1. **Write a program to store marks for n number of student in an array and print their marks.**

Code :-

```
#include <stdio.h>
int main() {
   int n;

   // Input the number of students
   printf("Enter the number of students: ");
   scanf("%d", &n);

   // Create an array to store marks
   float marks[n];

   // Input marks for each student
   for (int i = 0; i < n; i++) {
      printf("Enter marks for student %d: ", i + 1);
      scanf("%f", &marks[i]);
   }

   // Print the marks
   printf("\nMarks of students:\n");
   for (int i = 0; i < n; i++) {
      printf("Student %d: %.2f\n", i + 1, marks[i]);
   }
   return 0;
}
```

OUTPUT :-

```
Enter the number of students: 4
Enter marks for student 1: 89
Enter marks for student 2: 78
Enter marks for student 3: 85
Enter marks for student 4: 88
Marks of students:
Student 1: 89.00
Student 2: 78.00
Student 3: 85.00
Student 4: 88.00
```

2. **Write a program that stores the marks of the subject Mathematics and English of n number of students in an array and then prints their total marks.**

Code :-

```
#include <stdio.h>
int main() {
   int n;

   // Input the number of students
   printf("Enter the number of students: ");
   scanf("%d", &n);

   // Create arrays to store marks
   float mathMarks[n];
   float englishMarks[n];
   float totalMarks[n];

   // Input marks for each student
   for (int i = 0; i < n; i++) {
      printf("Enter Mathematics marks for student %d: ", i + 1);
      scanf("%f", &mathMarks[i]);
      printf("Enter English marks for student %d: ", i + 1);
      scanf("%f", &englishMarks[i]);

      // Calculate total marks
      totalMarks[i] = mathMarks[i] + englishMarks[i];
   }

   // Print total marks for each student
   printf("\nTotal marks of students:\n");
   for (int i = 0; i < n; i++) {
      printf("Student %d: Total Marks = %.2f\n", i + 1, totalMarks[i]);
   }

   return 0;
}
```

OUTPUT :-

```
Enter the number of students: 5
Enter Mathematics marks for student 1: 70
Enter English marks for student 1: 80
Enter Mathematics marks for student 2: 85
Enter English marks for student 2: 95
Enter Mathematics marks for student 3: 90
Enter English marks for student 3: 78
Enter Mathematics marks for student 4: 76
Enter English marks for student 4: 75
Enter Mathematics marks for student 5: 79
Enter English marks for student 5: 74

Total marks of students:
Student 1: Total Marks = 150.00
Student 2: Total Marks = 180.00
Student 3: Total Marks = 168.00
Student 4: Total Marks = 151.00
Student 5: Total Marks = 153.00
```

3. Write a program to insert an element in an array in a particular position.

Code :-

```
#include <stdio.h>

#define MAX_SIZE 100 // Define the maximum size of the array

int main() {
   int arr[MAX_SIZE], n, i, pos, value;

   // Input the number of elements in the array
   printf("Enter the number of elements in the array (max %d): ", MAX_SIZE);
   scanf("%d", &n);

   // Input the elements of the array
   printf("Enter %d elements:\n", n);
   for (i = 0; i < n; i++) {
      scanf("%d", &arr[i]);
   }

   // Input the position and the value to be inserted
   printf("Enter the position to insert the element (0 to %d): ", n);
   scanf("%d", &pos);
   printf("Enter the value to insert: ");
   scanf("%d", &value);

   // Check if the position is valid
   if (pos < 0 || pos > n) {
      printf("Invalid position! Please enter a position between 0 and %d.\n", n);
      return 1;
   }

   // Shift elements to the right to make space for the new element
   for (i = n; i > pos; i--) {
      arr[i] = arr[i - 1];
   }

   // Insert the new element
   arr[pos] = value;

   // Update the size of the array
   n++;
```

```
    // Print the updated array
    printf("\nUpdated array:\n");
    for (i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");

    return 0;
}
```

OUTPUT :-

```
Enter the number of elements in the array (max 100): 4
Enter 4 elements:
3
4
5
6
Enter the position to insert the element (0 to 4): 2
Enter the value to insert: 8

Updated array:
3 4 8 5 6
```

