Microprocessors (8086)-Programs

1. **Write ALP for 8-bit addition and store result in al register.**

```
code segment
assume cs:code
mov al,04h
mov bl,02h
add al,bl
hlt
code ends
end
```

2. **Write ALP for 8-bit subtraction and take input from 2000h memory location store result in 5000h memory location**

```
code segment
assume cs:code
mov si,2000h
mov al,[si]
mov bl,[si+1]
sub al,bl
mov di,5000h
mov [di],al
hlt
code ends
end
```

3. **Write ALP for 8-bit multiplication and take input from 2000h memory location store result in 5000h**

```
code segment
assume cs:code
mov si,2000h
mov di,5000h
mov al,[si]
mov bl,[si+1]
mul bl
mov [di],ax
hlt
code ends
end
```

4. **Write ALP for 8-bit division and store quotient in "pp" directive and remainder "aa" directive**

```
data segment
pp db 0
aa db 0
data ends
code segment
   assume cs:code, ds:data
   mov ax,0ah
   mov bl,02h
   div bl
   mov pp,al
   mov aa,ah
   hlt
   code ends
   end
```

5. **Write ALP for 16-bit addition and store sum in "cse" directive and carry in "ece" directive**

```
assume ds:data, cs:code
data segment
cse dw 0
ece db 0
data ends
code segment
mov ax,0876h
mov bx,0562h
mov cl,00h
add ax,bx
mov cse,ax
jnc up
inc cl
up:mov ece,cl
hlt
code ends
end
```

6. **Write ALP for 16-bit subtraction and store difference in 5000h and barrow in "mec" directive.**

```
assume ds:data,cs:code
data segment
mec db 0
data ends
code segment
mov ax,0254h
mov bx,0637h
sub ax,bx
mov cl,00h
jnc up
inc cl
neg ax
up:mov di,5000h
mov [di],ax
mov mec,cl
hlt
code ends
end
```

7. Write ALP for 16-bit multiplication and take input from 2000h memory location store result in 6000h

```
assume cs:code
code segment
mov si,2000h
mov di,6000h
sub ax,[si]
mov bx,[si+2]
mul bx
mov [di],ax
mov [di+2],dx
hlt
code ends
end
```

8. Write ALP for 16-bit division and take input from 3000h, store quotient and remainder also in that 3000h location only.

```
assume cs:code
code segment
mov si,3000h
mov ax,[si]
mov bx,[si+2]
div bx
mov [si+4],ax
mov [si+6],dx
hlt
code ends
end
```

9. Write ALP for two 8-bit BCD numbers

```
assume cs:code
code segment
mov al,08h
mov bl,02h
add al,bl
daa
hlt
code ends
end
```

10. Write ALP for finding a factorial of a number.

```
assume cs:code
code segment
mov ax,0001h
mov cx,0005h
up:mul cx
dec cx
jnz up
hlt
code ends
end
```

11. Write ALP for searching a 8-bit number,if number is found store al with 00h otherwise with FFh

```
code segment
assume cs:code
mov cl,07h
mov si,1000h
mov bl,77h
go:mov al,[si]
cmp al,bl
jz up
inc si
dec cl
jnz go
mov al,0ffh
jmp down
up:mov al,00h
down:hlt
code ends
end
```

12. Write ALP for ascending ordering of five 8-bit numbers

```
code segment
assume cs:code
mov cl,05h
dec cl
go:mov ch,05h
dec ch
mov si,4000h
down:mov al,[si]
inc si
cmp al,[si]
jc up
xchg al,[si]
xchg al,[si-1]
up:dec ch
jnz down
dec cl
jnz go
hlt
code ends
end
```

13. Write ALP for descending ordering of five 8-bit numbers

```
code segment
assume cs:code
mov cl,05h
dec cl
go:mov ch,05h // no of iterations
dec ch
mov si,4000h
down:mov al,[si]
inc si
cmp al,[si]
jnc up
xchg al,[si]
xchg al,[si-1]
up:dec ch
jnz down
dec cl
jnz go
hlt
code ends
end
```

14. Write ALP for transfering of 8-bit data(ten bytes) from one location to another location.

```
code segment
assume cs:code
mov si,4000h
mov di,5000h
mov cl,0ah
up:mov al,[si]
mov [di],al
inc si
inc di
dec cl
jnz up
hlt
code ends
end
```

15. Write ALP for creation of multiplication table for 8-bit number

```
code segment
assume cs:code
mov si,2000h
mov bl,00h
mov bh,05h
mov cl,0ah
go:inc bl
mul bl
mov [si],ax
inc si
inc si
mov al,bh
dec cl
jnz go
hlt
code ends
end
```

16. Write ALP for finding no of one's and zero's of a 16 bit-number

```
code segment
assume cs:code
mov ax,2b2ah
mov bx,0000h
mov cl,10h
down:rcr ax,0001h
jc up
inc bh
jmp go
up:inc bl
go:dec cl
jnz down
hlt
code ends
end
```

17. Write ALP for moving of a 12 string bytes of data

```
assume cs:code,ds:data
data segment
num1 db "computer2a2b"
data ends
code segment
mov ax,data
mov ds,ax
mov si,offset num1
mov cl,0ch
cld
repz movsb
hlt
code ends
end
```

