Write a program to pass matrices as parameters in the program to perform the following:

- Addition and subtraction
- Sum of diagonal elements
- Sum of rows and coloumns
- Transpose
- Check whether the matrix is symmetric

CODE:

```
#include <stdio.h>
#include <conio.h>
void add(int a[10][10], int b[10][10], int sum[10][10], int r, int c)
  for (int i = 0; i < r; i++)
      for (int j = 0; j < c; j++)
        sum[i][j] = a[i][j] + b[i][j];
void diff(int a[10][10], int b[10][10], int diff[10][10], int r, int c)
  for (int i = 0; i < r; i++)
      for (int j = 0; j < c; j++)
        diff[i][j] = a[i][j] - b[i][j];
void multiply(int a[10][10], int b[10][10], int mul[10][10], int r, int c)
  for (int i = 0; i < r; i++)
      for (int j = 0; j < c; j++)
```

```
{
       mul[i][j] = 0;
        for (int k = 0; k < c; k++)
          mul[i][j] += a[i][k] * b[k][j];
     }
  }
void rc_sum(int a[10][10], int r, int c)
  int rsum, csum;
  printf("Row sum:\n");
  for (int i = 0; i < r; i++) {
     rsum = 0;
     for (int j = 0; j < c; j++) {
        rsum += a[i][j];
     printf("Row %d: %d\n", i + 1, rsum);
  }
  printf("Coloumn sum:\n");
  for (int j = 0; j < \text{columns}; j++) {
     csum = 0;
     for (int i = 0; i < r; i++) {
       csum += a[i][j];
     printf("Column %d: %d\n", j + 1, csum);
  }
}
void transpose(int a[10][10], int r, int c) {
  int trans[10][10];
  for (int i = 0; i < r; i++) {
     for (int j = 0; j < c; j++) {
        trans[j][i] = a[i][j];
     }
  }
  printf("Transpose Matrix\n");
  for (int i = 0; i < c; i++)
     for (int j = 0; j < r; j++)
```

```
printf("%d", trans[i][j]);
  }
}
int isSymmetric(int a[10][10], int r, int c) {
  if (r!=c)
     {
     return 0;
    }
  for (int i = 0; i < r; i++)
     for (int j = 0; j < c; j++)
       if (a[i][j] != b[j][i])
          return 0;
     }
  }
  return 1;
}
int main() {
  int matrix1[100][100], matrix2[100][100], result[100][100];
  int rows1, cols1, rows2, cols2;
  int choice;
  do {
     printf("Matrix Operations:\n");
     printf("1. Addition\n");
     printf("2. Subtraction\n");
     printf("3. Multiplication\n");
     printf("4. Sum of Diagonals\n");
     printf("5. Sum of Rows and Columns\n");
     printf("6. Transpose\n");
     printf("7. Check Symmetry\n");
     printf("0. Exit\n");
     printf("Enter your choice: ");
     scanf("%d", &choice);
     switch (choice) {
       case 1:
          printf("Enter the number of rows and columns of the matrices: ");
```

```
scanf("%d %d", &rows1, &cols1);
  printf("Enter elements of matrix1:\n");
  for (int i = 0; i < rows1; i++) {
     for (int j = 0; j < cols1; j++) {
       scanf("%d", &matrix1[i][j]);
     }
  }
  printf("Enter elements of matrix2:\n");
  for (int i = 0; i < rows1; i++) {
     for (int j = 0; j < cols1; j++) {
       scanf("%d", &matrix2[i][j]);
     }
  }
  addMatrix(matrix1, matrix2, result, rows1, cols1);
  printf("Resultant matrix after addition:\n");
  for (int i = 0; i < rows1; i++) {
     for (int j = 0; j < cols1; j++) {
       printf("%d\t", result[i][j]);
     printf("\n");
