**Roll No: 123CS0021 Name: Koushik Sripathi**

**Assignment I: Programming Assignment Questions on Arrays and Matrices in C**

1. Reverse Array: Write a program to reverse an array of integers. The user should not input the number of elements and the elements themselves, and your program should output the reversed array.

**CODE:**

#include <stdio.h>

#include <stdlib.h>

int main(){

    //Size of the array

    int n;

    scanf("%d ", &n);

    int arr[n];

    printf("ORIGINAL ARRAY \n");

    for (int i=0; i<n; i++){

        arr[i] = rand()%(100);

        printf("%d ", arr[i]);

    }

    //reversing the array

    printf("\n");

    // Using a temporary variable to switch the elements in an array

    for (int i=0; i<= n/2; i++){

        int temp = arr[i];

        arr[i] = arr[n-i-1];

        arr[n-i-1] = temp;

    }

    printf("REVERSED ARRAY \n");

    //Printing the array which is reversed

    for (int i=0; i<n; i++){

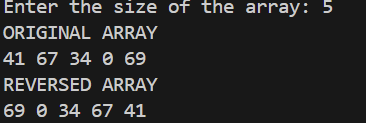
        printf("%d ", arr[i]);

    }

    return 0;

}

**OUTPUT:**



2. Matrix Transposition: Create a program that transposes a matrix. The program should generate the random matrix as the input and output the transposed matrix.

**CODE:**

#include <stdio.h>

#include <stdlib.h>

int main(){

    int n;

    printf("Enter the number of rows: \n");

    scanf("%d", &n);

    int c;

    printf("Enter the number of columns: \n");

    scanf("%d", &c);

    int arr[n][n];

    //Printing the original matrix as input

    printf("ORIGNAL MATRIX \n");

    for (int i=0; i<n; i++){

        for (int j=0; j<c ; j++){

            arr[i][j] = rand()%(200);

            printf("%d", arr[i][j]);

            printf("\t\t");

        }

        printf("\n");

    }

    //transpose the elements in the matrix

    int rev\_arr[c][n];

    printf("\n");

    printf("TRANSPOSE OF A MATRIX \n");

    for (int i=0; i<n; i++){

        for (int j=0; j<c; j++){

            rev\_arr[j][i] = arr[i][j];

        }

    }

    //transpose matrix

    for (int i=0; i<c; i++){

        for (int j=0; j<n; j++){

            printf("%d ", rev\_arr[i][j]);

            printf("\t\t");

        }

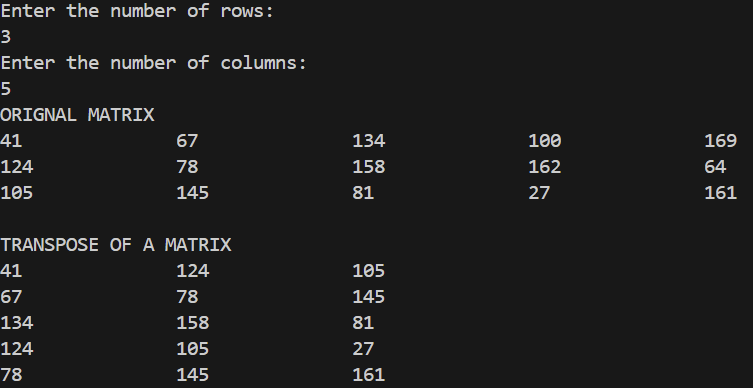
        printf("\n");

    }

    return 0;

}

**OUTPUT:**



3. Check Symmetric Matrix: Write a program that checks if a square matrix is symmetric. A matrix is symmetric if it is equal to its transpose.

**CODE:**

#include <stdio.h>

#include <stdlib.h>

int main(){

    int n;

    printf("Enter the number of rows: \n");

    scanf("%d", &n);

    int c;

    printf("Enter the number of columns: \n");

    scanf("%d", &c);

    int arr[n][n];

    //Printing the original matrix as input

    printf("ORIGNAL MATRIX \n");

    for (int i=0; i<n; i++){

        for (int j=0; j<c ; j++){

            arr[i][j] = rand()%(200);

            printf("%d", arr[i][j]);

            printf("\t\t");

        }

        printf("\n");

    }

    //transpose the elements in the matrix

    int rev\_arr[c][n];

    printf("\n");

    printf("TRANSPOSE OF A MATRIX \n");

    for (int i=0; i<n; i++){

        for (int j=0; j<c; j++){

            rev\_arr[j][i] = arr[i][j];

