1. **Input from user:**

```
.MODEL SMALL
.STACK 100H
.CODE

MAIN PROC
   ;INPUT A NUMBER
   MOV AH,1
   INT 21H
   MOV BL,AL

   ;print newline
   MOV AH,2
   MOV DL,13
   INT 21H
   MOV DL,10
   INT 21H


   ;INPUT ANOTHER NUMBER
   MOV AH,1
   INT 21H
   MOV BH,AL

   ;print newline
   MOV AH,2
   MOV DL,13
   INT 21H
   MOV DL,10
   INT 21H


   ;DISPLAY FIRST NUMBER

   MOV AH,2
   MOV DL,BL
   INT 21H

   ;print newline
   MOV AH,2
   MOV DL,13
   INT 21H
   MOV DL,10
   INT 21H
```

```
    ;DISPLAY SECOND VALUE
    MOV AH,2
    MOV DL,BH
    INT 21H

    EXIT:
    MOV AH,4CH
    INT 21H
    MAIN ENDP
END MAIN
```

- ☐ **When a number initialize this number print**

```
.MODEL SMALL
.STACK 100H
.DATA
MSG DB 3
MSG1 DB ?
.CODE

MAIN PROC
    MOV AX,@DATA
    MOV DS,AX

    MOV AH,2
    ADD MSG,48
    MOV DL,MSG
    INT 21H

    EXIT:
    MOV AH,4CH
    INT 21H


    MAIN ENDP
END MAIN
```

- ☐ **INPUT FROM USER AND SAVE A VARIABLE AND PRINT THIS**

```
.MODEL SMALL
.STACK 100H
.DATA
MSG DB 3
MSG1 DB ?
.CODE

MAIN PROC
    MOV AX,@DATA
    MOV DS,AX
MOV AH,2
ADD MSG,48
MOV DL,MSG
INT 21H

;NEWLINE
MOV AH,2
MOV DL,13
INT 21H
MOV DL,10
INT 21H

;store a value in msg1
MOV AH,1
INT 21H
MOV MSG1,AL

;PRINT NEWLINE
MOV AH,2
MOV DL,13
INT 21H
MOV DL,10
INT 21H

;DISPLAY
MOV AH,2
MOV DL,MSG1
INT 21H

EXIT:
MOV AH,4CH
INT 21H


MAIN ENDP
END MAIN
```

- ☐ **HOW TO INPUT A NUMBER AND HOW TO DISPLAY A STRING**

```
.MODEL SMALL
.STACK 100H
.DATA
M DB "HOW TO SHOW A STRING $"
.CODE
```

```
MAIN PROC
   ;1->SINGLE KEY INPUT
   ;2->SINGLE CHARACTER OUTPUT
   ;9->CHARACTER STRING OUTPUT

   MOV AX,@DATA
   MOV DS,AX


   ;HOW THIS TEXT WHICH IS STORE IN M VARIABLE
   MOV AH,9
   LEA DX,M
   INT 21H


   ;1->SINGLE KEY INPUT
   MOV AH,1
   INT 21H
   MOV BL,AL

   ;NEW LINE
   MOV AH,2
   MOV DL,13
   INT 21H
   MOV DL,10
   INT 21H

   ;2->SHOW SINGLE CHARACTER

   MOV AH,2
   MOV DL,BL
   INT 21H

   EXIT:
   MOV AH,4CH
   INT 21H

   MAIN ENDP
END MAIN
```

- ☐ **Print A-Z alphabet using Loop concept**

```
.MODEL SMALL
.STACK 100H
.DATA
A DB "lOOP CONCEPT $"
.CODE
```

```
MAIN PROC
   MOV AX,@DATA
   MOV DS,AX

   ;Print the loop concept message
   MOV AH,9
   LEA DX,A
   INT 21H

   ;Print newline
   MOV AH,2
   MOV DL,10
   INT 21H
   MOV DL,13
   INT 21H


   ;LOOP CONCEPT START(Print the alphabet A-Z)
   MOV CX,26
   MOV AH,2
   MOV DL, 'A'

   LEVEL1:
   INT 21H
   INC DL
   LOOP LEVEL1

   EXIT:
   MOV AH,4CH
   INT 21H
   MAIN ENDP
END MAIN
```

- ☐ **JMP Concept**

```
.MODEL SMALL
.STACK 100H
.DATA
A DB "JMP CONCEPT $"
B DB "ASSEMBLY LANGUAGE $"
C DB "PROGRAMMING $"
.CODE

MAIN PROC
   MOV AX,@DATA
   MOV DS,AX
   MOV AH,9
   LEA DX,A
   INT 21H

   MOV AH,2
   MOV DL,10
   INT 21H
   MOV DL,13
   INT 21H

  M:
  MOV AH,9
  LEA DX,B
  INT 21H
  JMP N

  N:
  MOV AH,9
  LEA DX,C
  INT 21H
  JMP EXIT

  EXIT:
  MOV AH,4CH
  INT 21H
  MAIN ENDP
END MAIN
```

- ☐ **Add two number**

```
.model small
.stack 100h
.data
a db "Enter first number:$"
b db "Enter second number:$"
c db "Summation of two number:$"

.code
main proc
    mov ax,@data
    mov ds,ax

    mov ah,9
    lea dx,a
    int 21h

    mov ah,1
    int 21h
    mov bl,al

    ;newline
    mov ah,2
    mov dl,10
    int 21h
    mov dl,13
    int 21h

    mov ah,9
    lea dx,b
    int 21h

    mov ah,1
    int 21h
    mov bh,al

    ;newline
    mov ah,2
    mov dl,10
    int 21h
    mov dl,13
    int 21h

    mov ah,9
    lea dx,c
    int 21h

    add bl,bh;bl=bl+bh
    sub bl,48
    mov ah,2
    mov dl,bl
    int 21h

    exit:
    mov ah,4ch
    int 21h
    main endp
end main
```