6.

a) Read two strings, store them in locations STR1 and STR2. Check whether they are equal or

not and display appropriate messages. Also display the length of the stored strings.

```
.model small

.stack

.data

 m2 db 10,13,"strings are not equal$"

 m3 db 10,13,"enter a string1$"

 m4 db 10,13,"enter a string2$"

 str1 db 20 dup(?)

 str2 db 20 dup(?)

 l1 db 00h

 l2 db 00h

 m6 db 10,13,"length of string 2 $"

.code

 mov ax,@data

 mov ds,ax

 lea dx,m3

 mov ah,09h

 int 21h

 lea si,str1

up: mov ah,01h

 cmp al,0dh

 je down1

 mov [si],al

 inc l1
```

```
 jmp up

down1:lea dx,m4

up2: mov ah,01h

down2:mov al,l1

up3: mov al,[si]

dm2: lea dx,m2

dm3:
```

b) Convert a 16-bit binary value (assumed to be an unsigned integer) to BCD and display it from

left to right and right to left for specified number of times on a 7-segment display interface.

```
.model small

.stack

.data

.code

Start: mov bl,10

 Lea di,m1

Back: mov si,di

 Push bx

 Call disp_m

 Call delay

 Pop bx

 Inc di

 Dec bl

 Jnz Back

 Mov bl,8

 Lea di,M1+8

Back1: mov si,di
```

```
 Push bx
 Call disp_m
 Call delay
 Pop bx
 Dec di
 Dec bl
 Jnz Back1
 Jmp Start
Bin_ssc PROC
 Lea si,bcd
 Mov ax,num
 Mov bx,10000
 Call conv
 Mov bx,100
 Call conv
 Mov bx,10
 Call conv
 Mov [si],dl
 Lea si,bcd
 Lea di,m1+8
 Lea bx,table
 Mov cx,5
Next: mov al,[si]
 Xlat
 Mov [di],al
 Dec di
 inc si
```

```
 Loop next
 Ret
Bin_ssc ENDP
Conv PROC
 Mov dx,0
 Div bx
 Mov [si],al
 Mov ax,dx
 Inc si
 RET
Conv ENDP
Disp_m PROC
 Mov cx,4
Next_char:
 Mov al,[si]
Next_bit:
 Mov dx,PB
 Out dx,al
 Push ax
 Mov al,00h
 Mov dx,PC
 Out dx,al
 Mov al,0ffh
 Mov dx,PC
 Out dx,al
 Pop ax
 Dec bl
```

```
 Jnz next_bit
 Inc si
 Loop next_char
 RET
Disp_m ENDP
Delay PROC
 Mov bx,0ffffh
 B2: mov cx,0ffffh
 B1:loop b1
 Dec bx
 Jnz b2
RET
Delay ENDP
End
```