

Institute of Computer Technology
B. Tech. Computer Science and Engineering

Sub: ESFP – I
Course Code: 2CSE102
Practical – 14

Name:Dwij vatsal desai
Roll No:BDA-08
Branch:BDA
Class: B
Batch:14

Q.1.Problem Definition:

Make a program to compute and print a multiplication table for number
1 to 5

using a 2-dimensional array.

Expected Output should be:

Multiplication Table

```
1 2 3 4 5
-----
1| 1 2 3 4 5
2| 2 4 6 8 10
3| 3 6 9 12 15
4| 4 8 12 16 20
5| 5 10 15 20 25
```

Code:

```
#include <stdio.h>

int main()
{
```

```
printf("Multiplication Table\n");
int n = 5;
int table[5][5];

for (int i = 0; i < n; i++)
{
    for (int j = 0; j < n; j++)
    {
        table[i][j] = (i + 1) * (j + 1);
    }
}

printf(" ");
for (int i = 1; i <= n; i++)
{
    printf("%d ", i);

printf("\n-----\n");
for (int i = 1; i <= n; i++)
{

    printf("%d| ", i);
    for (int j = 0; j < n; j++)
    {
        printf("%d ", table[i - 1][j]);
    }
    printf("\n");

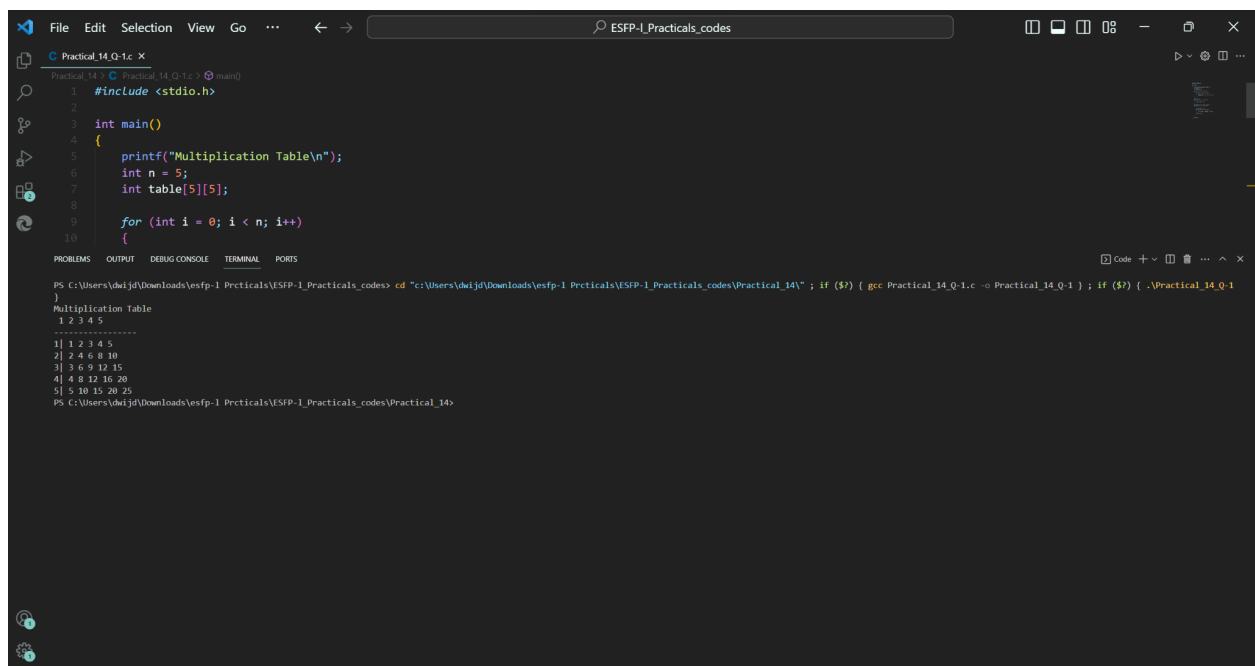
}

return 0;
```

```
}
```

