Name	Adrian Leo Pradana
NPM	2106718344

1. Two dimensional array is a list of one dimensional arrays. To declare two dimensional integer array of size [x][y], could be write as follows

   Type arrayName [x] [y];

   For example

   Int list [2][3] = {
   {0,1,2},
   {3,4,5}
   };

   The following initialization is equivalent to

   Int list[2][3] = {0,1,2,3,4,5};

   Below is how to access the array and the value base on following array :

	Column 0	Column 1	Column 2
Row 0	list[0][0] = 0	list[1][0] = 1	list[2][0] = 2
Row 1	list[1][0] = 3	list[1][1] = 4	list[2][1] = 5

   References:

   - https://www.javatpoint.com/two-dimensional-array-in-c

2. Yes we can make more than two dimensional array its called multidimensional array, for example we want to make three dimensional array, to initialize is the same with two dimensional arrays the only difference is as the number of dimensions increases so the number of nested braces will also increase. Below is the example :

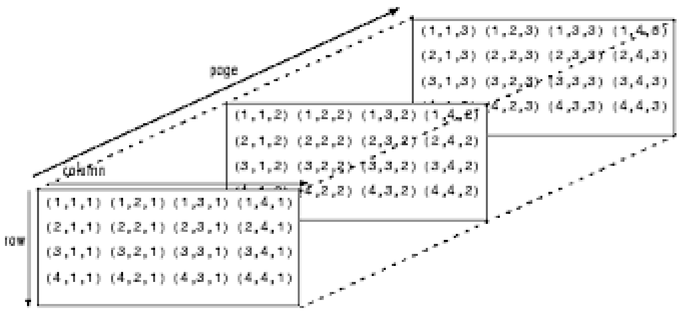
   Int dict[2][3][2] = {
   {{1,2} , {3,4} , {5,6}},
   {7,8},{9,10},{11,12}
   };

   The following initialization is equivalent to

   Int dict[2][3][4] = {

```
1,2,3,4,5,6,7,8,9,10,11,12
};
```

Accessing elements in Three-Dimensional Arrays is also similar to that of Two-Dimensional Arrays. The difference is we have to use three loops instead of two loops for one additional dimension in Three-dimensional Arrays. Below is three dimensional array illustration :



References:

- https://www.mathworks.com/help/matlab/math/multidimensional-arrays.html

3. The output is 123450, in this code it is using two loops in the nested forms. In the nested loop the outer loop represents the row and the inner loop represent the column and to print the 2D array, write displaying logic inside the inner loop.

   References:

   - https://www.tutorialspoint.com/cprogramming/c_arrays.htm

4. The output is not consistent this is happen because By the C standard, it explicitly has undefined behavior. The array indexing operation a[i] gains its meaning from the following features of C, The syntax ary[i] is equivalent to *(ary + i). In effect, array-indexing is a special case of pointer-indexing. Since a pointer can point to any place inside an array, any arbitrary expression that looks like ary[-1] is not wrong by examination, and so compilers don’t consider all such expressions as errors.

   References:

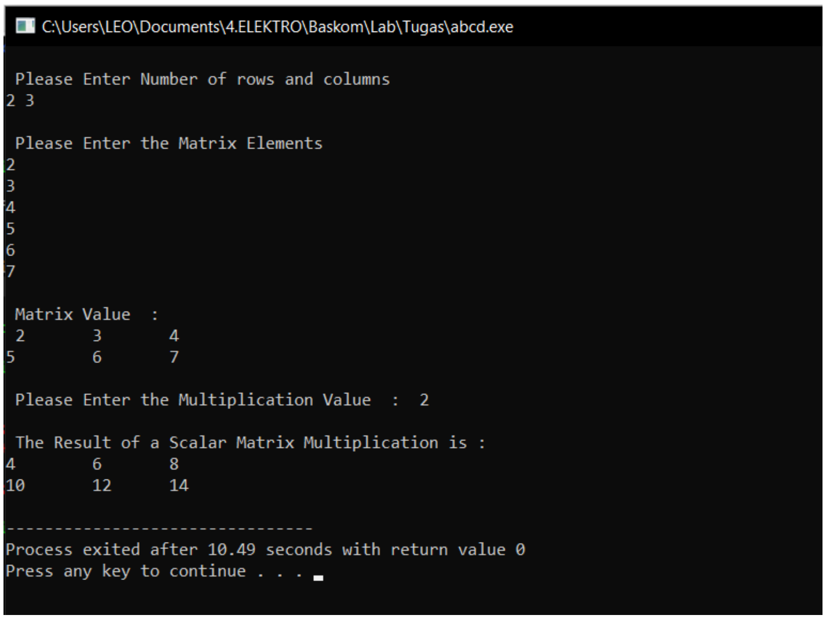
   - https://hackernoon.com/negative-indexing-multi-arrays-in-c-b551ce252972

5. Below is the program :

   ```
   #include<stdio.h>
   ```

```
int main()
{
        int i, j, rows, columns, Multiplication[10][10], N;

        printf("\n Please Enter Number of rows and columns\n");
        scanf("%d %d", &i, &j);

        printf("\n Please Enter the Matrix Elements \n");
        for(rows = 0; rows < i; rows++)
        {
                for(columns = 0;columns < j;columns++)
        {
                scanf("%d", &Multiplication[rows][columns]);
        }
        }
        printf("\n Matrix Value  : \n ");
                for(rows = 0; rows < i; rows++)
        {
                for(columns = 0; columns < j; columns++)
        {
                printf("%d \t ", Multiplication[rows][columns]);
        }
        printf("\n");
        }

        printf("\n Please Enter the Multiplication Value  :  ");
        scanf("%d", &N);

        for(rows = 0; rows < i; rows++)
        {
                for(columns = 0; columns < j; columns++)
        {
                Multiplication[rows][columns] = N * Multiplication[rows][columns];
                }
        }

        printf("\n The Result of a Scalar Matrix Multiplication is : \n");
        for(rows = 0; rows < i; rows++)
        {
                for(columns = 0; columns < j; columns++)
        {
                printf("%d \t ", Multiplication[rows][columns]);
        }
        printf("\n");
        }
        return 0;


}
```

THE OUTPUT :



```
C:\Users\LEO\Documents\4.ELEKTRO\Baskom\Lab\Tugas\abcd.exe

 Please Enter Number of rows and columns
2 3

 Please Enter the Matrix Elements
2
3
4
5
6
7

 Matrix Value  :
 2       3       4
5       6       7

 Please Enter the Multiplication Value  :  2

 The Result of a Scalar Matrix Multiplication is :
4       6       8
10      12      14

--------------------------------
Process exited after 10.49 seconds with return value 0
Press any key to continue . . .
```

References:

- https://www.javatpoint.com/matrix-multiplication-in-c