matrixAddition.c

```c
#include <stdio.h>

// Function to add two matrices of same dimensions

int main(){

    // Declaring two 2x3 matrices and a result matrix
    int a[2][3], b[2][3], sum[2][3], i,j;

    // Taking input for first matrix
    printf("Enter elements of first matrix:\n");
    for(i=0; i<2; i++){
        for(j=0; j<3; j++){
            printf("Element [%d][%d]: ", i, j);
            scanf("%d", &a[i][j]);
        }
    }

    // Taking input for second matrix
    printf("Enter elements of second matrix:\n");
    for(i=0; i<2; i++){
        for(j=0; j<3; j++){
            printf("Element [%d][%d]: ", i, j);
            scanf("%d", &b[i][j]);
        }
    }

    // Displaying the first matrix
    printf("First matrix:\n");
    for(i=0; i<2; i++){
        for(j=0; j<3; j++){
            printf("%d ", a[i][j]);
        }
        printf("\n");
    }

    // Displaying the second matrix
    printf("Second matrix:\n");
    for(i=0; i<2; i++){
        for(j=0; j<3; j++){
            printf("%d ", b[i][j]);
        }
        printf("\n");
    }

    // Adding the two matrices
    for(i=0; i<2; i++){
        for(j=0; j<3; j++){
            sum[i][j] = a[i][j] + b[i][j];
        }
    }
```

```c

    // Displaying the sum
    printf("Sum of the two matrices:\n");
    for(i=0; i<2; i++){
        for(j=0; j<3; j++){
            printf("%d ", sum[i][j]);
        }
        printf("\n");
    }

    return 0;
}


/**
 * *********************************
 * Example Input/Output:
 * *********************************
Enter elements of first matrix:
Element [0][0]: 2
Element [0][1]: 1
Element [0][2]: 3
Element [1][0]: 4
Element [1][1]: 5
Element [1][2]: 6
Enter elements of second matrix:
Element [0][0]: 6
Element [0][1]: 1
Element [0][2]: 9
Element [1][0]: 5
Element [1][1]: 4
Element [1][2]: 5
First matrix:
2 1 3
4 5 6
Second matrix:
6 1 9
5 4 5
Sum of the two matrices:
8 2 12
9 9 11
************************************/
```

matrixAddition2.c

```c
#include <stdio.h>
int main() {
  int r, c, a[100][100], b[100][100], sum[100][100], i, j;
  printf("Enter the number of rows (between 1 and 100): ");
  scanf("%d", &r);
  printf("Enter the number of columns (between 1 and 100): ");
  scanf("%d", &c);

  printf("\nEnter elements of 1st matrix:\n");
  for (i = 0; i < r; ++i)
    for (j = 0; j < c; ++j) {
      printf("Enter element a%d%d: ", i + 1, j + 1);
      scanf("%d", &a[i][j]);
    }

  printf("Enter elements of 2nd matrix:\n");
  for (i = 0; i < r; ++i)
    for (j = 0; j < c; ++j) {
      printf("Enter element b%d%d: ", i + 1, j + 1);
      scanf("%d", &b[i][j]);
    }

  // adding two matrices
  for (i = 0; i < r; ++i)
    for (j = 0; j < c; ++j) {
      sum[i][j] = a[i][j] + b[i][j];
    }

  // printing the result
  printf("\nSum of two matrices: \n");
  for (i = 0; i < r; ++i)
    for (j = 0; j < c; ++j) {
      printf("%d   ", sum[i][j]);
      if (j == c - 1) {
        printf("\n\n");
      }
    }

  return 0;
}