  }
  break;
case 2:
  printf("Enter the number of rows and columns of the matrices: ");
  scanf("%d %d", &rows1, &cols1);
  printf("Enter elements of matrix1:\n");
  for (int i = 0; i < rows1; i++) {
     for (int j = 0; j < cols1; j++) {
       scanf("%d", &matrix1[i][j]);
     }
  }
  printf("Enter elements of matrix2:\n");
  for (int i = 0; i < rows1; i++) {
     for (int j = 0; j < cols1; j++) {
       scanf("%d", &matrix2[i][j]);
     }
  }
  subtractMatrix(matrix1, matrix2, result, rows1, cols1);
```

```
printf("Resultant matrix after subtraction:\n");
  for (int i = 0; i < rows1; i++) {
     for (int j = 0; j < cols1; j++) {
       printf("%d\t", result[i][j]);
     printf("\n");
  }
  break;
case 3:
  printf("Enter the number of rows and columns of matrix1: ");
  scanf("%d %d", &rows1, &cols1);
  printf("Enter the number of columns of matrix2: ");
  scanf("%d", &cols2);
  printf("Enter elements of matrix1:\n");
  for (int i = 0; i < rows1; i++) {
     for (int j = 0; j < cols1; j++) {
       scanf("%d", &matrix1[i][j]);
     }
  }
  printf("Enter elements of matrix2:\n");
  for (int i = 0; i < cols1; i++) {
     for (int j = 0; j < cols2; j++) {
       scanf("%d", &matrix2[i][j]);
     }
  }
  multiplyMatrix(matrix1, matrix2, result, rows1, cols1, cols2);
  printf("Resultant matrix after multiplication:\n");
  for (int i = 0; i < rows1; i++) {
     for (int j = 0; j < cols2; j++) {
       printf("%d\t", result[i][j]);
     printf("\n");
  }
  break;
case 4:
  printf("Enter the number of rows and columns of the matrix: ");
  scanf("%d %d", &rows1, &cols1);
  printf("Enter elements of the matrix:\n");
```

```
for (int i = 0; i < rows1; i++) {
     for (int i = 0; i < cols1; i++) {
       scanf("%d", &matrix1[i][j]);
     }
  }
  diagonalSum(matrix1, rows1);
  break;
case 5:
  printf("Enter the number of rows and columns of the matrix: ");
  scanf("%d %d", &rows1, &cols1);
  printf("Enter elements of the matrix:\n");
  for (int i = 0; i < rows1; i++) {
     for (int j = 0; j < cols1; j++) {
       scanf("%d", &matrix1[i][j]);
     }
  }
  rowColumnSum(matrix1, rows1, cols1);
  break;
case 6:
  printf("Enter the number of rows and columns of the matrix: ");
  scanf("%d %d", &rows1, &cols1);
  printf("Enter elements of the matrix:\n");
  for (int i = 0; i < rows1; i++) {
     for (int j = 0; j < cols1; j++) {
       scanf("%d", &matrix1[i][j]);
     }
  }
  printTranspose(matrix1, rows1, cols1);
  break;
case 7:
  printf("Enter the number of rows and columns of the matrix: ");
  scanf("%d %d", &rows1, &cols1);
  printf("Enter elements of the matrix:\n");
  for (int i = 0; i < rows1; i++) {
     for (int j = 0; j < cols1; j++) {
       scanf("%d", &matrix1[i][j]);
     }
```

```
if (isSymmetric(matrix1, rows1, cols1)) {
    printf("The matrix is symmetric.\n");
} else {
    printf("The matrix is not symmetric.\n");
}
break;

case 0:
    printf("Exiting the program. Goodbye!\n");
break;

default:
    printf("Invalid choice. Please enter a valid option.\n");
}

printf("\n");
} while (choice != 0);
return 0;
}
```