4. Write a program to delete an element from a particular position of an array.

Code :-

```
#include <stdio.h>

#define MAX_SIZE 100 // Define the maximum size of the array

int main() {
   int arr[MAX_SIZE], n, i, pos;

   // Input the number of elements in the array
   printf("Enter the number of elements in the array (max %d): ", MAX_SIZE);
   scanf("%d", &n);

   // Input the elements of the array
   printf("Enter %d elements:\n", n);
   for (i = 0; i < n; i++) {
      scanf("%d", &arr[i]);
   }

   // Input the position of the element to be deleted
   printf("Enter the position of the element to delete (0 to %d): ", n - 1);
   scanf("%d", &pos);

   // Check if the position is valid
   if (pos < 0 || pos >= n) {
      printf("Invalid position! Please enter a position between 0 and %d.\n", n - 1);
      return 1;
   }

   // Shift elements to the left to fill the gap
   for (i = pos; i < n - 1; i++) {
      arr[i] = arr[i + 1];
   }

   // Update the size of the array
   n--;

   // Print the updated array
   printf("\nUpdated array:\n");
   for (i = 0; i < n; i++) {
```

```
        printf("%d ", arr[i]);
    }
    printf("\n");

    return 0;
}
```

OUTPUT :-

```
Enter the number of elements in the array (max 100): 4
Enter 4 elements:
5
6
7
8
Enter the position of the element to delete (0 to 3): 2

Updated array:
5 6 8
```

5. Write a program to convert a decimal number taken as input from a user to the corresponding binary number and store the result in an array.

Code :-

```
#include <stdio.h>

#define MAX_BITS 32 // Maximum bits for storing binary representation

int main() {
   int decimal, i = 0;
   int binary[MAX_BITS]; // Array to store binary digits

   // Input the decimal number
   printf("Enter a decimal number: ");
   scanf("%d", &decimal);

   // Convert decimal to binary
   while (decimal > 0) {
      binary[i] = decimal % 2; // Store the remainder (0 or 1)
      decimal = decimal / 2;    // Divide the number by 2
      i++;
   }

   // Print the binary representation
   printf("Binary representation: ");
   for (int j = i - 1; j >= 0; j--) { // Print in reverse order
      printf("%d", binary[j]);
   }
   printf("\n");

   return 0;
}
```

OUTPUT :-

```
Enter a decimal number: 34
Binary representation: 100010
```

6. Write a program to input a binary number in an array and convert it into a corresponding decimal number.

Code :-

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

int binaryToDecimal(char binary[]) {
   int decimal = 0;
   int length = strlen(binary);

   for (int i = 0; i < length; i++) {
      // Convert character '0' or '1' to integer 0 or 1
      if (binary[length - 1 - i] == '1') {
         decimal += pow(2, i);
      }
   }

   return decimal;
}

int main() {
   char binary[65];  // Array to hold a binary number (up to 64 bits + null terminator)

   printf("Enter a binary number: ");
   scanf("%64s", binary);  // Read a string input safely

   // Validate input: Check if it contains only '0's and '1's
   for (int i = 0; binary[i] != '\0'; i++) {
      if (binary[i] != '0' && binary[i] != '1') {
         printf("Invalid binary number!\n");
         return 1;
      }
   }
   int decimal = binaryToDecimal(binary);
   printf("Decimal equivalent: %d\n", decimal);
   return 0;
}
```

OUTPUT :-

```
Enter a binary number: 11001
Decimal equivalent: 25
```

7. Write a program to find the smallest and the largest elements in an array.

Code :-

```
#include <stdio.h>
int main() {
   int n;

   // Prompt user for the number of elements in the array
   printf("Enter the number of elements in the array: ");
   scanf("%d", &n);

   // Declare the array
   int arr[n];

   // Input the elements of the array
   printf("Enter %d elements: ", n);
   for (int i = 0; i < n; i++) {
      scanf("%d", &arr[i]);
   }

   // Initialize smallest and largest with the first element
   int smallest = arr[0];
   int largest = arr[0];

   // Traverse the array to find smallest and largest
   for (int i = 1; i < n; i++) {
      if (arr[i] < smallest) {
         smallest = arr[i];
      }
      if (arr[i] > largest) {
         largest = arr[i];
      }
   }

   // Output the results
   printf("Smallest element: %d\n", smallest);
   printf("Largest element: %d\n", largest);

   return 0;
}
```

OUTPUT :-

```
Enter the number of elements in the array: 5
Enter 5 elements: 12
23
24
35
10
Smallest element: 10
Largest element: 35
```

8. Write a program for deleting duplicate elements in an array.

Code :-

