READING A STRING (SERVICE 0Ah)

To read a string, we have to do two steps:

a. Define an array to store that string in DATA segment.

b. Invoke DOS function 0AH in CODE segment.

One way of defining the array is:

```
BUFFER_NAME DB Num1 , Num2 DUP(?)
```

where:

a. **Num1** = the maximum number of string characters to be read **+ 1**

b. **Num2 = Num1 + 1**

c. The maximum value for **Num1** is FFh i.e., 255

d. The last string character is the Carriage Return (0Dh).

The program will wait for the input. The user must press "Enter" key to end the inputting process. This is the reason for the point d above.

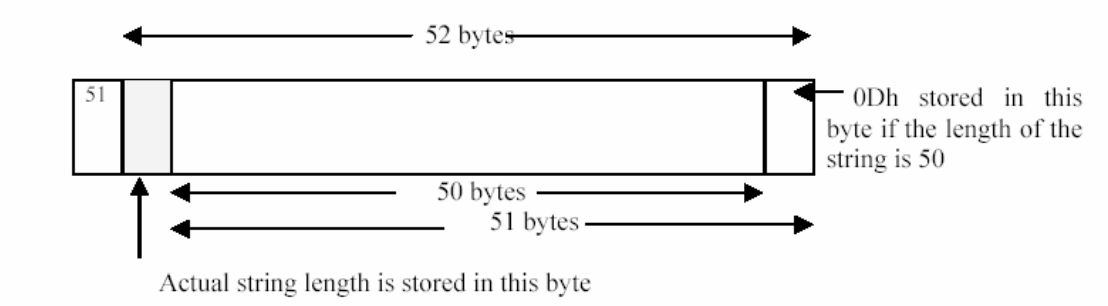
Example:
Define an array called STRING to store a string of maximum length 50 characters to be used by the service 0Ah of INT 21h.
Solution:

```
STRING DB 51, 52 DUP(?)
```

To read a string from the keyboard into an array called BUFFER as defined above, we invoke DOS function 0AH as:

```
MOV AH , 0AH
MOV DX , OFFSET STRING
INT 21H
```

The operation echoes the entered characters on the screen and advances the cursor. If more characters than the specified maximum (in byte 0) are entered, the speaker beeps and the additional characters are not read.



INDEXING AN ARRAY

Usually an array can be indexed by the register **DI, SI,** or **BX.**

Example:

```
. DATA
. . .
NUM DB 20, 4, 32, 50, 7, 15, 80, 12, 6, 125
  . . .
. CODE
MOV BH , NUM[0] ; BH is assigned the value 20
MOV AL , NUM[3] ; AL is assigned the value 50
MOV BX , 5
MOV AH , NUM[BX] ; AH is assigned the value 15
MOV SI , 7
MOV CL , NUM[SI + 2] ;CL is assigned the value 125
MOV DL , NUM[BX - 3] ;DL is assigned the value 32
MOV BX, 2
MOV AL, NUM[SI + BX] ;AL is assigned the value 125
```

Note: If the register DI, SI, or BX is initialized with the offset address of an array, the array elements can be accessed without using the array name:

```
MOV DI , OFFSET NUM
MOV CH , [DI] ; CH is assigned the value 20
INC DI
MOV AH , [DI] ; AH is assigned the value 4
```