        }

    }

    //transpose matrix

    for (int i=0; i<c; i++){

        for (int j=0; j<n; j++){

            printf("%d ", rev\_arr[i][j]);

            printf("\t\t");

        }

        printf("\n");

    }

    // condition for symmetric

    if (n == c){

        if (rev\_arr[c][n] == arr[n][c]){

            printf("SYMMETRIC \n");

        } else {

            printf("NON-SYMMETRIC \n");

        }

    } else {

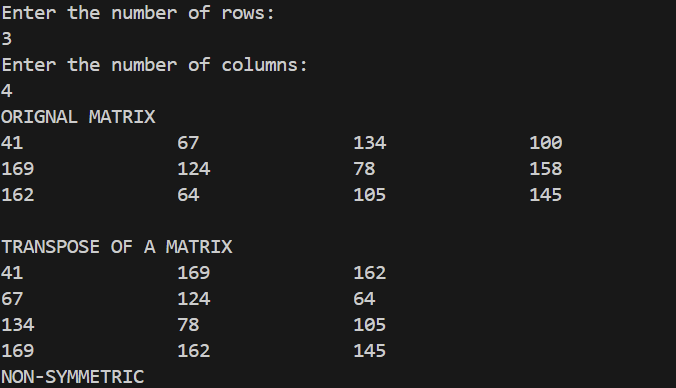
        printf("NON-SYMMETRIC \n");

    }

    return 0;

}

**OUTPUT:**



4. Sparse Matrix Representation: Implement a program that converts a matrix into its sparse matrix representation and displays the result. The sparse representation should list only the non-zero elements and their row and column indices.

**CODE:**

#include <stdio.h>

#include <stdlib.h>

int main(){

    //input matrix

    int n;

    scanf("%d", &n);

    int mat[n][n];

    for (int i=0; i<n; i++){

        for (int j=0; j<n; j++){

            printf("Enter the elements: ");

            scanf("%d", &mat[i][j]);

        }

    }

    //sparse matrix representation

    printf("SPARSE MATRIX \n");

    for (int i=0; i<n; i++){

        for (int j=0; j<n; j++){

            if (mat[i][j] != 0){

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

            }

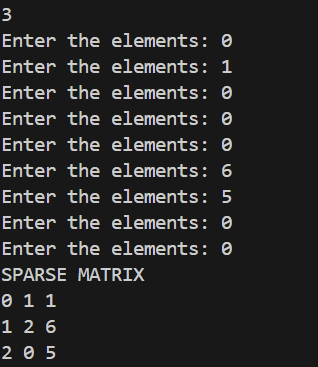
        }

    }

    return 0;

}

**OUTPUT:**



5. Diagonal Sum: Develop a program that calculates the sum of the main diagonal elements of a square matrix. Generate the random matrix of different data types to be given as input.

**CODE:**

#include <stdio.h>

#include <stdlib.h>

int main(){

    int n;

    printf("Enter the number of rows and columns:");

    printf("\n");

    scanf("%d", &n);

    int mat[n][n];

    // input matrix

    for (int i=0; i<n; i++){

        for (int j=0; j<n; j++){

            mat[i][j] = rand()%(100);

            printf("%d ", mat[i][j]);

            printf("\t\t");

        }

        printf("\n");

    }

    printf("\n");

    // adding the diagonal elements in amtrix

    int sum=0;

    for (int i=0; i<n;i++){

        for (int j=0; j<n;j++){

            if (i == j){

                sum += mat[i][j];

            } else {

                continue;

            }

        }

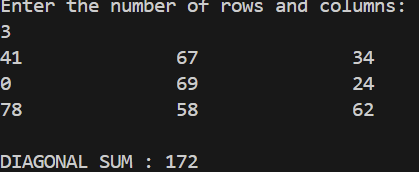
    }

    printf("DIAGONAL SUM : %d", sum);

    return 0;

}

**OUTPUT:**



6. Row and Column Sum: Write a program that calculates the sum of each row and each column of a matrix. The program should display the sums separately for rows and columns.

