

**VIT**[®]**Vellore Institute of Technology**
(Deemed to be University under section 3 of UGC Act, 1956)**Experiment 2:**

Calculating the Average of the elements present in Array

Programme	:	BTech. CSE Core	Semester	:	Win 2021-22
Course	:	Microprocessor and Interfacing	Code	:	CSE2006
Faculty	:	Dr. Florence Gnana Poovathy J	Slot	:	L15+L16
Name	:	Hariket Sukesh Kumar Sheth	Register No.	:	20BCE1975

Average of Array Elements

Aim: To Perform Average of the elements present in the Entered Array

Tool Used: Assembler – MASM611

Algorithm:

Step 1: First of all, mount the c drive using the command: **mount c c:\masm611\bin**

Step 2: After pressing **enter**, type **c:** and press enter.

Step 3: Now give a command, **array.asm** for writing/editing the code and the write the code.

Step 4: A pop window appears; there we have to write out code(instructions) following the logic given below.

- a) Create an array (DB).
- b) Store "DATA" in DS using MOV command.
- c) CX is given the value 5H and AX is given the value 0H.
- d) Assign SI the beginning address of the array using LEA command.
- e) Within the loop the contents of SI are progressively added to AX, and CX is decremented by 1 with each iteration.
- f) End the loop using the JNZ command when CX reaches the value 0H.
- g) Outside the loop AX is divided by 5H which is moved into BL and divided using DIV.

Step 5: Now give a command, **masm array.asm** for running the code. The object file is created.

Step 6: Now give a command, **link array.obj** to link the object file to library file present in the bin folder.

Step 7: Press **ENTER** four times.

Step 8: Write debug **array.exe**

-u

-g (followed by the **address of HLT or INT** to view the values in registers).

Program:

```

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
File Edit Search Options ARRAY.ASM Help
DATA SEGMENT
    NUM1 DB 3H, 5H, 7H, 4H, 1H
DATA ENDS
CODE SEGMENT
    ASSUME DS:DATA, CS:CODE
START:
    MOV AX, DATA
    MOV DS, AX
    MOV CX, 5
    LEA SI, NUM1
    MOV AX, 0000H
LOOP1: ADD AX, [SI]
        INC SI
        DEC CX
        JNZ LOOP1
    MOV BL, 05H
    MOV AH, 00H
    DIV BL
    HLT
CODE ENDS
END START

```

<u>Sample Input:</u>	<u>Sample Output:</u>
Data: 3H, 5H, 7H, 4H, 1H	SUM= 20H Average = 20H/5H = 04H Hence, AX=0004H

Register / Memory Contents for I/O:

```

C:\>debug array.exe
-u
0765:0000 B86407      MOV     AX,0764
0765:0003 8ED8        MOV     DS,AX
0765:0005 B90500      MOV     CX,0005
0765:0008 8D360000     LEA     SI,[0000]
0765:000C B80000      MOV     AX,0000
0765:000F 0304        ADD     AX,[SI]
0765:0011 46          INC     SI
0765:0012 49          DEC     CX
0765:0013 75FA        JNZ     000F
0765:0015 B305        MOV     BL,05
0765:0017 B400        MOV     AH,00
0765:0019 F6F3        DIV     BL
0765:001B F4          HLT
0765:001C 40          INC     AX
0765:001D 7D22        JGE     0041
0765:001F 6C          DB     6C

```

Output:

```

-g 0765:001B
AX=0004 BX=0005 CX=0000 DX=0000 SP=0000 BP=0000 SI=0005 DI=0000
DS=0764 ES=0754 SS=0763 CS=0765 IP=001B  NU UP EI PL ZR NA PE NC
0765:001B F4          HLT

```



Sum of Array Elements

Aim: To Perform Sum of the elements present in the Arrays

Tool Used: Assembler – MASM611

Algorithm:

Step 1: First of all, mount the c drive using the command: **mount c c:\masm611\bin**

Step 2: After pressing **enter**, type **c:** and press enter.

Step 3: Now give a command, **sum.asm** for writing/editing the code and then write the code.

Step 4: A pop window appears; there we have to write out code(instructions) following the logic given below.

- a) Initialize 2 DB arrays NUM1 and NUM2, create a RESULT array too with 4 values of 0
- b) Move "DATA" in DS.
- c) SI is assigned the beginning address of NUM1, DI is assigned that of NUM2 and BX is assigned that of RESULT using LEA
- d) The contents of SI and DI is added and moved into the location represented by BX, via AL
- e) Then BX, SI and DL are all incremented within the loop using the LOOP1 statement

Step 5: Now give a command, **masm sum.asm** for running the code. The object file is created.

Step 6: Now give a command, **link sum.obj** to link the object file to library file present in the bin folder.

Step 7: Press **ENTER** four times.

Step 8: Write debug **sum.exe**

-u

-g (followed by the **address of HLT** to view the values in registers).

-d (followed by address of Data Segment and index 0 to 7 for viewing array values)

-d (followed by address of Data Segment and index 8 to 11 for viewing result)

Program:

```

SUM.ASM
DATA SEGMENT
    NUM1 DB 8H, 6H, 2H, 5H
    NUM2 DB 2H, 7H, 8H, 1H
    RESULT DB 4 DUP (0)
DATA ENDS
CODE SEGMENT
    ASSUME DS:DATA, CS:CODE
START:
    MOV AX, DATA
    MOV DS, AX
    LEA SI, NUM1
    LEA DI, NUM2
    LEA BX, RESULT
    MOV CX, 4H

LOOP1:
    MOV AL, [SI]
    ADD AL, [DI]
    MOV [BX], AL
    INC BX
    INC SI
    INC DI
    LOOP LOOP1
    HLT
CODE ENDS
END START

```

<u>Sample Input:</u>	<u>Sample Output:</u>
NUM1: 8H, 6H, 2H, 5H NUM2: 2H, 7H, 8H, 1H	Result: 08H+02H = 0AH ; 06H+07H = 0DH 08H+0AH = 0DH ; 05H+01H = 06H

Register / Memory Contents for I/O:

```

-u
0765:0000 B86407      MOV     AX,0764
0765:0003 8ED8        MOV     DS,AX
0765:0005 8D360000     LEA     SI,[0000]
0765:0009 8D3E0400     LEA     DI,[0004]
0765:000D 8D1E0800     LEA     BX,[0008]
0765:0011 B90400      MOV     CX,0004
0765:0014 8A04        MOV     AL,[SI]
0765:0016 0205        ADD     AL,[DI]
0765:0018 8B07        MOV     [BX],AL
0765:001A 43          INC     BX
0765:001B 46          INC     SI
0765:001C 47          INC     DI
0765:001D E2F5      LOOP    0014
0765:001F F4          HLT

```

Output:

```

-g 0765:001F
AX=0706 BX=000C CX=0000 DX=0000 SP=0000 BP=0000 SI=0004 DI=0008
DS=0764 ES=0754 SS=0763 CS=0765 IP=001F  NU UP EI PL NZ NA PO NC
0765:001F F4          HLT
-d 0764:0000 0007
0764:0000 08 06 02 05 02 07 08 01
-d 0764:0008 000B
0764:0008 0A 0D 0A 06

```