Output:

```
PS C:\Users\dwijd\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes> cd "c:\Users\dwijd\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes\Practical_14"
}
Multiplication Table
 1 2 3 4 5
-----
1| 1 2 3 4 5
2| 2 4 6 8 10
3| 3 6 9 12 15
4| 4 8 12 16 20
5| 5 10 15 20 25
PS C:\Users\dwijd\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes\Practical_14>
```



```
1 #include <stdio.h>
2
3 int main()
4 {
5     printf("Multiplication Table\n");
6     int n = 5;
7     int table[5][5];
8
9     for (int i = 0; i < n; i++)
10    {
11        for (int j = 0; j < n; j++)
12        {
13            table[i][j] = (i+1)*(j+1);
14        }
15    }
16
17    for (int i = 0; i < n; i++)
18    {
19        for (int j = 0; j < n; j++)
20        {
21            printf("%d ", table[i][j]);
22            if (j % 4 == 3) printf("\n");
23        }
24    }
25}
```

```
PS C:\Users\dwijd\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes> cd "c:\Users\dwijd\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes\Practical_14" ; if ($?) { gcc Practical_14_Q-1.c -o Practical_14_Q-1 } ; if ($?) { .\Practical_14_Q-1 }
Multiplication Table
 1 2 3 4 5
-----
1| 1 2 3 4 5
2| 2 4 6 8 10
3| 3 6 9 12 15
4| 4 8 12 16 20
5| 5 10 15 20 25
PS C:\Users\dwijd\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes\Practical_14>
```

Q.2.Problem Definition:

Find out the calculation of array elements like (addition, subtraction, multiplication and division) using a 2-D array.

Code:

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

```
int main()
{
    int array1[3][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
    int array2[3][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
    int result[3][3];

    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            result[i][j] = array1[i][j] + array2[i][j];
        }
    }

    printf("Addition Result:\n");
    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            printf("%d ", result[i][j]);
        }
        printf("\n");
    }

    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            result[i][j] = array1[i][j] - array2[i][j];
        }
    }

    printf("\nSubtraction Result:\n");
```

```
for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 3; j++)
    {
        printf("%d ", result[i][j]);
    }
    printf("\n");
}

for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 3; j++)
    {
        result[i][j] = array1[i][j] * array2[i][j];
    }
}

printf("\nMultiplication Result:\n");
for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 3; j++)
    {
        printf("%d ", result[i][j]);
    }
    printf("\n");
}

for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 3; j++)
    {
        result[i][j] = array1[i][j] / array2[i][j];
    }
}
```

```

    }
}

printf("\nDivision Result:\n");
for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 3; j++)
    {
        printf("%d ", result[i][j]);
    }
    printf("\n");
}

return 0;
}

```

Output:

The screenshot shows the Visual Studio Code interface with a C program open in the editor. The program calculates the addition, subtraction, multiplication, and division of two 3x3 matrices. The terminal output displays the results of these operations.

```

1 #include <stdio.h>
2 int main()
3 {
4     int array1[3][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
5     int array2[3][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
6     int result[3][3];
7
8     for (int i = 0; i < 3; i++)
9     {
10        for (int j = 0; j < 3; j++)

```

Terminal Output:

```

PS C:\Users\dwijd\Downloads\esfp-1 Prcticals\ESFP-1 Practicals_codes> cd "c:\Users\dwijd\Downloads\esfp-1 Prcticals\ESFP-1 Practicals_codes\Practical_14\"; if ($?) { gcc Practical_14_Q-2.c -o Practical_14_Q-2 }; if ($?) { .\Practical_14_Q-2 }
Addition Result:
2 4 6
8 10 12
14 16 18

Subtraction Result:
0 0 0
0 0 0
0 0 0

Multiplication Result:
1 4 9
10 25 36
49 64 81

Division Result:
1 1 1
1 1 1
1 1 1