**CODE:**

#include <stdio.h>

#include <stdlib.h>

int main(){

    int n;

    scanf("%d", &n);

    int mat[n][n];

    for (int i=0; i<n; i++){

        for (int j=0; j<n; j++){

            mat[i][j] = rand()%(200);

            printf("%d ", mat[i][j]);

            printf("\t\t");

        }

        printf("\n");

    }

    int sum=0, i=0, j=0;

    printf("ROW SUM: \n");

    while(j<n){

        for (int i=0; i<n; i++){

            sum+=mat[i][j];

        }

        j++;

        printf("%d ", sum);

        sum=0;

    }

    printf("\n");

    printf("COLUMN SUM: \n");

    while(i<n){

        for (int j=0; j<n; j++){

            sum+=mat[i][j];

        }

        i++;

        printf("%d ", sum);

        sum=0;

    }

    return 0;

}

7. Rotate Matrix by 90 Degrees: Create a program that rotates a square matrix by 90 degrees clockwise. The user should input the matrix, and the output should be the rotated matrix.

**CODE:**

#include <stdio.h>

#include <stdlib.h>

int main(){

    //input matrix

    int n;

    printf("Enter the number of rows: ");

    printf("\n");

    scanf("%d", &n);

    int c;

    printf("Enter the number of rows: ");

    printf("\n");

    scanf("%d", &c);

    //rotating the matrix

    int mat[n][c];

    for (int i=0; i<n; i++){

        for (int j=0; j<c; j++){

            mat[i][j] = rand()%(200);

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

        }

        printf("\n");

    }

    int trans[c][n];

    for (int i=0; i<c; i++){

        for (int j=0; j<n; j++){

            trans[i][j] = mat[j][i];

        }

    }

    for (int i=0; i<c; i++){

        for (int j=0; j<n/2; j++){

            int temp = trans[i][j];

            trans[i][j] = trans[i][n-j-1];

            trans[i][n-j-1] = temp;

        }

    }

    printf("\n");

    printf("ROTATED MATRIX \n");

    //print the final matrix

    for (int i=0; i<c; i++){

        for (int j=0; j<n; j++){

            printf("%d \t\t", trans[i][j]);

        }

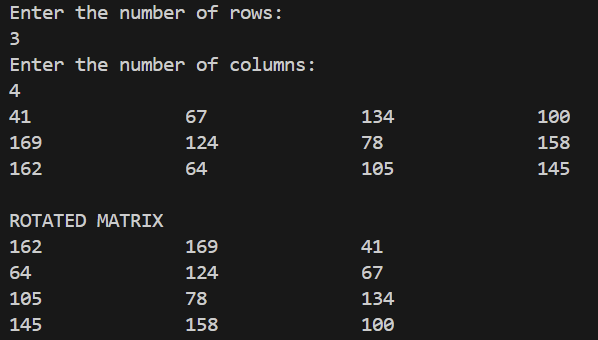
        printf("\n");

    }

return 0;

}

**OUTPUT:**



8. Multiplication of Two Matrices: Implement a program that multiplies two matrices. The user inputs the dimensions and elements of both matrices. Your program should output the resulting matrix.\

**CODE:**

#include <stdio.h>

#include <stdlib.h>

int main() {

    // Input matrices

    printf("MATRIX 1 \n");

    int r1, c1;

    printf("Enter the number of rows: ");

    scanf("%d", &r1);

    printf("Enter the number of columns: ");

    scanf("%d", &c1);

    int mat\_1[10][10];

    for (int i = 0; i < r1; i++) {

        for (int j = 0; j < c1; j++) {

            mat\_1[i][j] = rand() % 200;

            printf("%d \t\t", mat\_1[i][j]);

        }

        printf("\n");

    }

    printf("\nMATRIX 2 \n");

    int r2, c2;

    printf("Enter the number of rows: ");

    scanf("%d", &r2);

    printf("Enter the number of columns: ");

    scanf("%d", &c2);

    // Check if matrix multiplication is possible

    if (c1 != r2) {

        printf("Multiplication is not possible due to incompatible dimensions.\n");

        return 0;

    }

    int mat\_2[10][10];

    for (int i = 0; i < r2; i++) {

        for (int j = 0; j < c2; j++) {

            mat\_2[i][j] = rand() % 200;

            printf("%d \t\t", mat\_2[i][j]);

        }

        printf("\n");

    }

    // Matrix multiplication

    int prod[10][10] = {0};

    for (int i = 0; i < r1; i++) {

        for (int j = 0; j < c2; j++) {

            for (int k = 0; k < c1; k++) {

                prod[i][j] += mat\_1[i][k] \* mat\_2[k][j];