```
#include <stdio.h>
void deleteDuplicates(int arr[], int *n) {
   int temp[*n]; // Temporary array to hold unique elements
   int j = 0; // Index for temp array

   for (int i = 0; i < *n; i++) {
      int isDuplicate = 0;

      // Check if arr[i] is already in temp
      for (int k = 0; k < j; k++) {
         if (arr[i] == temp[k]) {
            isDuplicate = 1;
            break;
         }
      }

      // If it's not a duplicate, add it to temp
      if (!isDuplicate) {
         temp[j] = arr[i];
         j++;
      }
   }

   // Copy unique elements back to original array
   for (int i = 0; i < j; i++) {
      arr[i] = temp[i];
   }

   // Update the size of the original array
   *n = j;
}

int main() {
   int n;

   // Prompt user for the number of elements in the array
   printf("Enter the number of elements in the array: ");
   scanf("%d", &n);
```

```
    // Declare the array
    int arr[n];

    // Input the elements of the array
    printf("Enter %d elements: ", n);
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

    // Call the function to delete duplicates
    deleteDuplicates(arr, &n);

    // Output the unique elements
    printf("Array after deleting duplicates: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");

    return 0;
}
```

OUTPUT :-

```
Enter the number of elements in the array: 5
Enter 5 elements: 10
55
45
55
35
Array after deleting duplicates: 10 55 45 35
```

9. Write a program to search for a particular element in an array.

Code :-

```
#include <stdio.h>

int main() {
   int n, search, found = 0;

   // Prompt user for the number of elements in the array
   printf("Enter the number of elements in the array: ");
   scanf("%d", &n);

   // Declare the array
   int arr[n];

   // Input the elements of the array
   printf("Enter %d elements: ", n);
   for (int i = 0; i < n; i++) {
      scanf("%d", &arr[i]);
   }

   // Prompt user for the element to search
   printf("Enter the element to search for: ");
   scanf("%d", &search);

   // Search for the element in the array
   for (int i = 0; i < n; i++) {
      if (arr[i] == search) {
         printf("Element %d found at index %d.\n", search, i);
         found = 1;
         break; // Exit loop if the element is found
      }
   }

   if (!found) {
      printf("Element %d not found in the array.\n", search);
   }

   return 0;
}
```

OUTPUT :-

```
Enter the number of elements in the array: 4
Enter 4 elements: 3
5
7
9
Enter the element to search for: 7
Element 7 found at index 2.
```

10.Write a program to sort n elements (ascending order).

Code :-

```
#include <stdio.h>

void bubbleSort(int arr[], int n) {
   for (int i = 0; i < n - 1; i++) {
      for (int j = 0; j < n - 1 - i; j++) {
         if (arr[j] > arr[j + 1]) {
            // Swap arr[j] and arr[j + 1]
            int temp = arr[j];
            arr[j] = arr[j + 1];
            arr[j + 1] = temp;
         }
      }
   }
}

int main() {
   int n;

   // Prompt user for the number of elements in the array
   printf("Enter the number of elements: ");
   scanf("%d", &n);

   // Declare the array
   int arr[n];

   // Input the elements of the array
   printf("Enter %d elements: ", n);
   for (int i = 0; i < n; i++) {
      scanf("%d", &arr[i]);
   }

   // Sort the array
   bubbleSort(arr, n);

   // Output the sorted array
   printf("Sorted array in ascending order: ");
   for (int i = 0; i < n; i++) {
      printf("%d ", arr[i]);
   }
   printf("\n");
```

```
    return 0;
}
```

OUTPUT :-

```
Enter the number of elements: 5
Enter 5 elements: 3
1
8
2
9
Sorted array in ascending order: 1 2 3 8 9
```

11. Write a program to find second second-highest number from the array without using sorting.

Code :-

```
#include <stdio.h>

int main() {
   int n;

   // Prompt user for the number of elements in the array
   printf("Enter the number of elements in the array: ");
   scanf("%d", &n);

   // Check if there are enough elements
   if (n < 2) {
      printf("Array must contain at least two elements.\n");
      return 1;
   }

   // Declare the array
   int arr[n];

   // Input the elements of the array
   printf("Enter %d elements: ", n);
   for (int i = 0; i < n; i++) {
      scanf("%d", &arr[i]);
   }

   // Initialize the first and second highest
   int highest = arr[0];
   int second_highest = -1; // Using -1 to indicate that it may not be set

