 8 bit number

Data segment

Num1 db 84h

Num2 db 55h

Result db?

Data ends

Code segment

Start: assume ds: data, cs:code

Mov ax,data

Mov ds,ax

Mov al,num1

Mov bl,num2

sub al,bl

Mov result,al

Mov ax,4c00h

Int 21h

Code ends

End start

Multiplication of two 16 bit number

Data segment

Num1 db 84h

Num2 db 55h

Res dw?

Data ends

Code segment

Start: assume ds: data, cs:code

Mov ax,data

Mov ds,ax

Mov ah,00h

Mov al,num1

Mov bl,num2

Mul bl

Mov res,ax

Mov ax,4c00h

Int 21h

Code ends

End start

**Block transfer using movsb and rep:**

data segment

blk1 db 06 dup(84h)

counter db 06

data ends

extra segment

blk2 db 06 dup(02h)

extra ends

code segment

start:assume ds:data, cs:code, es:extra

mov ax,data

mov ds,ax

mov ax,extra

mov es,ax

mov ax,0000h

lea si,blk1

lea di,blk2

mov cl,counter

cld

rep movsb

mov ax,4c00h

int 21h

code ends

end start

**Block transfer using movsw and without rep:**

data segment

block1 db 06 dup(84h)

counter db 03h

data ends

extra segment

block2 db 06 dup(02h)

extra ends

code segment

start: assume ds:data, cs:code, es:extra

MOV ax,data

MOV ds,ax

MOV ax,extra

MOV es,ax

MOV ax,0000h

lea si,block1

lea di,block2

MOV cl,counter

up: movsw

DEC cl

JNZ up

MOV ax,4c00h

int 21h

code ends

end start

total even numbers in a block

**data segment**

**block db 12h, 8h, 30h, 77h, 14h, 5h**

**count db 6h**

**evenno db 0h**

**data ends**

**code segment**

**start: assume ds: data, cs: code**

**mov ax, data**

**mov ds, ax**

**mov cl, count**

**mov dl, evenno**

**lea si, block**

**up: mov al, [si]**

**ror al, 1h**

**jc down**

**inc dl**

**down: inc si**

**dec cl**

**jnz up**

**mov evenno, dl**

**mov ax, 4c00h**

**int 21**

**code ends**

**end start**

total odd numbers in a block

data segment

block db 12h, 8h, 30h, 77h, 14h, 5h

counter db 06h

total\_odd db 0h

data ends

code segment

start: assume cs: code, ds: data

mov ax, data

mov ds, ax

mov cl, counter

mov dl, 00h

lea si, block

up: mov al, [si]

ror al, 01h

inc dl

down: inc si

dec cl

jnz up

mov total\_odd, dl

mov ax, 4c00h

int 21h

code ends

end start

total even numbers in a block

data segment

block db 1Ah, 0DFh, 95h, 26h, 0C3h, 77h

counter db 06h

total\_even db ?

total\_odd db ?

data ends

code segment

start: assume cs: code, ds: data

mov ax, data

mov ds, ax

mov cl, counter

mov dl, 00h

lea si, block

up: mov al, [si]

ror al, 01h ;

jc down

inc dl

down: inc si

dec cl jnz

up mov total\_even, dl

mov bl, counter;

sub bl, dl ;

mov total\_odd, bl

mov ax, 4c00h

int 21h

code ends

end start

**Program to find the largest number**

data segment

blk db 21h,02h,84h,07h,20h

counter db 04h

max db ?

data ends

code segment

start: assume cs:code,ds:data

mov ax,data

mov ds,ax

lea si,blk

mov dl,[si]

up: inc si ; if carry flag = 1 then it jumps else it doesnt

cmp dl,[si]

jnc down

mov dl,[si]

down: dec counter

jnz up

mov max,dl

mov ax,4c00h

int 21h

code ends

end start

Program to find smallest number

data segment

nums db 21h,02h,84h,07h,20h

counter db 05h

min\_num db ?

data ends

code segment

start: assume ds:data,cs:code

MOV ax,data

MOV ds,ax

MOV si,0000h

MOV cl,counter

MOV al,00h

comp: CMP al,nums[si]

jnc store

inc si

dec cl

store:MOV al,nums[si]

loop comp

MOV min\_num,al

MOV ax,4c00h

int 21h

code ends

end start

Searching data ‘0Fh’ from a given array.

data segment

nums db 21h,02h,84h,22h,68h

counter db 05h

occurs db 00h

data ends

code segment

start: assume ds:data, cs: code

MOV ax,data

MOV ds,ax

MOV si,0000h

MOV cl,counter

MOV al,0fh

MOV bl,00h

comp: CMP al,nums[si]

jnz found

inc occurs

found:

inc si

dec cl

jnz comp

MOV ax,4c00h

int 21h

code ends

end start

Program of ASCII to bcd:

data segment

no1 db '8'

no2 db '4'

result db 00h

data ends

code segment

start: assume ds:data, cs:code

MOV ax,data

MOV ds,ax

MOV al,no1

MOV bl,no2

ADD al,bl

AAA

MOV result,al

MOV ax,4c00h

int 21h

code ends

end start

Program of Addition of 2 ASCII Numbers using AAA:

data segment

no1 db '8'

no2 db '4'

result db 00h

data ends

code segment

start: assume ds:data, cs:code

MOV ax,data

MOV ds,ax

MOV al,no1

MOV bl,no2

ADD al,bl

AAA

MOV result,al

MOV ax,4c00h

int 21h

code ends

end start

Program for arranging numbers in Ascending Order:

data segment

block1 db 21h,02h,84h,22h

counter db 04h

data ends

code segment

start:assume ds:data,cs:code

mov ax,data

mov ds,ax

mov ch,04h

loop2:mov cl,04h

lea si,block1

mov cl,counter

loop1:mov al,[si]

cmp al,[si+1]

jc down

XCHG al,[si+1]

XCHG al,[SI]

down:inc si

dec cl

jnz loop1

dec ch

jnz loop2

mov ax,4c00h

int 21h

code ends

end start

Program for arranging numbers in Descending Order :

data segment

block1 db 21h,02h,84h,22h

counter db 04h

data ends

code segment

start:assume ds:data,cs:code

mov ax,data

mov ds,ax

mov ch,04h

loop2:mov cl,counter

lea si,block1

loop1:mov al,[si]

cmp al,[si+1]

jnc down

XCHG al,[si+1]

XCHG al,[si]

down:inc si

dec cl

jnz loop1

dec ch

jnz loop2

mov ax,4c00h

int 21h

code ends

end start

**adding 2 hex numbers getting decimal result using DAA**

data segment

var1 db 48h

var2 db 24h

result db ?

data ends

code segment

start: assume ds:data, cs:code

MOV ax,data

MOV ds,ax MOV al,var1 MOV bl,var2 ADD al,bl DAA

MOV result,al MOV ax,4c00h int 21h

Code ends

end start