Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

AY: 2024-25

Class:	**SE**	**Semester:**	**IV**
Course Code:	**CSL404**	**Course Name:**	**Microprocessor Lab**

Name of Student:	**Shravani Sandeep Raut**
Roll No. :	**48**
Experiment No.:	**5**
Title of the Experiment:	**Program to find maximum number from given array**
Date of Performance:	**13/02/2025**
Date of Submission:	**20/02/2025**

Evaluation

Performance Indicator	Max. Marks	Marks Obtained
Performance	5	
Understanding	5	
Journal work and timely submission	10	
Total	20	

Performance Indicator	Exceed Expectations (EE)	Meet Expectations (ME)	Below Expectations (BE)
Performance	4-5	2-3	1
Understanding	4-5	2-3	1
Journal work and timely submission	8-10	5-8	1-4

Checked by

Name of Faculty : Ms. Sweety Patil

Signature :

Date:

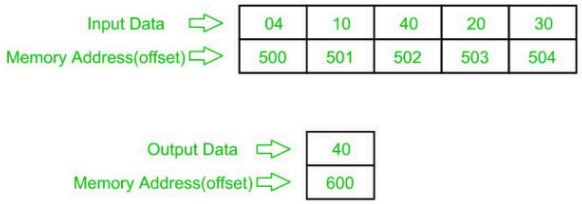
Aim: Assembly Language Program to find maximum number from a given array.

Theory: It is a simple and straightforward task that can be easily implemented in any programming language. It is a common operation used in many algorithms and applications, such as finding the maximum value in a data set or determining the winner of a game. It is a fast operation that can be completed in O(n) time complexity, where n is the number of elements in the array.

Algorithm:

1. Load data from offset 500 to register CL and set register CH to 00 (for count).
2. Load first number(value) from next offset (i.e 501) to register AL and decrease count by 1.
3. Now compare value of register AL from data(value) at next offset, if that data is greater than value of register AL then update value of register AL to that data else no change, and increase offset value for next comparison and decrease count by 1 and continue this till count (value of register CX) becomes 0.
4. Store the result (value of register AL) to memory address 2000 : 600.

Example:



Code:

```
org 100h
.data
arr db 10, 20, 30, 5, 8
m1 db 10, 13, 'Largest no is: $'

.code
mov ah, 09h
lea dx, m1
int 21h

lea si, arr
mov cx, 5
mov al, [si]
inc si
```

```
l1:
   cmp al, [si]
   jnc l2
   mov al, [si]

l2:
   inc si
   loop l1

   mov bl, al
   mov ah, 0
   mov cl, 10

   div cl
   add al, 30h
   mov dl, al
   mov ah, 02h
   int 21h

   mov al, bl
   mov ah, 0
   div cl
   mov al, ah
   add al, 30h
   mov dl, al
   mov ah, 02h
   int 21h

   mov ah, 4Ch
   int 21h

ret
```

Output:

Conclusion:

In this assembly language program, we efficiently find the maximum number in an array by comparing each element to the current maximum stored in the AL register. Using the CX register to track the number of elements left to compare, the program operates in O(n) time complexity. The final maximum value is stored at memory address 2000:600. This implementation showcases the effectiveness of assembly language for low-level operations, emphasizing direct memory access and register manipulation.

1. Explain the instructions AAS, IMUL, IDIV

   1. AAS (ASCII Adjust AX Before Division)
   Purpose: Adjusts the value in the AX register from ASCII representation to binary before division.

   2. IMUL (Integer Multiply)
   Purpose: Performs signed multiplication of integers.

   3. IDIV (Integer Divide)
   Purpose: Performs signed division of integers.

2. Explain the instructions AAM, AAD, CBW

   1. AAM (ASCII Adjust AX After Multiply)
   Purpose: Converts the binary value in the AX register to its ASCII representation after a multiplication operation, typically used to prepare the result for display or further processing as ASCII characters.

   2. AAD (ASCII Adjust AX Before Division)
   Purpose: Converts the binary values in the AL and AH registers to their ASCII representation before performing a division operation, ensuring that the division is done on the correct binary values.

   3. CBW (Convert Byte to Word)
   Purpose: Sign-extends the byte in the AL register to the word in the AX register, allowing for proper handling of signed values when performing arithmetic operations.