   // Find the highest and second highest
   for (int i = 1; i < n; i++) {
      if (arr[i] > highest) {
         second_highest = highest; // Update second highest
         highest = arr[i]; // Update highest
      } else if (arr[i] > second_highest && arr[i] != highest) {
         second_highest = arr[i]; // Update second highest
      }
   }

   // Output the result
```

```
    if (second_highest == -1) {
        printf("There is no second highest number (all elements might be the same).\n");
    } else {
        printf("The second highest number is: %d\n", second_highest);
    }

    return 0;
}
```

OUTPUT :-

```
Enter the number of elements in the array: 5
Enter 5 elements: 23
34
45
56
71
The second highest number is: 56
```

12. Write a program to perform addition and subtraction between two matrices.

Code :-

```
#include <stdio.h>

#define MAX_SIZE 10 // Define a maximum size for the matrices

void inputMatrix(int matrix[MAX_SIZE][MAX_SIZE], int rows, int cols) {
   printf("Enter the elements of the matrix (%dx%d):\n", rows, cols);
   for (int i = 0; i < rows; i++) {
      for (int j = 0; j < cols; j++) {
         scanf("%d", &matrix[i][j]);
      }
   }
}

void printMatrix(int matrix[MAX_SIZE][MAX_SIZE], int rows, int cols) {
   printf("Matrix (%dx%d):\n", rows, cols);
   for (int i = 0; i < rows; i++) {
      for (int j = 0; j < cols; j++) {
         printf("%d ", matrix[i][j]);
      }
      printf("\n");
   }
}

void addMatrices(int A[MAX_SIZE][MAX_SIZE], int B[MAX_SIZE][MAX_SIZE], int
C[MAX_SIZE][MAX_SIZE], int rows, int cols) {
   for (int i = 0; i < rows; i++) {
      for (int j = 0; j < cols; j++) {
         C[i][j] = A[i][j] + B[i][j];
      }
   }
}

void subtractMatrices(int A[MAX_SIZE][MAX_SIZE], int B[MAX_SIZE][MAX_SIZE], int
C[MAX_SIZE][MAX_SIZE], int rows, int cols) {
   for (int i = 0; i < rows; i++) {
      for (int j = 0; j < cols; j++) {
         C[i][j] = A[i][j] - B[i][j];
      }
   }
```

```
}

int main() {
    int A[MAX_SIZE][MAX_SIZE], B[MAX_SIZE][MAX_SIZE], sum[MAX_SIZE][MAX_SIZE],
difference[MAX_SIZE][MAX_SIZE];
    int rows, cols;

    // Input the dimensions of the matrices
    printf("Enter the number of rows and columns for the matrices: ");
    scanf("%d %d", &rows, &cols);

    // Input the matrices
    printf("Matrix A:\n");
    inputMatrix(A, rows, cols);

    printf("Matrix B:\n");
    inputMatrix(B, rows, cols);

    // Perform addition and subtraction
    addMatrices(A, B, sum, rows, cols);
    subtractMatrices(A, B, difference, rows, cols);

    // Output the results
    printMatrix(A, rows, cols);
    printMatrix(B, rows, cols);
    printMatrix(sum, rows, cols);
    printMatrix(difference, rows, cols);

    return 0;
}
```

OUTPUT :-

```
Enter the number of rows and columns for the matrices: 2
2
Matrix A:
Enter the elements of the matrix (2x2):
2
3
4
5
Matrix B:
Enter the elements of the matrix (2x2):
2
2
3
4
Matrix (2x2):
2 3
4 5
Matrix (2x2):
2 2
3 4
Matrix (2x2):
4 5
7 9
Matrix (2x2):
0 1
1 1
```

13.Write a program to transpose a matrix.

Code :-

```
#include <stdio.h>

#define MAX_SIZE 10 // Define a maximum size for the matrix

void inputMatrix(int matrix[MAX_SIZE][MAX_SIZE], int rows, int cols) {
   printf("Enter the elements of the matrix (%dx%d):\n", rows, cols);
   for (int i = 0; i < rows; i++) {
      for (int j = 0; j < cols; j++) {
         scanf("%d", &matrix[i][j]);
      }
   }
}

void printMatrix(int matrix[MAX_SIZE][MAX_SIZE], int rows, int cols) {
   printf("Matrix (%dx%d):\n", rows, cols);
   for (int i = 0; i < rows; i++) {
      for (int j = 0; j < cols; j++) {
         printf("%d ", matrix[i][j]);
      }
      printf("\n");
   }
}

void transposeMatrix(int matrix[MAX_SIZE][MAX_SIZE], int
transposed[MAX_SIZE][MAX_SIZE], int rows, int cols) {
   for (int i = 0; i < rows; i++) {
      for (int j = 0; j < cols; j++) {
         transposed[j][i] = matrix[i][j];
      }
   }
}

int main() {
   int matrix[MAX_SIZE][MAX_SIZE], transposed[MAX_SIZE][MAX_SIZE];
   int rows, cols;

