16-Bit Addition

Program :

```
Data segment
    msg db 0dh,0ah,"Enter first number: $"
    msg1 db 0dh,0ah,"Enter second number: $"
    result db 0dh,0ah,"The Result of addition is: $"
Data ends

Code segment
    assume CS:Code,DS:Data
start:
    mov ax, Data
    mov DS, ax
    mov dx,offset msg ;----- Display contents of variable msg
    mov ah,09h
    int 21h
    mov ah,01h ; To accept 4bits 10s input
    int 21h
    call AsciiToHex
    mov bl, al
    rol bl,4
    mov ah,01h ; To accept 4bits units input
    int 21h
    call AsciiToHex
    add bl,al
    mov ch,bl
    mov dx,offset msg1 ;----- Display contents of variable msg1
    mov ah,09h
    int 21h
    mov ah,01h ; To accept 4bits 10s input
    int 21h
    call AsciiToHex
    mov bl, al
```

```
    rol bl,4
    mov ah,01h ; To accept 4bits units input
    int 21h
    call AsciiToHex
    add bl,al
    add bl,ch;--------------------main addn
    mov dx,offset result ; Display contents of string result
    mov ah,09h
    rol bl,4h; interchange nibbles
    mov  al,bl
    call HexToAscii
    mov dl, al; display 10s place
    mov ah,02h
    int 21h
    mov ah,4ch ; Terminate the program
    int 21h
        AsciiToHex proc
        cmp al,41h ; If it is greater than or equal to 41 then we also need to sub 8h
along with 30
        jc skip
        sub al, 07h
        skip: sub al, 30h
        ret
        endp
        HexToAscii proc
        cmp al,0ah ; If it is greater than or equal to 0a then we also need to add 07
along with 30
        jc skip1
        add al,07h
        skip1: add al,30h
        ret
        endp
Code ends
end start
```

16-Bit Subtraction

Program :

```
Data segment
    msg db 0dh,0ah,"Enter first number: $"
    msg1 db 0dh,0ah,"Enter second number: $"
    result db 0dh,0ah,"The Result of addition is: $"
Data ends

Code segment
    assume CS:Code,DS:Data
start:
    mov ax, Data
    mov DS, ax
    mov dx,offset msg ;----- Display contents of variable msg
    mov ah,09h
    int 21h
    mov ah,01h ; To accept 4bits 10s input
    int 21h
    call AsciiToHex
    mov bl, al
    rol bl,4
    mov ah,01h ; To accept 4bits units input
    int 21h
    call AsciiToHex
    add bl,al
    mov ch,bl
    mov dx,offset msg1 ;----- Display contents of variable msg1
    mov ah,09h
    int 21h
    mov ah,01h ; To accept 4bits 10s input
    int 21h
    call AsciiToHex
    mov bl, al
```

16-Bit Multiplication

Program :

```
Data segment
    msg db 0dh,0ah,"Enter first number: $"
    msg1 db 0dh,0ah,"Enter second number: $"
    result db 0dh,0ah,"The Result of Multipication is: $"
Data ends

Code segment
    assume CS:Code,DS:Data
start:
    mov ax, Data
    mov DS, ax
    mov dx,offset msg ;----- Display contents of variable msg
    mov ah,09h
    int 21h
    mov ah,01h ; To accept 4bits 10s input
    int 21h
    call AsciiToHex
    mov bl, al
    rol bl,4
    mov ah,01h ; To accept 4bits units input
    int 21h
    call AsciiToHex
    add bl,al
    mov ch,bl
    mov dx,offset msg1 ;----- Display contents of variable msg1
    mov ah,09h
    int 21h
    mov ah,01h ; To accept 4bits 10s input
    int 21h
    call AsciiToHex
    mov bl, al
```

```
    rol bl,4
    mov ah,01h ; To accept 4bits units input
    int 21h
    call AsciiToHex
    add al,bl
    mul ch;--------------main multip--result in ax
    mov bx,ax
    mov cx,ax
    mov dx,offset result ; Display contents of string result
    mov ah,09h
    int 21h
    and bh,0f0h
    rol bh,4
    mov  al,bh
    call HexToAsciiDisp
    and ch,0fh
    mov al,ch
    call HexToAsciiDisp
    and bl,0f0h
    rol bl,4
    mov  al,bl
    call HexToAsciiDisp
    and cl,0fh
    mov al,cl
    call HexToAsciiDisp
    mov ah,4ch ; Terminate the program
    int 21h


        AsciiToHex proc
        cmp al,41h ; If it is greater than or equal to 41 then we also need to sub 8h
along with 30
        jc skip
        sub al, 07h
        skip: sub al, 30h
```

```
        ret
        endp
        HexToAsciiDisp proc
        cmp al,0ah ; If it is greater than or equal to 0a then we also need to add 07
along with 30
        jc skip1
        add al,07h
        skip1: add al,30h
        mov dl, al
        mov ah,02h
        int 21h
        ret
        endp
Code ends
end start
```