```

matrixSubstraction.c

```c
#include <stdio.h>

int main() {
    int rows, cols;

    // Get dimensions from the user
    printf("Enter the number of rows: ");
    scanf("%d", &rows);
    printf("Enter the number of columns: ");
    scanf("%d", &cols);

    // Declare matrices
    int matrixA[rows][cols];
    int matrixB[rows][cols];
    int resultMatrix[rows][cols];

    // Input elements for matrix A
    printf("\nEnter elements for Matrix A:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("Enter element A[%d][%d]: ", i, j);
            scanf("%d", &matrixA[i][j]);
        }
    }

    // Input elements for matrix B
    printf("\nEnter elements for Matrix B:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("Enter element B[%d][%d]: ", i, j);
            scanf("%d", &matrixB[i][j]);
        }
    }

    // display matrix A
    printf("\nMatrix A:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("%d\t", matrixA[i][j]);
        }
        printf("\n");
    }

    // display matrix B
    printf("\nMatrix B:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("%d\t", matrixB[i][j]);
        }
        printf("\n");
    }
```

```c

    // Perform matrix subtraction (A - B)
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            resultMatrix[i][j] = matrixA[i][j] - matrixB[i][j];
        }
    }

    // Print the result matrix
    printf("\nResult of Matrix A - Matrix B:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("%d\t", resultMatrix[i][j]);
        }
        printf("\n");
    }

    return 0;
}

/**
 * Example Input/Output:
 * Enter the number of rows: 2
Enter the number of columns: 2

Enter elements for Matrix A:
Enter element A[0][0]: 1
Enter element A[0][1]: 4
Enter element A[1][0]: 5
Enter element A[1][1]: 6

Enter elements for Matrix B:
Enter element B[0][0]: 2
Enter element B[0][1]: 3
Enter element B[1][0]: 1
Enter element B[1][1]: 6

Matrix A:
1       4
5       6

Matrix B:
2       3
1       6

Result of Matrix A - Matrix B:
-1      1
4       0
 */
```

multiplicationMatrix.c

```c
#include <stdio.h>

int main()
{
    int A[50][50], B[50][50], C[50][50], i, j, m1, n1, m2, n2;

    printf("\n\nMultiplication of two Matrices :\n");
    printf("------------------------------\n");
    // for A matrix
    printf("\n\n A Matrix :\n");
    printf("------------------------------\n");
    printf("Enter the number of rows of A matrix(between 1 to 50 ): ");
    scanf("%d", &m1);
    printf("Enter the number of columns of A matrix (between 1 to 50 ): ");
    scanf("%d", &n1);

    // for B matrix
    printf("\n\n B Matrix :\n");
    printf("------------------------------\n");
    printf("Enter the number of rows of B matrix(between 1 to 50 ): ");
    scanf("%d", &m2);
    printf("Enter the number of columns of B matrix (between 1 to 50 ): ");
    scanf("%d", &n2);

    if (n1 != m2)
    {
        printf("\n Error : Number of column of  the A matrix should be same as number of rows of B matrix. \n\n");
    }
    else
    {

        /* Stored values into the array*/
        printf("\n\n");

        printf("Enter elements of  A matrix :\n");
        printf("------------------------------\n");

        for (i = 0; i < m1; i++) //  row
        {
            for (j = 0; j < n1; j++) // column
            {
                printf("element - [%d],[%d] : ", i, j);
                scanf("%d", &A[i][j]);
            }
        }

        printf("\n\n");

        printf("Enter elements of  B matrix :\n");
        printf("------------------------------\n");

```

```c
        for (i = 0; i < m2; i++)
        {
            for (j = 0; j < n2; j++)
            {
                printf("element - [%d],[%d] : ", i, j);
                scanf("%d", &B[i][j]);
            }
        }
        printf("\n A matrix is :\n");
        printf("-----------------------------\n");

        for (i = 0; i < m1; i++)
        {
            printf("\n\t");
            for (j = 0; j < n1; j++)
                printf("%d\t", A[i][j]);
        }

        printf("\n\n");

        printf("\n B matrix is :\n");
        printf("-----------------------------\n");

        for (i = 0; i < m2; i++)
        {
            printf("\n\t");
            for (j = 0; j < n2; j++)
                printf("%d\t", B[i][j]);
        }
        /* calculate the multiplication of the matrix */

        printf("\n\n");

        for (i = 0; i < m1; i++)
        {
            for (j = 0; j < n2; j++)
            {
                // Calculate the result
                for (int k = 0; k < n1; k++)
                {
                    C[i][j] += A[i][k] * B[k][j];
                }
            }
        }

        // output
        printf("\nThe C matrix is : \n");
        printf("-----------------------------\n");

        for (i = 0; i < m1; i++)
        {
            printf("\n\t");
            for (j = 0; j < n2; j++)
                printf("%d\t", C[i][j]);
```

```
        }

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


/*
Output:
Multiplication of two Matrices :
------------------------------


 A Matrix :
------------------------------
Enter the number of rows of A matrix(between 1 to 50 ): 2
Enter the number of columns of A matrix (between 1 to 50 ): 3


 B Matrix :
------------------------------
Enter the number of rows of B matrix(between 1 to 50 ): 3
Enter the number of columns of B matrix (between 1 to 50 ): 2


Enter elements of  A matrix :
------------------------------
element - [0],[0] : 1
element - [0],[1] : 3
element - [0],[2] : 4
element - [1],[0] : 2
element - [1],[1] : 1
element - [1],[2] : 6


Enter elements of  B matrix :
------------------------------
element - [0],[0] : 4
element - [0],[1] : 6
element - [1],[0] : 8
element - [1],[1] : 7
element - [2],[0] : 9
element - [2],[1] : 4

 A matrix is :
------------------------------

        1       3       4
        2       1       6


 B matrix is :
------------------------------