OBSERVATION BOOK:

```
Jab 1

20 AT to pass matures at farameters in fragram

i) Addition I subtraction

i) Multiplication

ii) Sum of disperal elements

iv) Sum of disperal elements

iv) Sum of disperal elements

iv) Sum of disperal elements

void add (int a [re](re), int b (re)(re), subs(ro)(re), sinty

for (int i=0, ics, i+1)

for (int i=0, ics, i+1)
```

roid multiply (unt alcolled), int b (colled), authorities]

for (unt i=0; icose, i+t)

multility] = 0; bcc; be+t)

multility] += a(i)(b) × b(b)(g);

3

void secon (unt alcolled), unt se unt c)

int secon, coun,

for (unt i=0; icos i+t)

for (unt i=0; icos i+t)

second second (unt alcolled), unt se unt c)

for (unt i=0; icos i+t)

second second (unt alcolled), unt se unt c)

transtrolled;

for (unt i=0; icos; i+t)

int issymmetric (unt also)(so), sixt in, int c)

if (six = c)

sultime 0

for (unt = 0, i est; i et)

if (all [g]! = alg [c])

sultime 0;

g

sultime 1;

int also [so], b[so](so], rest [o][so], si, c, sel, cl, ch,

do

fruitf ("I Addition in Sultraction in S. Hultplink

In 4. Pow col sum in 5. Transpose

In 6. Symmetric)

fountf ("Enter your choice");

scarf ("I.d", dch);

switch (ch)

case 1: fruitf ("Enter views discoloures of first in

scarf ("I.d", dch);

fountf ("Enter alements);

for last i = 0; ic as, it)

Invinit i = 0; ic as, it)

add (a, b, vies, sel, cl).

for list ("Resultant matrice").

for list i=0, iea, iee)

for (ant j=0, jec, jet)

frish ("Id", &alid(jd)),

scant ("Id" Id", &alid(jd)),

frish ("Id" alments").

for (j=0; jec, j+t)

scant ("Id", &a(id(j)),

grant ("Id", &a(id(j)),

frish ("Edex elements of second matrix").

scant ("Id", &a(id(jd)),

grant ("Id", &a(id(jd)),

grant ("Id", &a(id(jd)),

grant ("Id", &a(id(jd)),

for (int j=0; jecl, jet)

scant ("Id", &blid(jd)),

diff (o,b, vies, or,c).

for (int j=0; jec, jet)

case 3. friently ("Extent elements of first malrix"),

for (unt i = 0; i cos; i+t)

for (unt i = 0; i cos; i+t)

for (int i = 0; i cos; i+t)

for (int i = 0; i cos; i+t)

for (int i = 0; i cos; i+t)

for (unt i = 0; i cos; i+t)

fruit ("Resultant matrix");

for (int i=0, i zm; i+1)

for (int j=0, jzc; jt+1)

for (int j=0, izc, j+1)

for (int j=0, jzc, j+1)

granf ("1.d", &a[i][ij]);

3

trangla, v,c);

for (int i=0, izz, i+1)

Joseph ("bymonthue");

else

fruitf ("bymonthue");

else

fruitf ("Not symmetrie");

default fruitf ("Wong choice !!!")

Joseph (ch = 0)

Jenus

Later size of matrices & Sutheration

Later size of matrices & Sutheration

Later elements of fruit nature 4 Diagonal sum

Result at matrix

Result at matrix

Enter elements of second matrix 6 Inampor

Result at matrix

Later your choice

Later elements of fruit matrix 5 6 7 8

Later elements of fruit matrix

Later size of matrices

Later size of size of

```
Enter your choice
    Enter says of matrices
    Enter elements of first matrix 1 4 68
Enter elements of second matrix 3 5 80
Enter elements of second matrix 21 14
Enter eleme Resultant matrix 21 14
    Enter your choice
   Enter size of matrix
    Enter the elements 1 5 12 6
   Sum 7
  Enter your choice
  Enter size of matrix
  Enler elements 1 5 2 8
Row | sum 6
 Colourn I sum 3
Row 2 sum 10
Colourn 2 sum 18
                                                Enter your chois
Enter your choice
Enter size of matrix
                                              2 2
Enler elements 15
Lymnetric
2 2
Enter selements
Transpose matrice 1 7
```