   // Input the dimensions of the matrix
   printf("Enter the number of rows and columns for the matrix: ");
   scanf("%d %d", &rows, &cols);
```

```
    // Input the matrix
    inputMatrix(matrix, rows, cols);

    // Transpose the matrix
    transposeMatrix(matrix, transposed, rows, cols);

    // Output the original and transposed matrices
    printMatrix(matrix, rows, cols);
    printMatrix(transposed, cols, rows); // Note the rows and cols are swapped

    return 0;
}
```

OUTPUT :-

```
Enter the number of rows and columns for the matrix: 2
3
Enter the elements of the matrix (2x3):
2
3
4
5
6
7
Matrix (2x3):
2 3 4
5 6 7
Matrix (3x2):
2 5
3 6
4 7
```

14.Write a program to add the elements of each row and each column of a matrix.

Code :-

```
#include <stdio.h>

#define MAX_SIZE 10 // Define a maximum size for the matrix

void inputMatrix(int matrix[MAX_SIZE][MAX_SIZE], int rows, int cols) {
   printf("Enter the elements of the matrix (%dx%d):\n", rows, cols);
   for (int i = 0; i < rows; i++) {
      for (int j = 0; j < cols; j++) {
         scanf("%d", &matrix[i][j]);
      }
   }
}

void printMatrix(int matrix[MAX_SIZE][MAX_SIZE], int rows, int cols) {
   printf("Matrix (%dx%d):\n", rows, cols);
   for (int i = 0; i < rows; i++) {
      for (int j = 0; j < cols; j++) {
         printf("%d ", matrix[i][j]);
      }
      printf("\n");
   }
}

void calculateSums(int matrix[MAX_SIZE][MAX_SIZE], int rows, int cols) {
   int rowSums[MAX_SIZE] = {0}; // Array to store sums of each row
   int colSums[MAX_SIZE] = {0}; // Array to store sums of each column

   // Calculate row sums and column sums
   for (int i = 0; i < rows; i++) {
      for (int j = 0; j < cols; j++) {
         rowSums[i] += matrix[i][j];
         colSums[j] += matrix[i][j];
      }
   }

   // Print row sums
   printf("Sum of each row:\n");
   for (int i = 0; i < rows; i++) {
      printf("Row %d: %d\n", i + 1, rowSums[i]);
```

```
    }

    // Print column sums
    printf("Sum of each column:\n");
    for (int j = 0; j < cols; j++) {
        printf("Column %d: %d\n", j + 1, colSums[j]);
    }
}

int main() {
    int matrix[MAX_SIZE][MAX_SIZE];
    int rows, cols;

    // Input the dimensions of the matrix
    printf("Enter the number of rows and columns for the matrix: ");
    scanf("%d %d", &rows, &cols);

    // Input the matrix
    inputMatrix(matrix, rows, cols);

    // Output the original matrix
    printMatrix(matrix, rows, cols);

    // Calculate and display the sums
    calculateSums(matrix, rows, cols);

    return 0;
}
```

OUTPUT :-

```
Enter the number of rows and columns for the matrix: 2
3
Enter the elements of the matrix (2x3):
2
4
6
8
9
3
Matrix (2x3):
2 4 6
8 9 3
Sum of each row:
Row 1: 12
Row 2: 20
Sum of each column:
Column 1: 10
Column 2: 13
Column 3: 9
```

15.Write a program to perform the multiplication of two matrices.

Code :-

