DEPARTMENT OF COMPUTING

COMPUTER ORGANIZATION AND ASSEMBLY LANGUAGE

COURSE INSTRUCTOR: DR. MUHAMMAD IMRAN

ASSIGNMENT 02

CS 235

NAME: MAHUM SAMAR

CMS ID : 290647

BSCS-9B

DESCRIPTION

This codes a one-dimensional array which is accessed in the form of a two-dimensional array. The code is divided into seven procedures. Following are the methods and their functionalities.

- rowMethod:

This method keeps track of the row of the matrix.

- columnMethod:

This method keeps track of the column of the matrix which is to be multiplied by the specific row.

- matrixValue:

In this method, the value from the row of the first matrix is multiplied by the column of the second matrix.

- ℵ MultiplyNumbers

This is the method to multiply the two numbers without using the multiplication operator. It multiplies numbers by using a loop and addition operator.

- ℵ updateMatrix:

When the value of an element is calculated then it is stored at the correct location of matrix3.

- ℵ printMatrix:

For printing the values on the console, this method uses a nested loop. The outer loop has a counter to 3 and keeping track of the row of the matrix. Three rows are printed containing three columns

- ℵ PrintRow:

In this method, the three elements are printed on the console.

CODE:

```
TITLE Multiplying Two 3x3 Matrices

Include irvine32.inc

;---------------------------------------------------------------------------------------------------------------------------

.data

;---------------------------------------------------------------------------------------------------------------------------


matrix1 BYTE 8,1,6,3,5,7,4,9,2                          ;1st matrix


matrix2  BYTE 1,3,9,3,8,3,8,2,2                         ;2nd matrix


matrix3  BYTE 9 dup (?)                     ;3rd matrix


result BYTE 0                                           ;storing the result of the element

                                                        ;of matrix 3
```

```
string1 BYTE 'Matrix 1: ',0

string2 BYTE 'Matrix 2: ',0

string3 BYTE 'Product: ',0



count BYTE 0                                          ;variable for keeping the track

                                                      ;how many values stored in the

                                                      ;matrix3


;------------------------------------------------------------------------------------------------------------------------------

.code

;------------------------------------------------------------------------------------------------------------------------------


Main Proc


;initializing all the registers

MOV eax,0

MOV ebx,0

MOV ecx,0

MOV edx,0

MOV esi,0

MOV edi,0


MOV esi,OFFSET matrix1                                ;moves OFFSET of matrix1 in esi

MOV edi,OFFSET matrix2                                ;moves OFFSET of matrix2 in edi
```

```
CALL rowMethod                                  ;calls the row method



Exit

Main endp


;-----------------------------------------------------------------------------------------------------------------------------

;rowMethod is used to keep the track of the row of the matix

;ecx is used as a counter

;after completing the elements for one row the esi is incremented to go on the next row

;columnMthod is call to calculate the elements

;-----------------------------------------------------------------------------------------------------------------------------


rowMethod PROC


MOV ecx,3


rowLoop:                                        ;loop for calling the columnMethod three times.

        CALL columnMethod

        ADD esi,3

loop rowLoop




call printMatrix                        ;method for printing the matrices on the console


RET                                             ;return

rowMethod ENDP
```

```
;------------------------------------------------------------------------------------------------------------------------------------

;columnMethod is used to keep track of the column which is to be multiplied by the row.

;ecx is used as a counter

;edi is used to keep track of matrix2 columns

;matrixValue is called to calculate each element of matrix3

;edi and ecx are pushed on the stack before changing its value

;------------------------------------------------------------------------------------------------------------------------------------


columnMethod PROC USES edi ecx                  ;method using push pop edi and ecx on stack


MOV ecx,3


columnLoop:                                     ;loop for calling matrixValue for calculating the single element

        CALL matrixValue

        inc edi                                 ;incrementing edi

loop columnLoop



RET

columnMethod ENDP


;------------------------------------------------------------------------------------------------------------------------------------

;matrixValue is used to calculate each element of the row of matrix3

;ecx is used as a counter

;ebx and edi are used to store the values which are to be multiplied

;MultipleNumbers is called to multiply the two numbers without using multiplication operator
```

```
;----------------------------------------------------------------------------------------------------------------------------------

matrixValue PROC USES esi edi ecx               ;method using push pop esi edi and ecx on stack

MOV ecx,3

rowValue:
            MOV bl, [esi]                        ;storing current value of matrix pointed by esi
            MOV dl, [edi]                        ;storing current value of matrix pointed by edi
            call MultiplyNumbers                 ;calling method for multiplying the numbers.
            ADD result, al                       ;saving result in result variable
            inc esi                              ;incremeted to go on next element of the row

            add edi,3                            ;incremented to go on the next column value
loop rowValue

MOV al,result                                    ;move final value of result into eax so that stored in matrix3
CALL updateMatrix                                ;call updateMatrix to update the matrix3
MOV result,0                                     ;result is made 0 for next calculation

RET
matrixValue ENDP

;----------------------------------------------------------------------------------------------------------------------------------
;MultiplyNumbers is used to multiply the two numbers without using multiplication operator
;one value is used as a counter for loop and
;other value is stored in the eax register and its added into itself until the loop executes
```