            }

        }

    }

    // Printing the product matrix

    printf("\nPRODUCT MATRIX \n");

    for (int i = 0; i < r1; i++) {

        for (int j = 0; j < c2; j++) {

            printf("%d \t\t", prod[i][j]);

        }

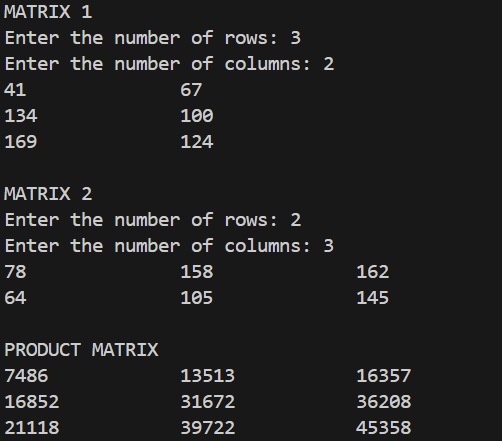
        printf("\n");

    }

    return 0;

}

**OUTPUT:**



9. Dynamic Array Operations: Write a program that performs operations like insert, delete, and search on an array. The size of the array should dynamically increase or decrease as elements are added or removed.

**CODE:**

#include <stdio.h>

#include <stdlib.h>

int main(){

    int \*arr;

    int size = 0;

    int max = 5;

    int choice;

    // Dynamically allocate memory for the array

    arr = (int\*)malloc(max \* sizeof(int));

    if (arr == NULL){

        printf("Memory allocation failed!\n");

        return 1;

    }

    while (1) {

        // Display options

        printf("\nMenu:\n");

        printf("1. Insert an element into the array\n");

        printf("2. Delete an element from the array\n");

        printf("3. Search for an element in the array\n");

        printf("4. Print the array\n");

        printf("5. Exit\n");

        printf("Enter your choice: ");

        scanf("%d", &choice);

        switch (choice) {

            case 1: {

                // Insert operation

                int insert;

                printf("Enter the element to be inserted: ");

                scanf("%d", &insert);

                // Check if the array is full and resize if needed

                if (size == max) {

                    max \*= 2;

                    arr = realloc(arr, max \* sizeof(int));

                    if (arr == NULL) {

                        printf("Memory re-allocation failed.\n");

                        return 1;

                    }

                }

                // Insert the new element

                arr[size] = insert;

                size++;

                // Print the array after insertion

                printf("Array: ");

                for (int i = 0; i < size; i++) {

                    printf("%d ", arr[i]);

                }

                printf("\n");

                break;

            }

            case 2: {

                // Delete operation

                int element, found = 0;

                printf("Enter the element to be deleted: ");

                scanf("%d", &element);

                // Search for the element to delete

                for (int i = 0; i < size; i++) {

                    if (arr[i] == element) {

                        found = 1;

                        for (int j = i; j < size - 1; j++) {

                            arr[j] = arr[j + 1];

                        }

                        size--;

                        break;

                    }

                }

                if (found) {

                    printf("Element %d has been deleted.\n", element);

                    if (size <= max / 2 && max > 5) {

                        max /= 2;

                        arr = realloc(arr, max \* sizeof(int));

                        if (arr == NULL) {

                            printf("Memory re-allocation failed.\n");

                            return 1;

                        }

                    }

                } else {

                    printf("Element not found.\n");

                }

                break;

            }

            case 3: {

                // Search operation

                int element, found = 0;

                printf("Enter the element to be searched: ");

                scanf("%d", &element);

                // Search for the element

                for (int i = 0; i < size; i++) {

                    if (arr[i] == element) {

                        printf("Element %d is found at index %d.\n", element, i);

                        found = 1;

                        break;

                    }

                }

                if (!found) {

                    printf("Element %d is not in the array.\n", element);

                }

                break;

            }

            case 4: {

                // Print the array

                printf("Array: ");

                for (int i = 0; i < size; i++) {

                    printf("%d ", arr[i]);

                }

                printf("\n");

                break;

            }

            case 5: {

                // Exit the program

                printf("Exiting.\n");

                free(arr);

                return 0;

            }

            default: {

                printf("Invalid choice! Please choose a valid option.\n");

                break;