```
#include <stdio.h>

#define MAX_SIZE 10 // Define a maximum size for the matrices

void inputMatrix(int matrix[MAX_SIZE][MAX_SIZE], int rows, int cols) {
   printf("Enter the elements of the matrix (%dx%d):\n", rows, cols);
   for (int i = 0; i < rows; i++) {
      for (int j = 0; j < cols; j++) {
         scanf("%d", &matrix[i][j]);
      }
   }
}

void printMatrix(int matrix[MAX_SIZE][MAX_SIZE], int rows, int cols) {
   printf("Matrix (%dx%d):\n", rows, cols);
   for (int i = 0; i < rows; i++) {
      for (int j = 0; j < cols; j++) {
         printf("%d ", matrix[i][j]);
      }
      printf("\n");
   }
}

void multiplyMatrices(int A[MAX_SIZE][MAX_SIZE], int B[MAX_SIZE][MAX_SIZE], int
C[MAX_SIZE][MAX_SIZE], int rowsA, int colsA, int rowsB, int colsB) {
   // Initialize the resulting matrix to zero
   for (int i = 0; i < rowsA; i++) {
      for (int j = 0; j < colsB; j++) {
         C[i][j] = 0;
      }
   }

   // Perform multiplication
   for (int i = 0; i < rowsA; i++) {
      for (int j = 0; j < colsB; j++) {
         for (int k = 0; k < colsA; k++) {
            C[i][j] += A[i][k] * B[k][j];
         }
      }
   }
```

```
}

int main() {
    int A[MAX_SIZE][MAX_SIZE], B[MAX_SIZE][MAX_SIZE], C[MAX_SIZE][MAX_SIZE];
    int rowsA, colsA, rowsB, colsB;

    // Input dimensions for the first matrix
    printf("Enter the number of rows and columns for the first matrix: ");
    scanf("%d %d", &rowsA, &colsA);

    // Input dimensions for the second matrix
    printf("Enter the number of rows and columns for the second matrix: ");
    scanf("%d %d", &rowsB, &colsB);

    // Check if multiplication is possible
    if (colsA != rowsB) {
        printf("Matrix multiplication not possible. Number of columns of A must equal number
of rows of B.\n");
        return 1;
    }

    // Input the first matrix
    printf("Matrix A:\n");
    inputMatrix(A, rowsA, colsA);

    // Input the second matrix
    printf("Matrix B:\n");
    inputMatrix(B, rowsB, colsB);

    // Multiply the matrices
    multiplyMatrices(A, B, C, rowsA, colsA, rowsB, colsB);

    // Output the result
    printf("Result of Matrix A * Matrix B:\n");
    printMatrix(C, rowsA, colsB);

    return 0;
}
```

OUTPUT :-

```
Enter the number of rows and columns for the first matrix: 2
2
Enter the number of rows and columns for the second matrix: 2
2
Matrix A:
Enter the elements of the matrix (2x2):
3
4
5
6
Matrix B:
Enter the elements of the matrix (2x2):
6
7
8
9
Result of Matrix A * Matrix B:
Matrix (2x2):
50 57
78 89
```

16.Write a program to check whether a matrix is an identity matrix or not.

Code :-

```
#include <stdio.h>

#define MAX_SIZE 10 // Define a maximum size for the matrix

void inputMatrix(int matrix[MAX_SIZE][MAX_SIZE], int size) {
   printf("Enter the elements of the matrix (%dx%d):\n", size, size);
   for (int i = 0; i < size; i++) {
      for (int j = 0; j < size; j++) {
         scanf("%d", &matrix[i][j]);
      }
   }
}

int isIdentityMatrix(int matrix[MAX_SIZE][MAX_SIZE], int size) {
   for (int i = 0; i < size; i++) {
      for (int j = 0; j < size; j++) {
         if (i == j) {
            // Check diagonal elements
            if (matrix[i][j] != 1) {
               return 0; // Not an identity matrix
            }
         } else {
            // Check non-diagonal elements
            if (matrix[i][j] != 0) {
               return 0; // Not an identity matrix
            }
         }
      }
   }
   return 1; // It is an identity matrix
}

int main() {
   int matrix[MAX_SIZE][MAX_SIZE];
   int size;

   // Input the size of the square matrix
   printf("Enter the size of the matrix (n x n): ");
   scanf("%d", &size);

   // Input the matrix
```

```
    inputMatrix(matrix, size);

    // Check if the matrix is an identity matrix
    if (isIdentityMatrix(matrix, size)) {
        printf("The matrix is an identity matrix.\n");
    } else {
        printf("The matrix is not an identity matrix.\n");
    }

    return 0;
}
```

OUTPUT :-

```
Enter the size of the matrix (n x n): 2
Enter the elements of the matrix (2x2):
1
0
0
1
The matrix is an identity matrix.
```

17.Write a program to check whether a matrix is a sparse matrix or not .

Code :-