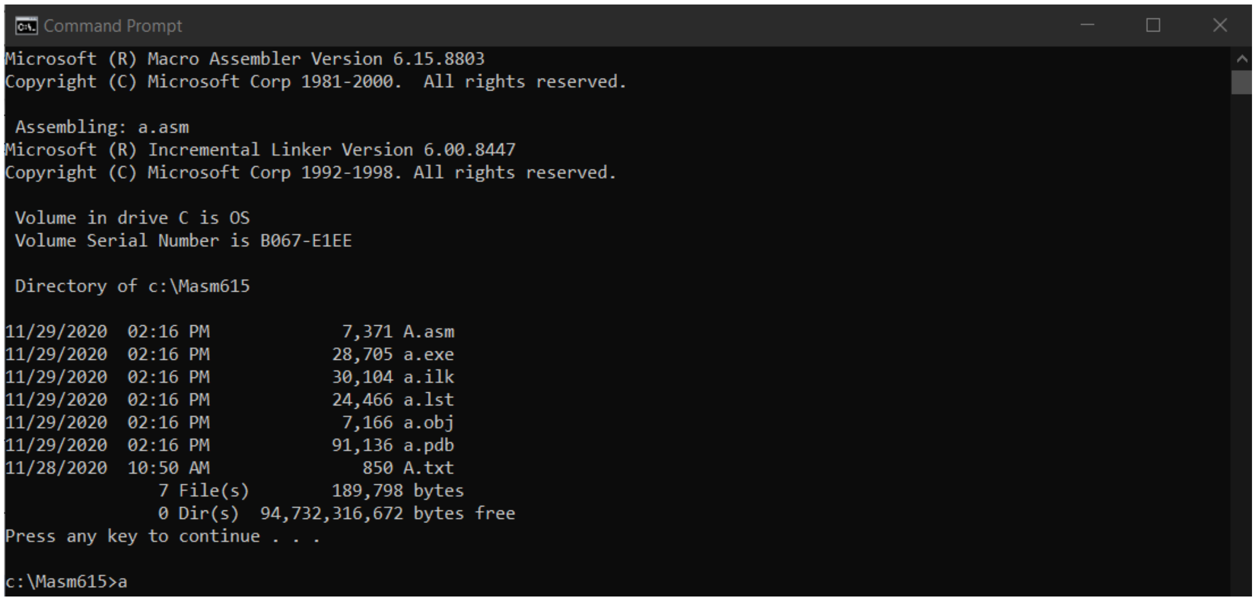
```
;-----------------------------------------------------------------------------------------------------------------------------



MultiplyNumbers PROC USES ecx                              ;matrix using push pop ecx

MOV eax,0

MOV cl, bl


Multiply:

            add al,dl

loop Multiply


RET

MultiplyNumbers ENDP


;-----------------------------------------------------------------------------------------------------------------------------

;updateMatrix is used to update the matrix3 after its each element is calculated

;-----------------------------------------------------------------------------------------------------------------------------


updateMatrix PROC USES esi ecx


MOV esi,OFFSET matrix3

MOV cl,count


l:

            inc esi

loop l
```

```
                                                            ;count variable to keep track of element of matrix3 where value is to
be stored.

MOV [esi],al

inc count


RET

updateMatrix ENDP


;-------------------------------------------------------------------------------------------------------------------------------

;printMatrix is used to display the matrices on the screen

;-------------------------------------------------------------------------------------------------------------------------------


printMatrix PROC


MOV edx, OFFSET string1                                     ;prompting matrix1

CALL writestring

CALL crlf


mov esi,OFFSET matrix1                                      ;printing matrix1 on screen

mov ecx,3

l1:

        CALL PrintRow

loop l1


MOV edx, OFFSET string2                                     ;prompting matrix2

CALL writestring

CALL crlf
```

