```
. model small                    ; comparing two strings program

display macro msg
        lea dx, msg
        mov ah, 09h
        int 21h

endm
. data
msg1 db 0dh, 0ah, " enter first string        : $"
msg2 db 0dh, 0ah, " enter second string       : $"
msg3 db 0dh, 0ah, " length of first string: $"
msg4 db 0dh, 0ah, " length of second string: $"
msg5 db 0dh, 0ah, " --- strings are equal --- $"
msg6 db 0dh, 0ah, " --- strings are not equal --- $"

string1 db 80h dup (?)
string2 db 80h dup (?)

. code
start : mov ax, @data
        mov ds, ax
        display msg1
        ~~call readstr~~
        mov ~~s~~ si, offset string1
        call readstr
        mov bl, cl                ; store the length of the first string
        display msg2
        mov si, offset string2
        call readstr
```

```
push bx
push cx
display msg3
mov ~~At~~ al, bl
call len_dis
display msg4
mov al, cl
call len_dis
pop cx
pop bx
cmp cl, bl                    ; compare the lengths
jne fail                      ; if lengths are equal, process
                                next statement
mov si, offset string1
mov di, offset string2
cld
chk : mov al, [si]            ; compare both the string
      cmp al, [di]
      jne fail
      inc si
      inc di
      dec cl
      jnz chk
      display msg5
      jmp final
```

```
len-dis  proc near
            xor  ah,ah
            add  al,00h
            aam
            add  ax,3030h
            mov  bh,al
            mov  dl,ah
            mov  ah,02h
            int 21h
            mov  dl, bh
            mov  ah,02h
            int  21h

  ret
len-dis  endp
readstr  proc near
            xor  cl,cl
back:    mov ah,01h
            int 21h
            cmp al,0dh
            je finish
            mov [si],al
            inc si
            inc cl
            jmp back

finish : mov [si], byte ptr '$'
            ret
readstr  endp
fail: display msg6
```

final: mov ah, 4ch
int 21h

end start