1.Hello World.

```
org 100h
mov dx, offset msg
mov ah,9h
int 21h
ret
msg DB "Hello World $"
```

2.Add numbers by taking input.

```
org 100h
mov ah, 1h
int 21h
mov bl,al
mov ah, 1h
int 21h
sub al,30
add al,bl
hlt
```

3.Print number in an array.

#WAP to print the numbers in array

```
org 100h
.data
a db 7,9,1,2,3
```

```
.code
mov ax,@data
mov ds,ax
mov cx,5
mov si,0

L1:
mov bh,a[si]
mov dl,bh
# next step add 48 is to get ascii value
add dl,48
mov ah,2h
int 21h
mov dl,0h
mov ah,2h
int 21h
inc si
loop L1

hlt
```

4.Print A-Z.

```
org 100h
mov cx,26
mov bh,0h
```

```
add bh,65
L1:
mov dl,bh
mov ah,2h
int 21h
mov dl,0h
mov ah,2h
int 21h
inc bh
loop L1
hlt
```

5.Displaying the addition of two numbers by taking input.

```
org 100h
mov ah, 1h
int 21h
mov bl,al
mov ah, 1h
int 21h
add al,bl
sub al,30h
mov dl,al
mov ah,2h
int 21h
hlt
```

6.Display elements in array and addition of elements in an array.

```
org 100h
.data
a db 2,2,3
.code
mov ax,@data
mov ds,ax
mov cx,3
mov si,0
mov bl,0
L1:
mov bh,a[si]
mov dl,bh
add bl,dl
# next step add 48 is to get ascii value
add dl,48
mov ah,2h
int 21h
mov dl,0h
mov ah,2h
int 21h
inc si
loop L1
mov dl,bl
add dl,48
mov ah,2h
```

```
int 21h
hlt
```

7.Even or Odd numbers.

```
org 100h
.data
e db "It is an Even number $"
o db "It is a Odd number $"
.code
mov dl,9h
mov bl,2h
div bl
cmp ah,00h
add dl,30h
mov ah,2h
int 21h
JE even:
mov dx,offset o
mov ah,9h
int 21h
mov ah,4ch
int 21h
even:
mov dx, offset e
mov ah,9h
int 21h
```

```
mov ah,4ch
int 21h
hlt
```

8.Even or Odd by taking input.

```
org 100h
.data
e db "It is an Even number $"
o db "It is a Odd number $"
.code
mov ah,1h
int 21h
mov dl,al
mov bl,2h
div bl
cmp ah,00h
mov ah,2h
int 21h
mov dl,0h
int 21h
JE even:
mov dx,offset o
mov ah,9h
int 21h
mov ah,4ch
int 21h
```

```
even:
mov dx, offset e
mov ah,9h
int 21h
mov ah,4ch
int 21h
hlt
```

9.Average of array elements.

```
org 100h
.data
a db 1,2,3,4,5
avg db ?
.code
mov ax, @data
mov ds, ax

mov cx, 5
mov si, 0

mov bx, cx
mov ax, 0

L1:
add al, a[si]
```

```
adc ah, 0
inc si
loop L1

div bl
mov avg, al
mov dl, avg
add dl, 48
mov ah, 2h
int 21h

hlt
```

10.S1-S2 in array.

```
org 100h
.data
a db 2,2,3,5
.code
mov ax,@data
mov ds,ax
mov cx,2
mov si,0
mov bl,0
mov dh,0h
mov al,0h
```

```
L1:
mov bh,a[si]
mov dl,bh
add bl,dl
add dh,dl
inc si
loop L1
mov ax,@data
mov ds,ax
mov cx,2
mov si,2
mov bl,0
L2:
mov bh,a[si]
mov dl,bh
add bl,dl
add al,dl
inc si
loop L2
sub al,dh
mov dl,al
add dl,48
mov ah,2h
int 21h
hlt
```

11.Factorial of a number.

```
org 100h
mov al,03h
mov dl,al
dec dl
mov cl,dl
L1:
mul dl
dec dl
loop L1
mov dl,al
add dl,30h
mov ah,2h
int 21h
hlt
```

12.Factorial of a number by taking input.

```
org 100h
.data
a db "Factorial of $ "
b db " is $"
.code
mov ax,@data
```

```
mov dx,offset a
mov ah,9h
int 21h
mov ah,1h
int 21h
sub al,30h
mov dl,al
dec dl
mov cl,dl
L1:
mul dl
dec dl
loop L1
mov bl,al
mov dx,offset b
mov ah,9h
int 21h
mov dl,bl
add dl,30h
mov ah,2h
int 21h
hlt
```

13. nPr.

```
org 100h
.data
```

```
n db 4
r db 2
.code
mov al,n
sub al,r
mov dl,al
dec dl
mov cl,dl
l1:
mul dl
dec dl
loop l1
mov bl,al
mov al,n
mov dl,al
dec dl
mov cl,dl
l2:
mul dl
dec dl
loop l2
div bl
mov dl,al
add dl,30h
mov ah,2h
int 21h
```

```
hlt
```

14. nCr

```
org 100h
.data
n db 4
r db 2
.code
;n-r (nCr lo n-r)
mov al,n
sub al,r
mov dl,al
dec dl
mov cl,dl
l1:
mul dl
dec dl
loop l1
mov bl,al
;r!(nCr lo r)
mov al,r
mov dl,al
dec dl
mov cl,dl
l3:
```

```
mul dl
dec dl
loop l3
mul bl
mov bl,al

;n(nCr lo n)
mov al,n
mov dl,al
dec dl
mov cl,dl
l2:
mul dl
dec dl
loop l2
div bl
mov dl,al
add dl,30h
mov ah,2h
int 21h

hlt
```