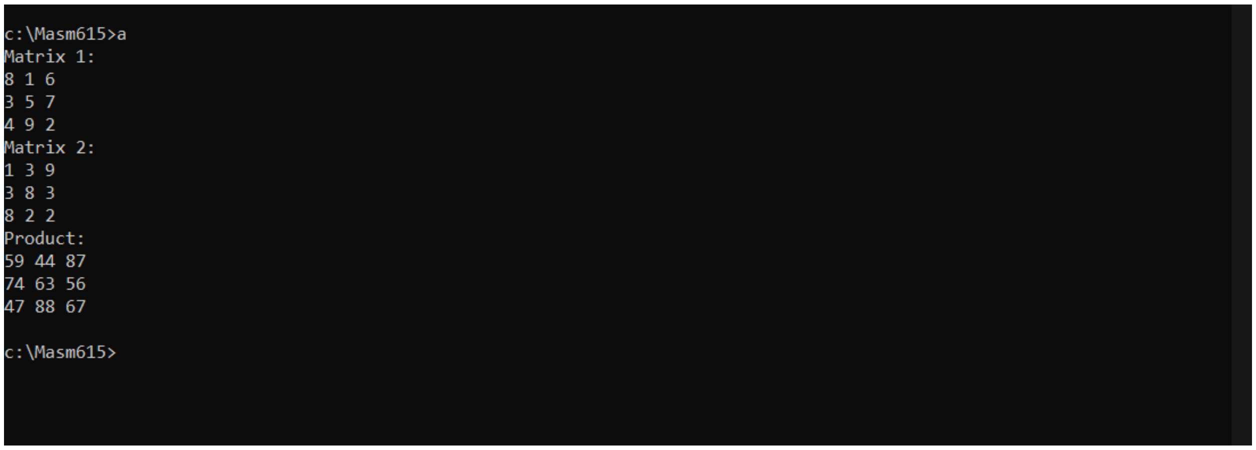
```
mov esi,OFFSET matrix2                          ;printing matrix2 on screen

mov ecx,3

l2:

        CALL PrintRow

loop l2


MOV edx, OFFSET string3                         ;prompting matrix3

CALL writestring

CALL crlf



mov esi,OFFSET matrix3                          ;printing matrix3 on screen

mov ecx,3

l3:

        CALL PrintRow

loop l3


RET

printMatrix ENDP


;------------------------------------------------------------------------------------------------------------------------

;PrintRow is used to print a single row containing 3 elements at a time.

;------------------------------------------------------------------------------------------------------------------------


PrintRow PROC USES ecx                          ;method using push pop ecx
```

```
MOV ecx,3


l4:
        MOV al,[esi]
        CALL writedec
        MOV eax,32
        CALL writechar
        inc esi
loop l4


CALL crlf


RET
PrintRow ENDP


;-----------------------------------------------------------------------------------------------------------------------------
End main
;-----------------------------------------------------------------------------------------------------------------------------
```

OUTPUT:



```
Microsoft (R) Macro Assembler Version 6.15.8803
Copyright (C) Microsoft Corp 1981-2000.  All rights reserved.

 Assembling: a.asm
Microsoft (R) Incremental Linker Version 6.00.8447
Copyright (C) Microsoft Corp 1992-1998. All rights reserved.

 Volume in drive C is OS
 Volume Serial Number is B067-E1EE

 Directory of c:\Masm615

11/29/2020  02:16 PM             7,371 A.asm
11/29/2020  02:16 PM            28,705 a.exe
11/29/2020  02:16 PM            30,104 a.ilk
11/29/2020  02:16 PM            24,466 a.lst
11/29/2020  02:16 PM             7,166 a.obj
11/29/2020  02:16 PM            91,136 a.pdb
11/28/2020  10:50 AM               850 A.txt
               7 File(s)        189,798 bytes
               0 Dir(s)  94,732,316,672 bytes free
Press any key to continue . . .

c:\Masm615>a
```



```
c:\Masm615>a
Matrix 1:
8 1 6
3 5 7
4 9 2
Matrix 2:
1 3 9
3 8 3
8 2 2
Product:
59 44 87
74 63 56
47 88 67

c:\Masm615>
```