```

```
        4       6
        8       7
        9       4


The C matrix is :
------------------------------

        64      43
        70      43
*/
```

transposeMatrix.c

```c
#include <stdio.h>

int main(){

    int rows, cols;
    // Get dimensions from the user
    printf("Enter the number of rows: ");
    scanf("%d", &rows);
    printf("Enter the number of columns: ");
    scanf("%d", &cols);

    // Declare matrix
    int matrix[rows][cols];
    int transposedMatrix[cols][rows];

    // Get matrix elements from the user
    printf("Enter the elements of the matrix:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("Element [%d][%d]: ", i, j);
            scanf("%d", &matrix[i][j]);
        }
    }
    // print original matrix
    printf("Original matrix:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("%d ", matrix[i][j]);
        }
        printf("\n");
    }

    // Transpose the matrix
    for (int i = 0; i < cols; i++) {
        for (int j = 0; j < rows; j++) {
            transposedMatrix[i][j] = matrix[j][i];
        }
    }

    // Display the transposed matrix
    printf("Transposed matrix:\n");
    for (int i = 0; i < cols; i++) {
        for (int j = 0; j < rows; j++) {
            printf("%d ", transposedMatrix[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```

transposeSquareMatrix.c

```c
#include <stdio.h>

int main(){
    int rows, cols;
    // Get dimensions from the user
    printf("Enter the number of rows/cols: ");
    scanf("%d", &rows);
    cols = rows;

    // Declare matrix
    int matrix[rows][cols];
    int transposedMatrix[cols][rows];

    // Get matrix elements from the user
    printf("Enter the elements of the matrix:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("Element [%d][%d]: ", i, j);
            scanf("%d", &matrix[i][j]);
        }
    }
    // print original matrix
    printf("Original matrix:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("%d ", matrix[i][j]);
        }
        printf("\n");
    }

    // Transpose of square matrix
    for (int i = 0; i < rows; i++) {
        for (int j = i; j < cols; j++) {
            int temp = matrix[i][j];
            matrix[i][j] = matrix[j][i];
            matrix[j][i] = temp;
        }
    }

    // Display the transposed matrix
    printf("Transposed matrix:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("%d ", matrix[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```

sumPrime2.c

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

// function to check if a number is prime
int isPrime(int n) {
    if (n <= 1) {
        return 0; // not prime
    }
    if (n == 2) {
        return 1; // 2 is prime
    }
    if (n % 2 == 0) {
        return 0; // even numbers (except 2) are not prime
    }
    for (int i = 3; i <= sqrt(n); i += 2) {
        if (n % i == 0) {
            return 0; // found a divisor, not prime
        }
    }
    return 1; // prime
}

int main() {
    int num, sum = 0, count = 0;

    printf("Enter a positive integer: ");
    scanf("%d", &num);

    if (num < 2) {
        printf("No prime numbers exist below %d.\n", num);
        return 0;
    }

    printf("Prime numbers between 1 and %d:\n", num);

    for (int i = 2; i <= num; i++) {
        if (isPrime(i)) {
            printf("%d ", i);
            sum += i;
            count++;
        }
    }

    printf("\n\nTotal prime numbers found: %d\n", count);
    printf("Sum of all prime numbers: %d\n", sum);

    return 0;
}
```

checkPrime2.c

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

// function to check if a number is prime
int isPrime(int n) {
    if (n <= 1) {
        return 0; // not prime
    }
    if (n == 2) {
        return 1; // 2 is prime
    }
    if (n % 2 == 0) {
        return 0; // even numbers (except 2) are not prime
    }
    for (int i = 3; i <= sqrt(n); i += 2) {
        if (n % i == 0) {
            return 0; // found a divisor, not prime
        }
    }
    return 1; // prime
}

int main() {
    int num;

    printf("Enter a positive integer: ");
    scanf("%d", &num);

    if (num < 2) {
        printf("No prime numbers exist below %d.\n", num);
        return 0;
    }

    // Check if the number is prime
    if (isPrime(num)) {
        printf("%d is a prime number.\n", num);
    } else {
        printf("%d is not a prime number.\n", num);
    }


    return 0;
}
```