```

```

PS C:\Users\dwijd\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes> cd "c:\Users\dwijd\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes\Practical_14_Q-2" ; if ($?) { .\Practical_14_Q-2 }
Addition Result:
2 4 6
8 10 12
14 16 18

Subtraction Result:
0 0 0
0 0 0
0 0 0

Multiplication Result:
1 4 9
16 25 36
49 64 81

Division Result:
1 1 1
1 1 1
1 1 1
PS C:\Users\dwijd\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes\Practical_14>

```

Q.3.Problem Definition:

Make a program in C to accept any random 16 number from user in array list in the format of 4*4 matrix, and then calculate transpose of matrix and then print these all transposed array element in new array using 2-D array.

Code:

```

#include <stdio.h>
int main()
{
    int matrix[4][4];
    int transpose[4][4];

    printf("Input 4x4 Matrix \n");
    for (int i = 0; i < 4; i++)
    {

```

```
        for (int j = 0; j < 4; j++)
        {
            scanf("%d", &matrix[i][j]);
        }
    }

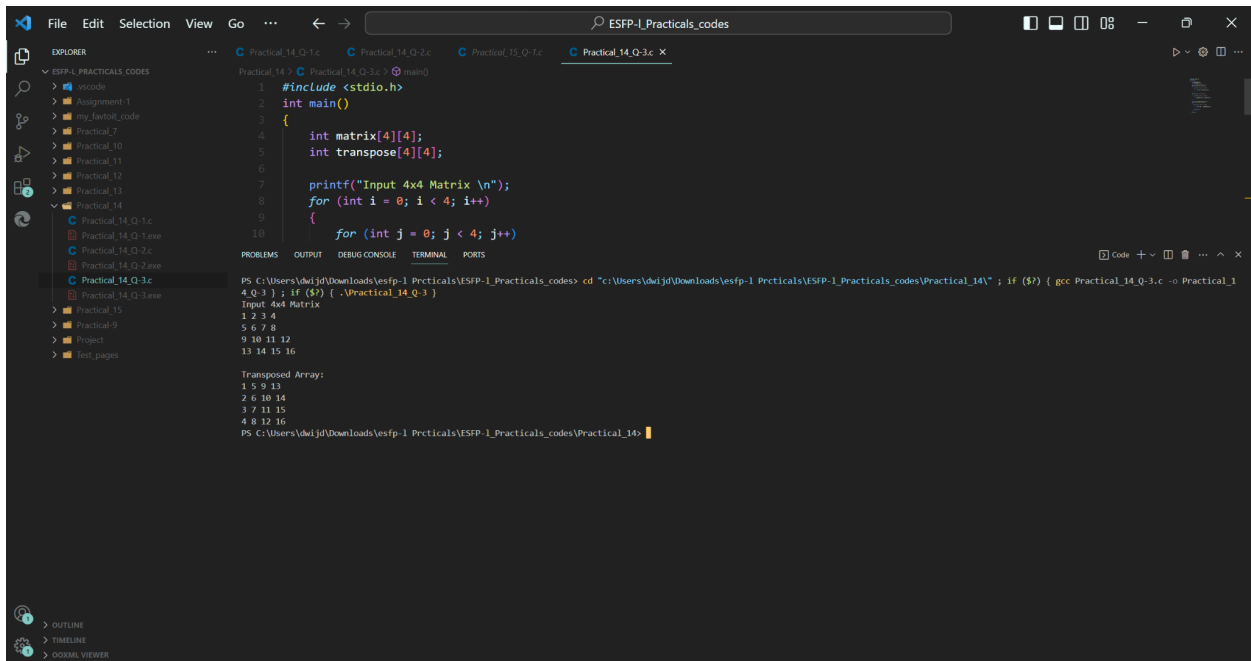
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            transpose[i][j] = matrix[j][i];
        }
    }

    printf("\nTransposed Array:\n");
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            printf("%d ", transpose[i][j]);
        }

        printf("\n");
    }

    return 0;
}
```