            }

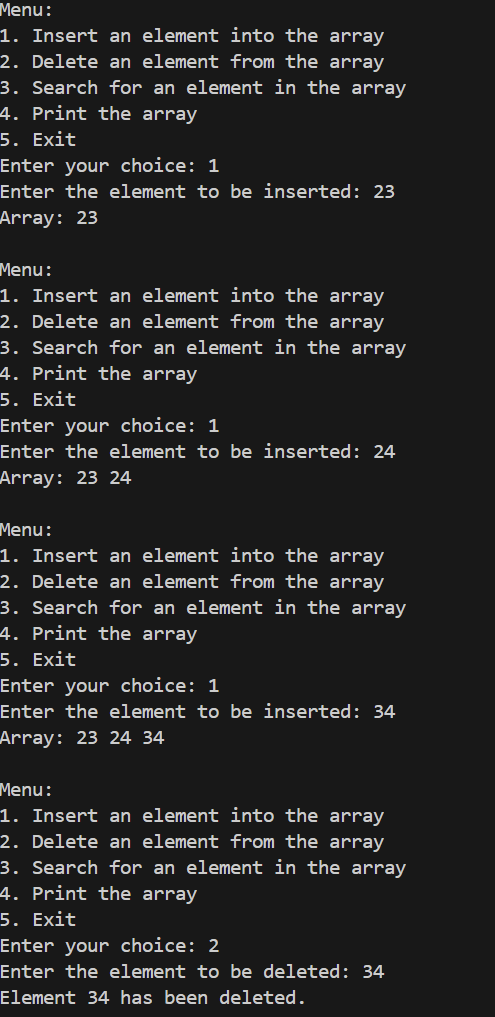
        }

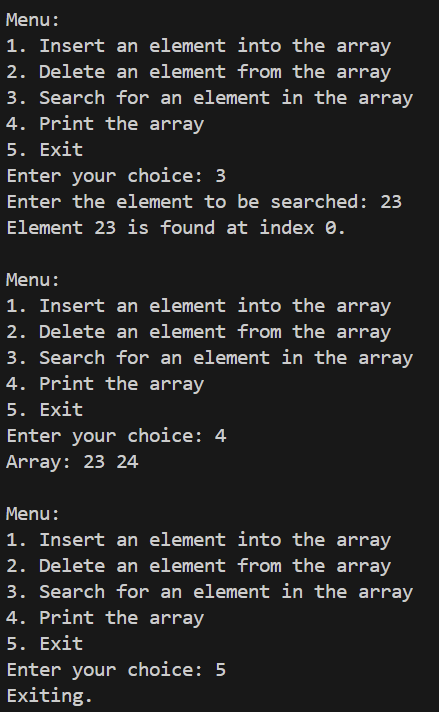
    }

    return 0;

}

**OUTPUT:**





10. Pascal’s Triangle Using Arrays: Develop a program that generates Pascal's triangle up to a given number of rows. The number of rows should be input by the user, and the output should display the triangle.

**CODE:**

#include <stdio.h>

int main(){

    int n;

    printf("Enter the number of rows: ");

    scanf("%d", &n);

    int triangle[n][n];

    //initialize the first row to 1's

    for (int i=0; i<n; i++){

        triangle[0][i] = 1;

    }

    // compute subsequent forms

    for (int i=1; i<n; i++){

        for (int a=0; a<=i; a++){

            if (a == 0 | a==i){

                triangle[i][a] = 1;

            } else {

                triangle[i][a] = triangle[i-1][a-1] + triangle[i-1][a];

            }

        }

    }

    //display triangle

    for (int i=0; i<n; i++){

        for (int j=0; j<=i; j++){

            printf("%d ", triangle[i][j]);

        }

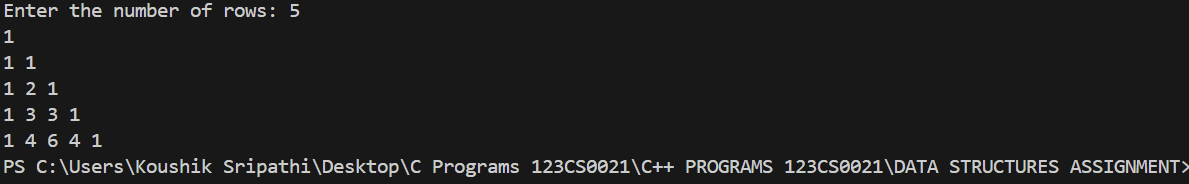
        printf("\n");

    }

    return 0;

}

**OUTPUT:**



**----------------------------------- Good luck ----------------------------------**