Important programs of 8086 (Exam point of view)

1. Write an ALP to find factorial of number for 8086.

```
        MOV AX, 05H
        MOV CX, AX
Back:   DEC CX
        MUL CX
        LOOP back
        ; results stored in AX
        ; to store the result at D000H
        MOV [D000], AX
        HLT
```

2. The 8 data bytes are stored from memory location E000H to E007H. Write 8086 ALP to transfer the block of data to new location B001H to B008H.

```
        MOV BL, 08H
        MOV CX, E000H
        MOV EX, B001H
Loop: MOV DL, [CX]
        MOV [EX], DL
        DEC BL
        JNZ loop
        HLT
```

3. Write a program to display string 'Electrical and Electronics Engineering' for 8086.

```
        Title display the string
        Dosseg
```

```
.model small
.stack 100h
.data
String1 db ‘Electrical and Electronics Engineering’, $
.code
Main proc
MOV AX, @data
MOV DS, AX
MOV AH, 09H
MOV DX, offset String1
INT 21H
MOV AH, 4CH
INT 21H
Main endp
End Main
```

4. Write a program to reverse the given string for 8086.

```
Title reverse the given string
Dosseg
.model small
.stack 100h
.data
String1 db ‘assembly language program’, $
Length dw $-String1-1
.code
```

```
        Main proc
        MOV AX, @data
        MOV DS, AX
        MOV SI, offset String1
        MOV  CX, Length
        ADD SI, CX

Back: MOV DL, [SI]
        MOV AH, 02H
        INT 21H
        DEC SI
        LOOP Back
        MOV AH, 4CH
        INT 21H
        Main endp
        End Main
```

5. Write a program to multiply 2 numbers (16-bit data) for 8086.

```
        Title multiply two numbers
        Dosseg
        .model small
        .stack 100h
        .data
        Multiplier dw 1234H
        Multiplicant dw 3456H
        Product dw ?
```

```
.code
MULT proc
MOV AX, @data
MOV DS, AX
MOV AX, Multiplicant
MUL Multiplier
MOV Product, AX
MOV Product+2, DX
MOV AH, 4CH
INT 21H
MULT endp
End MULT
```

6. Sum of series of 10 numbers and store result in memory location total.

```
Title Sum of series
Dosseg
.model small
.stack 100h
.data
List db 12,34,56,78,98,01,13,78,18,36
Total dw ?
.code
Main proc
MOV AX, @data
MOV DS, AX
MOV AX, 0000H
```

```
        MOV CX, 0AH ; counter
        MOV BL, 00H ; to count carry
        MOV SI, offset List
Back: ADD AL, [SI]
        JC Label
Back1: INC SI
        LOOP Back
        MOV Total, AX
        MOV Total+2, BL
        MOV AH, 4CH
        INT 21H
Label: INC BL
        JMP Back1
        Main endp
        End Main
```

7. Write a program to find Largest No. in a block of data. Length of block is 0A. Store the maximum in location result.

```
        Title maximum in given series
        Dosseg
        .model small
        .stack 100h
        .data
        List db 80, 81, 78, 65, 23, 45, 89, 90, 10, 99
        Result db ?
```

```
        .code
        Main proc
        MOV AX, @data
        MOV DS, AX
        MOV SI, offset List
        MOV AL, 00H
        MOV CX, 0AH
Back: CMP AL, [SI]
        JNC Ahead
        MOV AL, [SI]
Ahead: INC SI
        LOOP Back
        MOV Result, AL
        MOV AH, 4CH
        INT 21H
        Main endp
        End Main
```

8. Find number of times letter ‘e’ exist in the string ‘exercise’, Store the count at memory ans.

```
        Title string operation
        Dosseg
        .model small
        .stack 100h
        .data
```

```
        String db ‘exercise’, $
        Ans db ?
        Length db $-String
        .code
        Main proc
        MOV AX, @data
        MOV DS, AX
        MOV AL,00H
        MOV SI, offset String
        MOV CX, Length
Back: MOV BH, [SI]
        CMP BH, ‘e’
        JNZ Label
        INC AL
Label: INC SI
        LOOP Back
        MOV Ans, AL
        MOV AH, 4CH
        INT 21H
        Main endp
        End Main
```

9. Write an ALP to generate square wave with period of 200μs and address of output device is 55H for 8086 microprocessor.

```
START:  MOV AX, 01H
        OUT 30H, AX
```

```
        ; to generate loop for 200 μs using system frequency 5MHz
        MOV BX, Count       ;7T
Label: DEC BX               ;4T
        JNZ Label           ;10T/7T

        MOV AX, 00H
        OUT 30H, AX
        MOV BX, Count
Label1: DEC BX
        JNZ Label1
        JMP START
```

Note: Find the value of Count using technique used in 8085 so that delay will be of 200 μs.

10. Write an assembly language program to count number of vowels in a given string.

```
        Title to count number of vowels in given line of a text
        Dosseg
        .model small
        .stack 100h
        .code
        Main proc
        MOV AX, @data
        MOV DS, AX
        MOV SI, offset String       ;initialize p
        MOV CX, Len                 ;length in CX register
        MOV BL, 00                  ;vowel count=0
```

```
Back: MOV AL, [SI]
      CMP AL, 'a'
      JB VOW
      CMP AL, 'z'          ;Convert the character to upper case
      JA VOW
      SUB AL, 20H
VOW: CMP AL, 'A'
      JNZ a3
      INC BL
      JMP a2
a3: CMP AL, 'E'
      JNZ a4
      INC BL
      JMP a2
a4: CMP AL, 'I'
      JNZ a5
      INC BL
      JMP a2
a5: CMP AL, 'O'
      JNZ a6
      INC BL
      JMP a2
a6: CMP AL, 'U'
      JNZ a2
      INC BL
```

```
a2: INC SI

    LOOP Back

    MOV Vowel,  BL

    MOV AX, 4C00H

    INT 21H

    Main endp

    .data

    String db ‘The quick brown fox jumped over lazy sleeping dog’, $

    Len dw $-string

    Vowel db ?

    End Main
```

11. Write an 8086 ALP which will input the user name from the keyboard. If the user is ‘Pokhara’ it will output ‘The username is valid’ else it will output ‘Invalid user name’.

Note: This program is not verified in MASM so, please verify this program. This program can be done in the same approach as question 10, which is done above by comparing each character input.

```
title input name and comparision
dosseg
.model small
.stack 100h

.data
input db 7 dup(?)
comparestring db 'Pokhara','$'
outputstring1  db 'The username is valid','$'
outputstring2  db 'The username is invalid','$'

.code

main proc
mov ax, @data
mov ds, ax

; read string
mov dx, offset input
```

```
        mov ah,0ah
        int 21h
        ;string comparision
        mov si, offset input
        mov di, offset comparestring
        mov cx,07h              ;length of string in cx
        CLD                     ; DF-> direction flag clear i.e. autoincrement mode
        repe cmpsw              ;compare words of two string if equal then ZF will be set

        JZ label1
        mov dx, offset outputstring2
         jmp label2

label1: mov dx, offset outputstring1
 label2: mov ah, 0ah
        int 21h
        mov ah,4ch
        int 21h
        main endp
        end main
```