ADD

```
.model small
.code
mov al,02h
mov bl,05h
add al,bl
int 21h
end
```

BINARY SEARCH

ALP to write the program for binary search

```
.model small
.data
array dw 1234h,1342h,1423h,2161h, 5678h
len dw 5
;($-array)/2
key dw 1342h
msg1 db "element found",10,13,"$"
msg2 db "element not found", 10,13,"$"
.code
 mov ax,@data
 mov ds,ax
 mov bx,01
 mov dx,len
 mov cx,key
 again: cmp bx,dx
        ja fail
        mov ax,bx
         add ax,dx
         shr ax,01
           mov si,ax
         dec si
         add si,si
        cmp cx, [si]
       jae bigger
        dec ax
 mov dx,ax
 jmp again
bigger:JE success
       inc ax
       mov bx,ax
       jmp again
success: lea dx,msg1
         jmp display
fail:    lea dx,msg2
display:  mov ah,09h
          int 21h
          mov ah,4ch
          int 21h
           end
```

BINARY TO HEXADECIMAL

```
.MODEL SMALL
.CODE
MOV AL,25H
MOV BL,AL
AND AL,0FH
MOV BH,AL
AND BL,0F0H
MOV CL,04H
ROR BL,CL
MOV AL,0AH
MUL BL
ADD AL,BH
INT 21H
END
```

BUBBLE SORT2

```
.model small
.data
string1 db 56h,99h,12h
.code
mov ax,@data
mov ds,ax
mov ch,02h
up2:mov cl,02h
lea si,string1
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 ah,4ch
int 21h
end
```

BUBBLE SORT

```
.MODEL SMALL     N 8-BIT NUMBERS
.STACK 64

.DATA      ;stack segment

;data segment

ARR DB 10H, 08H, 04H, 09H, 06H    ;byte definition of array
```

```
LEN DB ($-ARR)    ;length of the array
.CODE

MOV AX,@DATA      ; initialise data segment MOV DS, AX
MOV BL, LEN ;move the length into BL
DEC BL      ;decrement BL by 01
OUTLOOP: MOV CL,  BL   ;move BL value into CL MOV SI,  00H
;initialize SI to 00H
INLOOP: MOV AL, ARR[SI]      ;move A[SI] into AL register
CMP AL, ARR[SI+1];compare values in AL and ARR[SI] JB NEXT     ;if
AL<A[SI+1],jump to NEXT
XCHG AL,ARR[SI+1];swap values of AL and A[SI] MOV ARR[SI],AL
NEXT: INC SI      ;increment SI value by 01 LOOP INLOOP
DEC BL      ;decrement BL value by 01
JNZ OUTLOOP ;jump to outloop till ZF=1 MOV AH,4CH   ;end program
INT 21H
END
```

COUNTER

ALP to display counter

```
.model small
.stack
.code
mov al,030h
loop1:mov dl,al
      mov ah,02h
      int 21h
      push ax
      mov bl,030h
Loop2:mov dl,bl
      mov ah,02h
      int 21h
      inc bl
      call delay
      mov ah,03h
      int 10h
      mov dl,01h
      mov ah,02h
      int 10h
      cmp bl,039h
      JLE loop2
      mov dl,00h
      mov ah,02h
      int 10h
      pop ax
      inc al
      cmp al,039h
      JLE loop1
      mov ah,4ch
      int 21h
```

```
      Delay proc
      push cx
      push bx
      mov cx,0fffh
loop3: mov bx,0ffh
loop4: dec bx
       JNZ loop4
       loop loop3
       pop bx
       pop cx
       ret
       delay endp
       end
```

DIVISION

```
.model small
.data
a dw 0000h
b dw 0000h
.code
mov ax, 0008h
mov cx, 0002h
div cx
mov bx,@data
mov ds,ax
mov a,ax
mov b,dx
int 21h
end
```

