Week-1

WAP to pass the matrices as parameters and do following function.

(i) add and subtract
(ii) multiply
(iii) Sum of principle diagonal
(iv) Sum of rows and columns
(v) Transpose
(vi) Check given matrix is symmetric or not.

```
# include <stdio.h>
Void addsub [int c[2][2], int d[2][2])
{
    int f[2][2], f1[2][2];
    printf (" the sum of matrix is : \n");
     for (int i=0 ; i<2 ; i++)
{
    for (int j=0 ; j<2; j++)
    {
        f[i][i] = c[i][j] + d[i][j];
      printf (" %d", f[i][j]);
      }

  printf ("\n");
     }

printf (" the subtraction of matrix is :\n");
     for (int i=0 ; i<2 ; i++)
  { for (int j=0 ; j<2 ; j++)
  {  f1[i][j] = c[i][j] - d[i][j];
     printf ("%d", f1[i][j]);
          }
```

```
        printf("\n");
          {
          }

Void multiply (int c[2][2], int d[2][2])
  {
     int i, j, k, mul[2][2];
  printf("multiply of matrix:\n");
   for (i=0; i<2; i++)
  {
     for (j=0; j<2; j++)
      {
          mul[i][j] = 0;
  for (k=0; k<2; k++)
     {
        mul[i][j] += c[i][k] * d[k][j];
          }
        } }
   for (i=0; i<2; i++)
    { for (j=0; j<2; j++)
     { printf("%d \t", mul[i][j]);
          {
        printf("\n");
            }}

Void Sumpreinciding (int c[2][2])
  { int sum=0;
      for (int i=0; i<2; i++)
  {   for (int j=0; j<2; j++)
      { if (i==j)
      { sum += c[i][j];
          {
            }}
```

```
Printf ("Sum of principal diagonal element
        is %d\n", sum); }

Void rowcolsum (int mat[][100], int r, int c)
{ for (int i=0 ; i<r ; i++)
  {
    int rsum =0 ;
    for (int j=0; j<c ; j++)
    {
      rsum += mat [i][j];
    }
Printf ("Sum of elements in row %d; %d\n",
        i+1, r);
  }
for (int j=0; j<c; j++)
  { int csum =0;
    for (int i=0 ; i<r ; i++)
    {
      csum += mat [i][j];
    }
Printf (" Sum of elements in column %d: %d\n
        j+1, c);
  }}

Void transpose (int mat[][100], int r, int c)
{ printf (" transpose of matrix:\n");
  for (int j=0 ; j<c ; j++)
  {
    for (int i=0 ; i<r ; i++)
    {
```

```
        printf("%d\t", mat[i][j]);
    }
    printf("\n");
}

int issymmetric(int mat[][20], int rows, int cols)
{
    if(rows != cols)
    {
        printf("not symmetric");
    }
for (int i=0; i<rows; i++)
{
    for (int j=0; j<cols; j++)
{
    if(matrix[i][j] != matrix[j][i])
{
    printf("not symmetric");
}}}

printf("matrix is symmetric");
}
```

PAGE NO :
DATE :

Output

enter elements of 1st matrix
0 1
1 0
enter the elements of 2nd matrix
2 3
3 2

1. add & Sub 2. multiply 3. Sum of principle diagonal.
4. row column Sum 5. Transpose 6. Symmetric check 7. exit.

1.
the sum of matrix is :
2 4
4 2

2. the subtraction of matrix is :
-2 -2
-2 -2

2.
Multiply of matrix :
3 2
2 3

3.
Sum of principal diagonal element is 0.

4.

Sum of elements in row 1 : 1
Sum of elements in row 2 : 1
Sum of elements in column 1 : 1
Sum of elements in column 2 : 1

5.

0 1
1 0
6
matrix is symmetric.

```
int main()
{
  int a[2][2], b[2][2];
  int i, j;
  int ch;
  printf("enter the elements of 1st matrix");
   for(i=0 ; i<2 ; i++)
{
   for(j=0 ; j<2 ; j++)
{
     scanf("%d", &a[i][j]);
  }
}

printf("enter the elements of 2nd matrix");
   for(i=0 ; i<2 ; i++)
{
   for(j=0 ; j<2 ; j++)
   {
     scanf("%d", &b[i][j]);
   }
```

```
while (1)
{
  printf (" 1. add & sub  2. multiply 3. Sum of
    principal diagonal 4. row column Sum 5. Transpose
        6. Symmetric check 7. exit);
  scanf ("%d", &ch);
  Switch (ch)
{
   Case 1:
        addsub (a, b);
        break;

   Case 2:
        multiply (a, b);
        break;
   Case 3:
        Sumprincidiag (a);
          break;
   Case 4:
        ~~Transpose (a, 2, 2);~~ rowColumnSum (a, 2, 2);
        break;

   Case 5:
          Transpose (a, 2, 2);
          break;

   Case 6:
          isSymmetric (a, 2, 2);
            break;
   Case 7:
          return 0;
  }
 }
}
```

21/6/23