Output :

```
Assembling file:   D:\test.asm  to  test.OBJ
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  467k


D:\>TLINK D:\test
Turbo Link  Version 7.1.30.1. Copyright (c) 1987, 1996 Borland International
Warning: No stack

D:\>D:\test

Enter first number: 22
Enter second number: 04
The Result of Multipication is: 0088
D:\>test

Enter first number: 3C
Enter second number: A0
The Result of Multipication is: 2580
```

16-Bit Division

Program :

```
Data segment
    msg db 0dh,0ah,"Enter the denominator: $"
    msg1 db 0dh,0ah,"Enter the denominator: $"
    result db 0dh,0ah,"The Quotient is: $"
    result1 db 0dh,0ah,"The Result of Remainder is: $"
Data ends

Code segment
    assume CS:Code,DS:Data
start:
    mov ax, Data
    mov DS, ax
    mov dx,offset msg ;----- Display contents of variable msg
    mov ah,09h
    int 21h
    mov ah,01h ; To accept 4bits 10s input
    int 21h
    call AsciiToHex
    mov bl, al
    rol bl,4
    mov ah,01h ; To accept 4bits units input
    int 21h
    call AsciiToHex
    add bl,al
    mov cl,bl
    mov dx,offset msg1 ;----- Display contents of variable msg1
    mov ah,09h
    int 21h
    mov ah,01h ; To accept 4bits 10s input
    int 21h
    call AsciiToHex
```

```
mov bl, al
rol bl,4
mov ah,01h ; To accept 4bits units input
int 21h
call AsciiToHex
add al,bl
mov ah,00h
div cl;--------------main divn --al(quotient), ah(remainder)
mov bx,ax
mov dx,offset result ; Display contents of string result
mov ah,09h
int 21h
mov cl,bl
and bl,0f0h
rol bl,4h; interchange nibbles
mov  al,bl
call HexToAsciiDisp
mov al,cl
and al,0fh
call HexToAsciiDisp
mov dx,offset result1 ; Display contents of string result
mov ah,09h
mov cl,bl
and bl,0f0h
rol bl,4h; interchange nibbles
mov  al,bl
call HexToAsciiDisp
mov al,cl
and al,0fh
call HexToAsciiDisp
mov ah,4ch ; Terminate the program
int 21h
    AsciiToHex proc
```

```
        cmp al,41h ; If it is greater than or equal to 41 then we also need to sub 8h
along with 30
        jc skip
        sub al, 07h
        skip: sub al, 30h
        ret
        endp
        HexToAsciiDisp proc
        cmp al,0ah ; If it is greater than or equal to 0a then we also need to add 07
along with 30
        jc skip1
        add al,07h
        skip1: add al,30h
        mov dl, al
        mov ah,02h
        int 21h
        ret
        endp
Code ends
end start
```

Output :

```
D:\>TLINK D:\test
Turbo Link  Version 7.1.30.1. Copyright (c) 1987, 1996 Borland International
Warning: No stack

D:\>D:\test

Enter the denominator: 01
Enter the denominator: 99
The Quotient is: 99
The Result of Remainder is: 00
D:\>test

Enter the denominator: A0
Enter the denominator: 41
The Quotient is: 00
The Result of Remainder is: 41
```