15. Number of negative numbers in an array.

```
org 100h
```

```
.data
a db -1,-2,3,4,5
.code
mov dx,@data
mov si,0h
mov ch,5h
mov dh,0h
l1:
mov al,a[si]
cmp al,0h
jl v
inc si
dec ch
cmp ch,0h
je l2
loop l1

v:
inc dl
inc si
dec ch
cmp ch,0h
jne l1
je l2
l2:
add dl,48
```

```
mov ah,2h
int 21h

ret
```

16. Number of positive numbers in an array.

```
org 100h

.data
a db -1,-2,3,4,5
.code
mov dx,@data
mov si,0h
mov ch,5h
mov dh,0h
l1:
mov al,a[si]
cmp al,0h
jg v
inc si
dec ch
cmp ch,0h
je l2
loop l1

v:
```

```
inc dl
inc si
dec ch
cmp ch,0h
jne l1
je l2
l2:
add dl,48
mov ah,2h
int 21h

ret
```

17.Reverse of an array.

```
org 100h
.data
a db 1,2,3,4,5,6
.code
mov ax,@data
mov si,5h
mov cx,6h
l1:
mov dl,a[si]
add dl,48
mov ah,2h
int 21h
```

```
mov dl,0h
mov ah,2h
int 21h
dec si
loop l1
ret
```

18.Find element in an array.

```
org 100h
.data
a db 1,2,3,4,5,6
b db "element found at index $"
c db "element not found$"
.code
mov ax,@data
mov si,5h
mov cl,6h
mov dh,5h
l1:
mov al,a[si]
cmp dh,al
je l2
dec si
loop l1
mov dx,offset c
mov ah,9h
```

```
int 21h
jmp j
l2:
mov dx,offset b
mov ah,9h
int 21h
dec cx
mov dl,cl
add dl,48
mov ah,2h
int 21h
j:
```

19.Count number of even numbers in an array.

```
org 100h
.data
a db 1,2,3,4,5,6,7,9
count db 0
.code
mov ax,@data
mov ds,ax
mov cx,9
mov si,0
L1:
mov al,a[si]
and ax,0FFh
```

```
mov bl,2h
div bl
cmp ah,0h
JNE L2
inc count
L2:
inc si
loop L1
mov dl,count

add dl,30h
mov ah,2h
int 21h
mov ah,4ch
int 21h
hlt
```

20.Minimum element in array.

```
org 100h
.data
a db 5,7,3,4,1
b db "The minimum element is $"
.code
mov ax,@data
mov ds,ax
mov si,0h
```

```
mov cl,4
mov bh,a[si]
inc si
l1:
mov bl,a[si]
cmp bh,bl
jl l2
mov bh,a[si]
l2:
inc si
loop l1
mov dx,offset b
mov ah,9h
int 21h
mov dl,bh
add dl,48
mov ah,2h
int 21h

hlt
```

21.Bubble Sort.

```
org 100h

.data
a db 5,7,9,3,4,5
```

```
temp dw ?
b db "The sorted array is: $"
.code
mov ax, @data
mov ds, ax
mov cx, 5
l1:
mov temp,cx
mov si,0h
l2:
mov al, a[si+1]
cmp a[si],al
jle l3
mov bl,a[si+1]
mov bh,a[si]
mov a[si],bl
mov a[si+1],bh
l3:
inc si
inc si+1
loop l2
l4:
mov cx,temp
loop l1
mov si,0h
mov cx,6
```

```
mov dx,offset b
mov ah,9h
int 21h
l5:
mov dl,0h
mov ah,2h
int 21h
mov dl,a[si]
add dl,30h
mov ah,2h
int 21h
inc si
loop l5
hlt
```

22.LCM of two numbers.

```
org 100h
.data
rd db ?
a db "The lcm is: $"
.code
mov ax,@data
mov bl,5h
mov bh,7h
cmp bl,bh
```

```
jl l2
je l1
jmp l3
l1:
mov dx,offset a
mov ah,9h
int 21h
mov dl,bl
add dl,48
mov ah,2h
int 21h
l2:
xchg bl,bh
jmp l3
l3:
mov cl,bl
mov rd,1h
l4:
mov ax,0h
mov al,bh
mul rd
mov dl,al
div bl
inc rd
cmp ah,0h
je l5
```

```
loop l4
l5:

mov ah,2h
int 21h

ret
```

23.HCF of two numbers.

```
org 100h
.data
a db "THE HCF IS:$"
.code
mov ax,@data
mov dx,offset a
mov ah,9h
int 21h
mov dx,0h
mov ax,0h
mov bl,3h
mov bh,6h
cmp bl,bh
je l1
cmp bl,bh
jl l2
jmp l3
l2:
```

```
xchg bl,bh
l3:
mov cl,bl
l4:
mov al,bl
div bh
je l5
mov al,bh
mov dh,bh
mov bh,ah
cmp ah,0h
je l5
mov dh,ah
mov ah,0h
div dh
cmp ah,0h
je l5
loop l4
l1:
mov dl,bh
add dl,48
mov ah,2h
int 21h
jmp l6
l5:
mov dl,dh
```

```
add dl,48
mov ah,2h
int 21h
l6:

ret
```