

**Course Title:** Microprocessors and Assembly Language Lab (CSE-4504)  
Department of Computer Science and Engineering (CSE)  
**Islamic University of Technology (IUT), Gazipur**

**Lab # 04**

*Understanding Advanced 8086 I/O Instructions using Array in Assembly Language Program.*

**Objective:**

To understand some advanced 8086 instructions and getting familiar with the use of Array in Assembly Language Program.

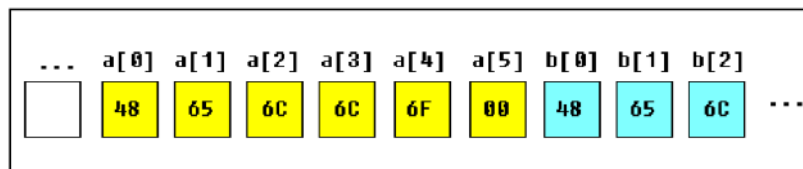
**Theory:**

• **Array**

Arrays can be seen as chains of variables. A text string is an example of a byte array; each character is presented as an ASCII code value (0..255). Here are some array definition examples:

```
a DB 48h, 65h, 6Ch, 6Ch, 6Fh, 00h
b DB 'Hello', 0
```

b is an exact copy of the an array, when compiler sees a string inside quotes it automatically converts it to set of bytes. This chart shows a part of the memory where these arrays are declared:



You can access the value of any element in array using square brackets, for example:

```
MOV AL, a[3]
```

You can also use any of the memory index registers **BX, SI, DI, BP**, for example:

```
MOV SI, 3
MOV AL, a[SI]
```

If you need to declare a large array with same value you can use **DUP** operator. The syntax for **DUP**: For example:

```
c DB 5 DUP(0)
c DB 0, 0, 0, 0, 0 ; is an alternative way of declaring:
```

one more example:

```
d DB 5 DUP(1, 2)
d DB 1, 2, 1, 2, 1, 2, 1, 2, 1, 2 ; is an alternative way of declaring:
```

Of course, you can use **DW** instead of **DB** if it's required to keep values larger then 255, or smaller then -128. **DW** cannot be used to declare strings!

### Assembly Language Program Example for Array:

To derive summation of a series  $1 + 2 + 3$  using array. Here, value of  $N$  is given by user where  $N=3$  and output is shown in the output window:

```
org 100h
.DATA                                     ; Data segment starts
A db 3, 1, 2                             ;1-D array for number
B db 00h
message db 'Enter the value of N:$'      ;1-D array for string
.CODE                                     ; Code segment starts
MAIN PROC
mov ax, @DATA
mov ds, ax
xor ax,ax
mov si, OFFSET A
mov di, OFFSET B
mov dx, OFFSET message ; Load Effective Address of the message in DX register
; lea dx, message ; (similar meaning like Load Effective Address)
mov ah, 09h                             ;display string function
int 21h                                  ;display message
mov ah, 01h
int 21h
mov cl, al
sub cl, 48                               ; to convert the ascii value of 3 to decimal 3
xor al, al
Loop_1:
    add al, [Si]
    inc Si
    loop Loop_1
mov bl, al
add bl, 48                               ; to convert the ascii value of the output to decimal
mov ah, 02h
mov dl, bl
int 21h
MAIN ENDP
END MAIN
RET
```

### Tasks to do:

1. Write an assembly language code to search the frequency of appearance of a given character from a declared array string (to do this **use ARRAY and Loop**). In your code, declare the string array named 'Input' and also display it as output.

### **Sample Input / Output:**

islamic university of technology  
Given Character: y  
Number of Frequency: 2