18. Write ALP for searching of a sting byte.

```
assume ds:data,cs:code
data segment
rock db "mplab@12"
data ends
code segment
mov ax,data
```

```
mov es,ax
mov cl,08h
mov di,offset rock
mov al,'@'
cld
repne scasb
hlt
code ends
end
```

19. Write ALP for comparision of a two strings.

```
assume cs:code,ds:data,es:extra
data segment
rise db "dcme2a2b"
data ends
extra segment
rock db " dcme2a2c "
extra ends
code segment
mov ax,data
mov ds,ax
mov ax,extra
mov es,ax
mov cl,08h
mov si,offset rise
mov di,offset rock
cld
repe cmpsb
hlt
code ends
end
```

20. Write a program to compare two 16 bit quantities stored in AX&BX registers. If both contents are equal, then increment SI register.

```
Code segment
Assume cs:code
mov ax,1122h
mov bx,3344h
mov si,1000h
cmp ax, bx
jz go
nop
go: inc si
int 03h
code ends
end
```

21. Write a program to add two 16 bit numbers stored in AX&BX register. If no carry exists after addition increment SI register

```
Code segment
Assume cs:code
mov ax,1122h
mov bx,3344h
mov si,1000h
add ax, bx
jnc go
nop
go: inc si
int 03h
code ends
end
```

22. Write a program to compare two 8 bit quantities, if both contents are not equal, transfer 04h into CL register otherwise increment SI register.

```
Code segment
Assume cs:code
mov ah,22h
mov bh,44h
mov si,1000h
cmp ah, bh
jnz go
inc si
nop
go: mov cl, 04
int 03h
code ends
end
```

23. Write a program to add two 8 bit quantities, compare sum with 10h if the result is less than 10h then increment si and load cl with 04h otherwise then compare sum with 40h ,if sum is less than 40h then load cl with 04h otherwise cl with 00h.

```
Code segment
Assume cs:code
mov ah,22h
mov bh,44h
mov si,1000h
add ah, bh
cmp ah, 10h
jc L1
cmp ah, 40h
jc L2
mov cl, 00h
nop
L1: inc si
L2: mov cl, 04h
int 03h
code ends
end
```

24. Write a program using loop instruction

```
Code segment
Assume cs:code
mov al, data
mov cl, al
dec cl
L1: mul cl
dec cl   ; multiplication until cl=0
loopnz L1
int 03h
code ends
end
```

25. Write Program for subtraction of two numbers using parameter passing through *registers.*

```
assume  cs: code, ds: data
data segment
a db 05h
b  db 02h
c db ?
data ends
code segment
mov ax,data
mov ds,ax
mov  al, a
mov  bl, b
call  subtraction
mov  c,al
int  03h
subtraction  proc
sub  al, bl
ret
subtraction  endp
code  ends
end
```

26. Program for subtraction of two numbers using parameters passing through *memory locations*

```
assume  cs: code, ds: data
data segment
      a  db 33h
      b  db 55h
      c  db ?
data ends
code segment
        mov ax,data
        mov ds,ax
        mov  al, a
        mov  [8000h], b
        call  subtraction
        mov c,al
        int 03h
        subtraction  proc
        sub  ax, [8000]
        ret
        subtraction  endp
        code ends
        end
```

27. Write a Program for square of a number using parameter passing to procedure

```
Data segment
Sp db 56h
C dw ?
Data ends
Code segment
Assume cs:code,ds:data
Mov ax,data
Mov ds,ax
Mov al,sp
Call square
Mov c,ax
Int 03h
Square proc
Mul al
Ret
Square endp
Code ends
End
```

27. Write a Program for finding length of the string

```
Assume cs:code,ds:data
 Data segment
Num db “india$”
Result db ?
Data ends
Code segment
Mov ax,data
Mov ds,ax
Mov si,offset num
Mov cl,00h
Up:mov ax,[si]
Cmp ax, ‘$’
JZ xyz
Inc si
Inc cl
Jmp up
Xyz: mov result,cl
 Nop
 Hlt
Code ends
end
```

28. Write a Program for reverse of a string

```
Assume cs:code,ds:data
 Data segment
Num db “india$”
Data ends
Code segment
Mov ax,data
Mov ds,ax
Mov si,offset num
Mov di,4FFFh
Mov cl,06h
Up:mov al,[si]
mov [di],al
inc si
dec di
dec cl
jnz up
hlt
code ends
end
```

29. Write a program for finding largest number

```
Code segment
Assume cs:code
Mov si,4000h
Mov cx,0006h
Mov al,00h
Up:Cmp al,[si]
JAE down
Mov al,[si]
Inc si
Down: loop up
Inc si
Mov [si],al
Hlt
Code ends
end
```

30. Write a program for finding smallest number

```
Assume cs:code,ds:data
Data segment
List db 13h,12h,10h,04h,03h
Num db 05h
Result db ?
Data ends
Code segment
Mov ax,data
Mov ds,ax
Mov cl,num
Lea si,list
X:inc si
 Mov bl,[si]
 Cmp al,bl
 Jc next
 Mov al,bl
 Dec cl
 Next:loop x
 Mov result,al
 Int 03h
 Code ends
 end
```