```
#include <stdio.h>

#define MAX_SIZE 10 // Define a maximum size for the matrix

void inputMatrix(int matrix[MAX_SIZE][MAX_SIZE], int rows, int cols) {
   printf("Enter the elements of the matrix (%dx%d):\n", rows, cols);
   for (int i = 0; i < rows; i++) {
      for (int j = 0; j < cols; j++) {
         scanf("%d", &matrix[i][j]);
      }
   }
}

int isSparseMatrix(int matrix[MAX_SIZE][MAX_SIZE], int rows, int cols) {
   int zeroCount = 0;
   int totalElements = rows * cols;

   // Count the number of zero elements
   for (int i = 0; i < rows; i++) {
      for (int j = 0; j < cols; j++) {
         if (matrix[i][j] == 0) {
            zeroCount++;
         }
      }
   }

   // A matrix is considered sparse if more than half of its elements are zero
   return (zeroCount > totalElements / 2);
}

int main() {
   int matrix[MAX_SIZE][MAX_SIZE];
   int rows, cols;

   // Input the dimensions of the matrix
   printf("Enter the number of rows and columns for the matrix: ");
   scanf("%d %d", &rows, &cols);

   // Input the matrix
   inputMatrix(matrix, rows, cols);
```

```
    // Check if the matrix is sparse
    if (isSparseMatrix(matrix, rows, cols)) {
        printf("The matrix is a sparse matrix.\n");
    } else {
        printf("The matrix is not a sparse matrix.\n");
    }

    return 0;
}
```

OUTPUT :-

```
Enter the number of rows and columns for the matrix: 2
3
Enter the elements of the matrix (2x3):
2
0
0
0
3
0
The matrix is a sparse matrix.
```

18.Write a C program to create a structure named company which has name, address, phone and no Of Employee as member variables. Read the name of the company, its address, phone and no Of Employee. Finally display these members‟ values.

Code :-

```
#include <stdio.h>

struct company {
   char name[100];
   char address[200];
   char phone[15];
   int noOfEmployees;
};

int main() {
   struct company comp;

   // Read company details
   printf("Enter the name of the company: ");
   fgets(comp.name, sizeof(comp.name), stdin);

   printf("Enter the address of the company: ");
   fgets(comp.address, sizeof(comp.address), stdin);

   printf("Enter the phone number of the company: ");
   fgets(comp.phone, sizeof(comp.phone), stdin);

   printf("Enter the number of employees: ");
   scanf("%d", &comp.noOfEmployees);

   // Display company details
   printf("\nCompany Details:\n");
   printf("Name: %s", comp.name);
   printf("Address: %s", comp.address);
   printf("Phone: %s", comp.phone);
   printf("Number of Employees: %d\n", comp.noOfEmployees);

   return 0;
}
```

OUTPUT :-

```
Enter the name of the company: tcs
Enter the address of the company: sector5
Enter the phone number of the company: 9988776655
Enter the number of employees: 80

Company Details:
Name: tcs
Address: sector5
Phone: 9988776655
Number of Employees: 80
```

19.Define a structure “complex” (typedef) to read two complex numbers and perform addition, and subtraction of these two complex numbers and display the result.

Code :-

```
#include <stdio.h>

typedef struct {
   float real;   // Real part
   float imag;   // Imaginary part
} complex;

// Function to add two complex numbers
complex add(complex a, complex b) {
   complex result;
   result.real = a.real + b.real;
   result.imag = a.imag + b.imag;
   return result;
}

// Function to subtract two complex numbers
complex subtract(complex a, complex b) {
   complex result;
   result.real = a.real - b.real;
   result.imag = a.imag - b.imag;
   return result;
}

// Function to display a complex number
void display(complex c) {
   if (c.imag >= 0) {
      printf("%.2f + %.2fi\n", c.real, c.imag);
   } else {
      printf("%.2f - %.2fi\n", c.real, -c.imag);
   }
}

int main() {
   complex num1, num2, sum, difference;

   // Read first complex number
   printf("Enter first complex number (real and imaginary parts): ");
   scanf("%f %f", &num1.real, &num1.imag);
```

```
    // Read second complex number
    printf("Enter second complex number (real and imaginary parts): ");
    scanf("%f %f", &num2.real, &num2.imag);

    // Perform addition and subtraction
    sum = add(num1, num2);
    difference = subtract(num1, num2);

    // Display results
    printf("\nSum: ");
    display(sum);

    printf("Difference: ");
    display(difference);

    return 0;
}
```

OUTPUT :-

```
Enter first complex number (real and imaginary parts): 3.5 2.5
Enter second complex number (real and imaginary parts): 1.5 4.5