Output:



```
#include <stdio.h>
int main()
{
    int matrix[4][4];
    int transpose[4][4];

    printf("Input 4x4 Matrix \n");
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            // Input logic would go here
        }
    }

    // Transpose logic would go here

    printf("Transposed Array:\n");
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            // Output logic would go here
        }
    }
}
```

PS C:\Users\dwij\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes> cd "c:\Users\dwij\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes\Practical_14" ; if (\$?) { gcc Practical_14_Q-3.c -o Practical_14_Q-3 } ; if (\$?) { .\Practical_14_Q-3 }
Input 4x4 Matrix
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16

Transposed Array:
1 5 9 13
2 6 10 14
3 7 11 15
4 8 12 16
PS C:\Users\dwij\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes\Practical_14>

```
PS C:\Users\dwij\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes> cd "c:\Users\dwij\
4_Q-3 } ; if ($?) { .\Practical_14_Q-3 }
Input 4x4 Matrix
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16

Transposed Array:
1 5 9 13
2 6 10 14
3 7 11 15
4 8 12 16
PS C:\Users\dwij\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes\Practical_14>
```

Q.4.Problem Definition:

Make a program in C to accept any random 9 number from user in array list in the format of 4*4 matrix, and then calculate both diagonal element addition and then print these diagonal element addition result value on screen/terminal using 2-D array.

Code:

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

```

int main()
{
    int matrix[4][4];
    int sum_diagonal1 = 0;
    int sum_diagonal2 = 0;

    printf("Input 4x4 Matrix :\n");
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            scanf("%d", &matrix[i][j]);
        }
    }
    for (int i = 0; i < 4; i++)
    {
        sum_diagonal1 += matrix[i][i];
        sum_diagonal2 += matrix[i][3 - i];
    }

    printf("\nBoth diagonal array element addition
is=%d\n", sum_diagonal1 + sum_diagonal2);
    return 0;
}

```

Output:

The screenshot shows the Visual Studio Code interface with a C program open in the editor. The Explorer sidebar on the left shows a project structure with folders for 'ESFP-1_PRACTICALS_CODES' and 'Practical_14'. The main editor window displays the source code for 'Practical_14_Q-4.c'. The code includes `<stdio.h>` and defines a `main` function. Inside `main`, a 4x4 integer matrix is declared, and two variables, `sum_diagonal1` and `sum_diagonal2`, are initialized to 0. A `printf` statement prompts the user to 'Input 4x4 Matrix :'. A `for` loop iterates from `i = 0` to `i < 4`, incrementing `i` by 1 in each iteration. The output window at the bottom shows the execution results, including the input matrix and the calculated sum of diagonal elements.

```
1 #include <stdio.h>
2 int main()
3 {
4     int matrix[4][4];
5     int sum_diagonal1 = 0;
6     int sum_diagonal2 = 0;
7
8     printf("Input 4x4 Matrix :\\n");
9     for (int i = 0; i < 4; i++)
10    {
11
12    }
13 }
```

PS C:\Users\dwijjd\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes> cd "c:\Users\dwijjd\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes\Practical_14"; if (\$?) { gcc Practical_14_Q-4.c -o Practical_14_Q-4.exe; if (\$?) { .\Practical_14_Q-4.exe } }

Input 4x4 Matrix :

```
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
```

Both diagonal array element addition is=68

PS C:\Users\dwijjd\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes\Practical_14>

```
PS C:\Users\dwijjd\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes> cd "c:\Users\dwijjd\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes\Practical_14"; if ($?) { gcc Practical_14_Q-4.c -o Practical_14_Q-4.exe; if ($?) { .\Practical_14_Q-4.exe } }
```

Input 4x4 Matrix :

```
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
```

Both diagonal array element addition is=68

PS C:\Users\dwijjd\Downloads\esfp-1 Prcticals\ESFP-1_Practicals_codes\Practical_14>