OUTPUT:

```
Matrix Operations:

1. Addition

2. Subtraction

3. Multiplication

4. Sum of Diagonals

5. Sum of Rows and Columns

6. Transpose

7. Check Symmetry

9. Exit
Enter your choice:

8. Matrix Operations:

1. Addition

2. Subtraction

6. Transpose

7. Check Symmetry

9. Exit
Enter the number of rows and columns of the matrix:

2. 2 Enter the number of the matrix:

1. 3. 4. 6

5. Sum of principal diagonal:

7. Sum of principal diagonal:

7. Sum of principal diagonal:

7. Sum of principal diagonal:

8. Sum of Rows and Columns

1. Addition

2. Subtraction

3. Multiplication

4. Sum of Diagonals

5. Sum of Rows and Columns

6. Transpose

7. Check Symmetry

8. Exit
Enter your choice:

5. Enter the number of rows and columns of the matrix:

7. 9. 1

5. Sum of elements in Row 1: 12

5. Sum of elements in Row 2: 1

5. Sum of elements in Column 1: 5

5. Sum of elements in Column 2: 8

Matrix Operations:

1. Addition

2. Subtraction

3. Multiplication

4. Sum of Diagonals

5. Sum of Rows and Columns

6. Transpose

7. Check Symmetry

9. Exit
Enter your choice: 6
Enter the number of rows and columns of the matrix: 2 2

Enter elements of the matrix:

4. 6. 9

1. Matrix Operations:

1. Addition

2. Subtraction

3. Multiplication

4. Sum of Diagonals

5. Sum of Rows and Columns

6. Transpose

7. Check Symmetry

8. Exit
Enter your choice: 7

Enter the number of rows and columns of the matrix: 2 2

Enter elements of the matrix: 4

8. Sum of Rows and Columns

6. Transpose

7. Check Symmetry

8. Exit
Enter your choice: 7

Enter the number of rows and columns of the matrix: 2 2

Enter elements of the matrix: 5

Enter the number of rows and columns of the matrix: 2 2

Enter elements of the matrix: 4

Enter your choice: 7

Enter the number of rows and columns of the matrix: 2 2

Enter elements of the matrix: 4

Enter the number of rows and columns of the matrix: 5

Enter the number of rows and columns of the matrix: 5

Enter the number of rows and columns of the matrix: 5

Enter the number of rows and columns of the matrix: 5

Enter the
```

```
C:\Users\Admin\Desktop\18M21CS065\matrix\bin\Debug\matrix.exe

3. Multiplication
1. Sum of Diagonals
2. Sum of Rows and Columns
3. Transpose
7. Check Symmetry
9. Exit
1. Inter your choice: 7
1. Inter the number of rows and columns of the matrix: 2 2
1. Inter elements of the matrix:
1. 3 8 9
1. Multiplication
1. Addition
2. Subtraction
3. Multiplication
3. Sum of Diagonals
5. Sum of Rows and Columns
5. Transpose
7. Check Symmetry
9. Exit
1. At the number of rows and columns of matrix: 2 2
1. Inter the number of rows and columns of matrix: 2 2
1. Tere the number of rows and columns of matrix: 2 2
1. Tere elements of matrix: 1
1. 4 6 8
1. Sum of Diagonals
1. 4 6 8
1. Sum of Diagonals
1. Addition
2. Subtraction
3. Multiplication
4. Sum of Diagonals
5. Sum of Rows and Columns
5. Transpose
7. Check Symmetry
9. Exit
1. Sum of Diagonals
5. Sum of Rows and Columns
6. Transpose
7. Check Symmetry
9. Exit
1. Exit
1. Term your choice: Exiting the program. Goodbye!
1. Process returned 0 (0x0) execution time: 69.229 s
1. Press any key to continue.
```