Sum: 5.00 + 7.00i
Difference: 2.00 - 2.00i
```

20.Write a C program to read the RollNo, Name, Address, and Age marks of 12 students in the BCT class and display the details from the function.

Code :-

```
#include <stdio.h>

#define MAX_STUDENTS 12

typedef struct {
   int rollNo;
   char name[100];
   char address[200];
   int age;
   float marks;
} Student;

void readStudentDetails(Student students[], int count) {
   for (int i = 0; i < count; i++) {
      printf("\nEnter details for student %d:\n", i + 1);
      printf("Roll No: ");
      scanf("%d", &students[i].rollNo);
      getchar(); // Consume newline character left by scanf

      printf("Name: ");
      fgets(students[i].name, sizeof(students[i].name), stdin);

      printf("Address: ");
      fgets(students[i].address, sizeof(students[i].address), stdin);

      printf("Age: ");
      scanf("%d", &students[i].age);

      printf("Marks: ");
      scanf("%f", &students[i].marks);
   }
}

void displayStudentDetails(const Student students[], int count) {
   printf("\nStudent Details:\n");
   for (int i = 0; i < count; i++) {
      printf("\nStudent %d:\n", i + 1);
      printf("Roll No: %d\n", students[i].rollNo);
      printf("Name: %s", students[i].name);
```

```
        printf("Address: %s", students[i].address);
        printf("Age: %d\n", students[i].age);
        printf("Marks: %.2f\n", students[i].marks);
    }
}

int main() {
    Student students[MAX_STUDENTS];

    // Read student details
    readStudentDetails(students, MAX_STUDENTS);

    // Display student details
    displayStudentDetails(students, MAX_STUDENTS);

    return 0;
}
```

OUTPUT :-

```
Enter details for student 1:
Roll No: 2
Name: abc
Address: pqr
Age: 22
Marks: 79

Enter details for student 2:
Roll No: 3
Name: rst
Address: efg
Age: 23
Marks: 87

Enter details for student 3:
Roll No: 4
Name: xyz
Address: dsf
Age: 26
Marks: 90

Enter details for student 4:
Roll No: 8
Name: jkl
Address: rfc
```

Age: 27
Marks: 8

Enter details for student 5:
Roll No: 9
Name: xzc
Address: bnm
Age: 28
Marks: 92

Enter details for student 6:
Roll No: 1
Name: asd
Address: wqe
Age: 21
Marks: 88

Enter details for student 7:
Roll No: 12
Name: vnn
Address: dcy
Age: 19
Marks: 76

Enter details for student 8:
Roll No: 14
Name: eds
Address: tcg
Age: 22
Marks: 89

Enter details for student 9:
Roll No: 51
Name: ttt
Address: yyy
Age: 30
Marks: 94

Enter details for student 10:
Roll No: 17
Name: eee
Address: fff
Age: 34
Marks: 99

Enter details for student 11:
Roll No: 33
Name: ccs
Address: bbb
Age: 38
Marks: 98

Enter details for student 12:
Roll No: 37
Name: zzz
Address: mmm
Age: 32
Marks: 95

Student Details:

Student 1:
Roll No: 2
Name: abc
Address: pqr
Age: 22
Marks: 79.00

Student 2:
Roll No: 3
Name: rst
Address: efg
Age: 23
Marks: 87.00

Student 3:
Roll No: 4
Name: xyz
Address: dsf
Age: 26
Marks: 90.00

Student 4:
Roll No: 8
Name: jkl
Address: rfc
Age: 27
Marks: 8.00

Student 5:
Roll No: 9
Name: xzc
Address: bnm
Age: 28
Marks: 92.00

Student 6:
Roll No: 1
Name: asd
Address: wqe
Age: 21
Marks: 88.00

Student 7:
Roll No: 12
Name: vnn
Address: dcy
Age: 19
Marks: 76.00

Student 8:
Roll No: 14
Name: eds
Address: tcg
Age: 22
Marks: 89.00

Student 9:
Roll No: 51
Name: ttt
Address: yyy
Age: 30
Marks: 94.00

Student 10:
Roll No: 17
Name: eee
Address: fff
Age: 34
Marks: 99.00

Student 11:

Roll No: 33
Name: ccs
Address: bbb
Age: 38
Marks: 98.00

Student 12:
Roll No: 37
Name: zzz
Address: mmm
Age: 32
Marks: 95.00