LOGICAL 0S AND 1S

```
.model small
.data
num dw 2345h
z dw 0000h
o dw 0000h
.code
mov ax,@data
mov ds,ax
mov ax,num
mov bx,00h
mov cx,10h
;mov bl,10h
mov dx,00h
up: rol ax,01
    JC one
    inc bx
```

```
    jmp next
one: inc dx
next: dec cx
      jnz up
mov z,bx
mov o,dx
int 21h
end
```

MULTIPLY

```
.model small
.code
mov ax,02
mov bx,03
mul bx
mov cx,ax
mov ax,dx
mov cx,ax
mov ah,4ch
int 21h
end
```

ODD EVEN

```
.model small
.data
num1 dw 0026h
num2 db 02h
msg1 db 'number is even$'
msg2 db 'number is odd$'
.code
mov ax,@data
mov ds,ax
mov ax,num1
mov bl,num2
div bl
cmp ah,00h
jz even
lea dx,msg2
jmp exit
even:lea dx, msg1
exit: mov ah,09h
int 21h
mov ah,4ch
int 21h
end
```

PALINDROME

```
.model small
.data
str db "malayalam$"
len equ 09
```

```
msg1 db 10,13, "palindrome$"
msg2 db 10,13, "not a palindrome$"
rstr db 40 dup(0)
.code
mov ax,@data
mov ds,ax
mov es,ax
lea si,str
lea di,rstr
add di,len
dec di
mov cl,len
again: mov al,[si]
mov [di],al
dec di
inc si
dec cl
jnz again
cld
mov ch,0h
mov cl,len
lea si, str
lea di,rstr
repe cmpsb
je palin
lea dx,msg2
mov ah,09h
int 21h
exit: mov ah,4ch
int 21h
palin:lea dx,msg1
mov ah,09h
int 21h
jmp exit
end
```

POSITIVE OR NEGATIVE

```
.model small
.data
num db -12h
res db ?
msg1 db 'number is positive$'
msg2 db 'number is negative$'
.code
mov ax,@data
mov ds,ax
mov al,num
rol al,01
jc dmsg2
lea dx,msg1
jmp exit
```

```
dmsg2: lea dx, msg2
exit:
mov ah,09h
int 21h
mov ah,4ch
int 21h
end
```

STRING COMPARISON

```
disp macro msg
lea dx,msg
mov ah,09h
int 21h
endm
.model small
.data
m1 db 13,10, "enter S1:$"
m2 db 13,10, "enter S2:$"
m3 db 13,10, "len1:$"
m4 db 13,10, "len 2:$"
m5 db 13,10, "str1=str2 $"
m6 db 13,10, "str1!=str2:$"
str1 db 80,?,80 dup(?)
str2 db 80,?,80 dup(?)
L1 db ?
L2 db ?
.code
mov ax,@data
mov ds,ax
mov es,ax

disp m1
lea dx, str1
call read

disp m2
lea dx, str2
call read

mov al,[str1+1]
mov l1,al

mov al,[str2+1]
mov l2,al
cmp al,l1
jne strnotequal
mov ch,0
mov cl,l1
lea si, str1+2
lea di, str2+2
cld
```

```
repe cmpsb
jne strnotequal

disp m5
jmp next
strnotequal: disp m6
next: disp m3
mov al,l1
call display
disp m4
mov al,l2
call display
mov ah,4ch
int 21h

read proc
mov ah,0AH
int 21h
ret
read endp

display proc
aam
mov bx,ax
add bx,3030h
mov ah,02
mov dl,bh
int 21h
mov dl,bl
int 21h
ret
display endp
end
```

SYSTEM TIME

```
model small
.code
mov ah,2ch
int 21h
mov al,ch
AAM
mov bx,ax
call disp
mov dl, ":"
mov ah,02h
int 21h
mov al,cl
AAM
mov bx,ax
```

```
call disp
mov dl, ":"
mov ah,02h
int 21h
mov al,dh
AAM
mov bx,ax
call disp
mov ah,4ch
int 21h
DISP: mov dl,bh
      add dl,30h
      mov ah,02h
      int 21h
      mov dl,bl
      add dl, 30h
      mov ah,02h
      int 21h
      ret
      end
```