EDU

EAST DELTA UNIVERSITY

LAB REPORT- 02

Course title: Structured Programming Laboratory

Course Code: CSE 114

Submitted by,

Name: Mohammad Fahim
ID: 242002112
Section: 4
Department: CSE

Submitted to,

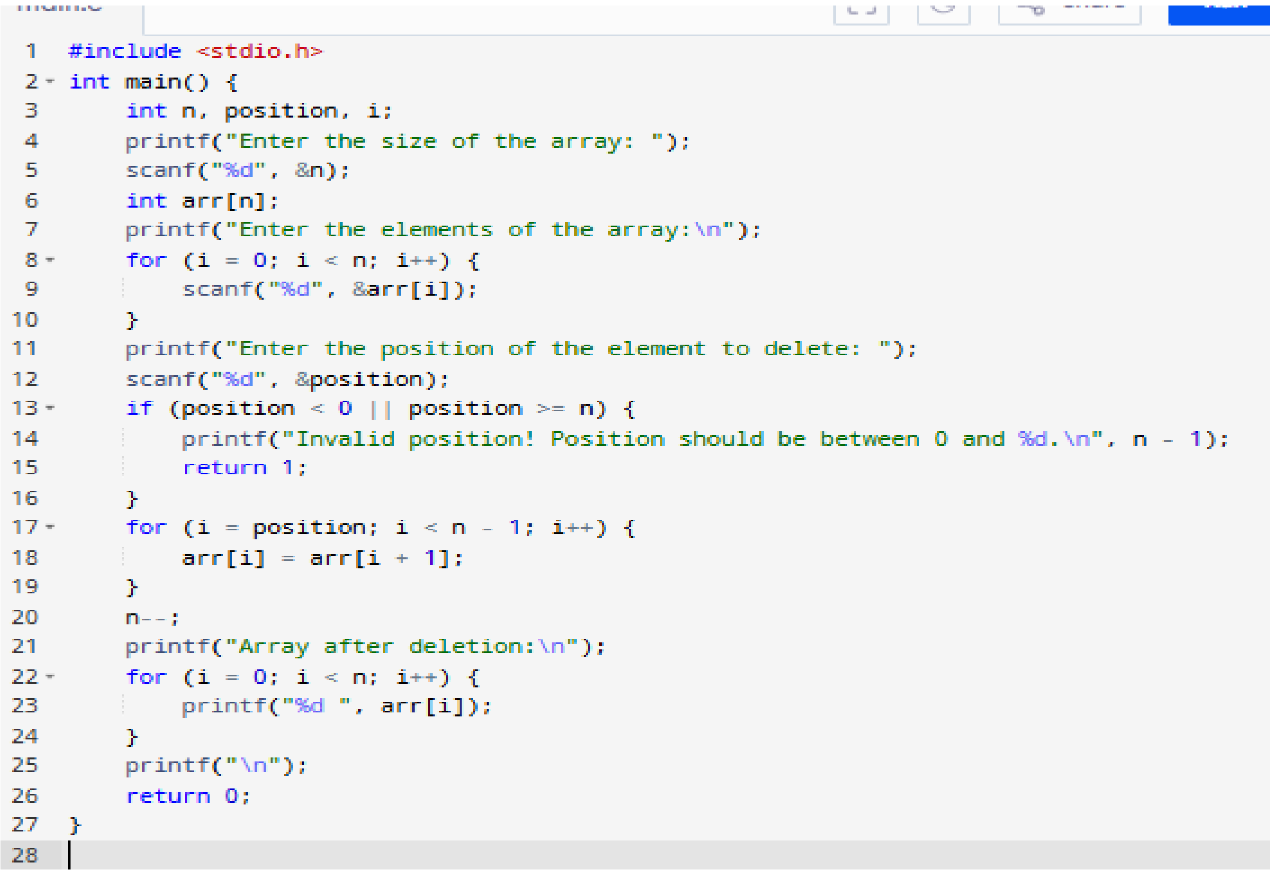
Name: ANIKA BUSHRA

Designation: Lecturer, SoSET

Date Of Submission: 17-12-2024

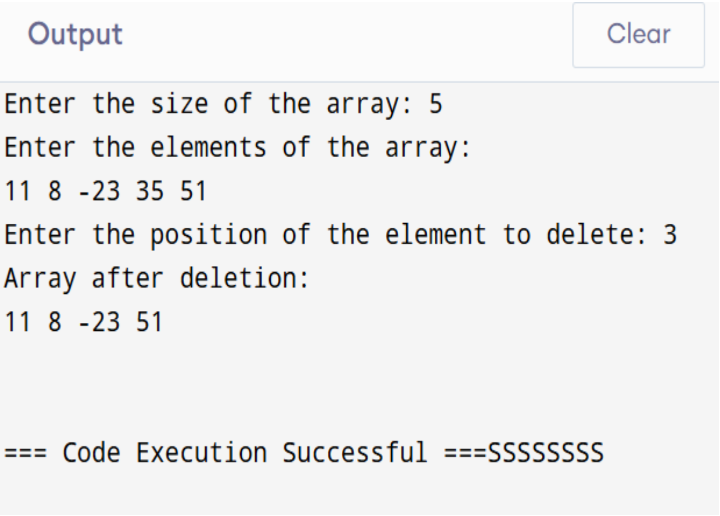
1. The Disappearing Number

Code:

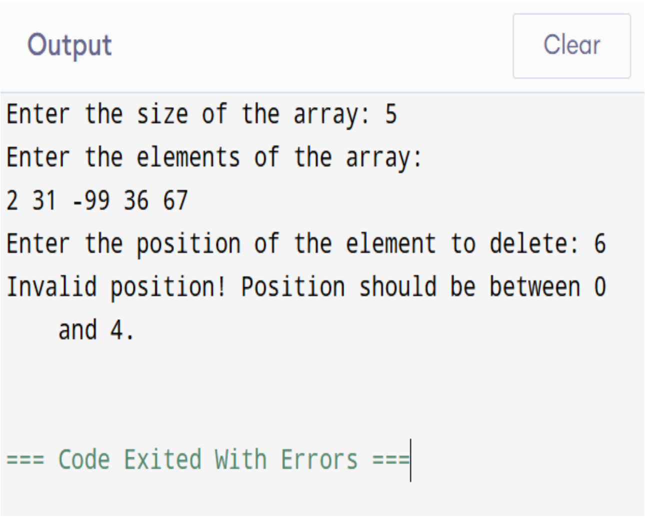


```c
#include <stdio.h>
int main() {
    int n, position, i;
    printf("Enter the size of the array: ");
    scanf("%d", &n);
    int arr[n];
    printf("Enter the elements of the array:\n");
    for (i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    printf("Enter the position of the element to delete: ");
    scanf("%d", &position);
    if (position < 0 || position >= n) {
        printf("Invalid position! Position should be between 0 and %d.\n", n - 1);
        return 1;
    }
    for (i = position; i < n - 1; i++) {
        arr[i] = arr[i + 1];
    }
    n--;
    printf("Array after deletion:\n");
    for (i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
    return 0;
}

```

Input/Output:



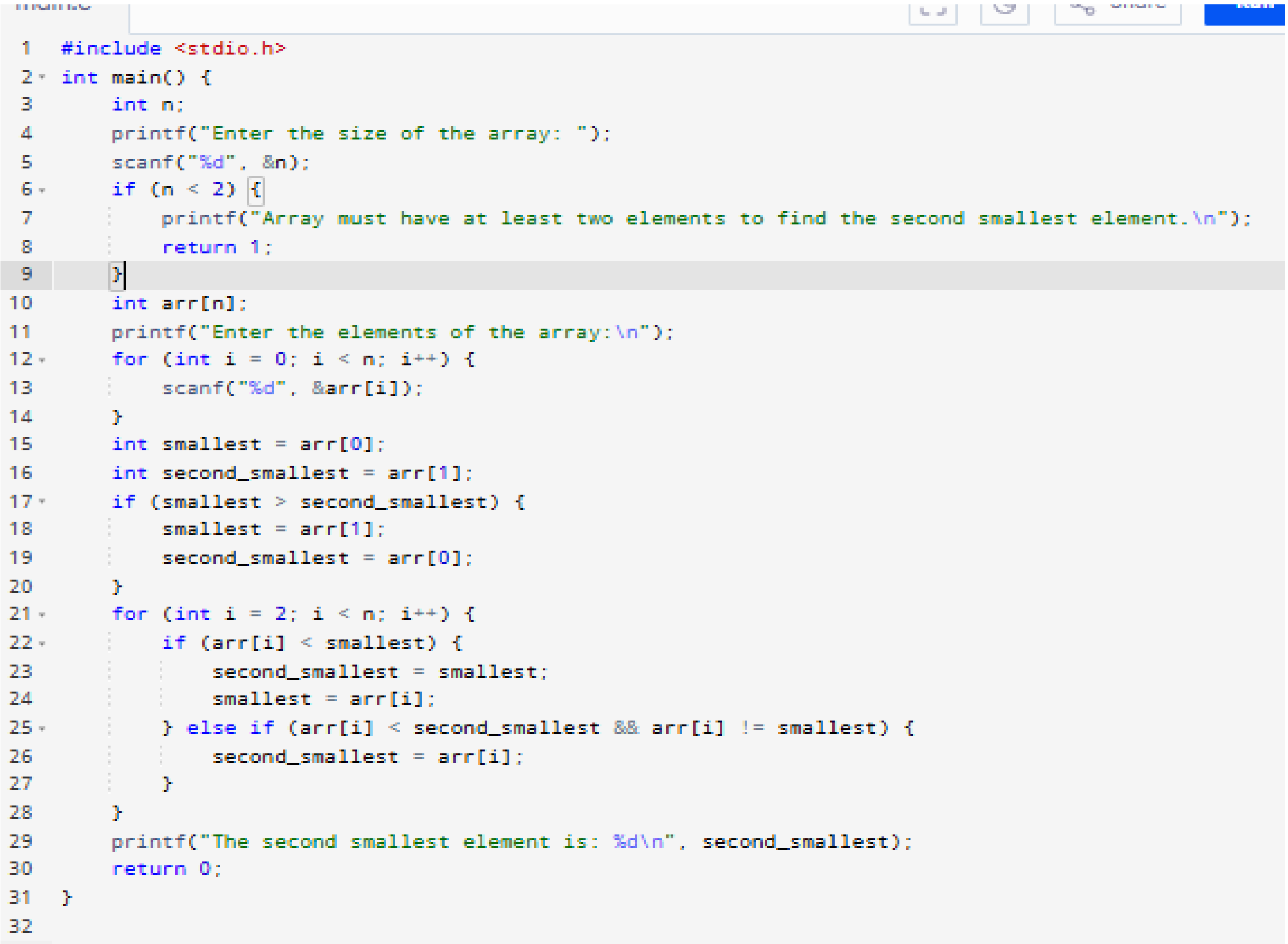
```
Enter the size of the array: 5
Enter the elements of the array:
11 8 -23 35 51
Enter the position of the element to delete: 3
Array after deletion:
11 8 -23 51


=== Code Execution Successful ===SSSSSSSS
```



```
Enter the size of the array: 5
Enter the elements of the array:
2 31 -99 36 67
Enter the position of the element to delete: 6
Invalid position! Position should be between 0
    and 4.


=== Code Exited With Errors ===
```

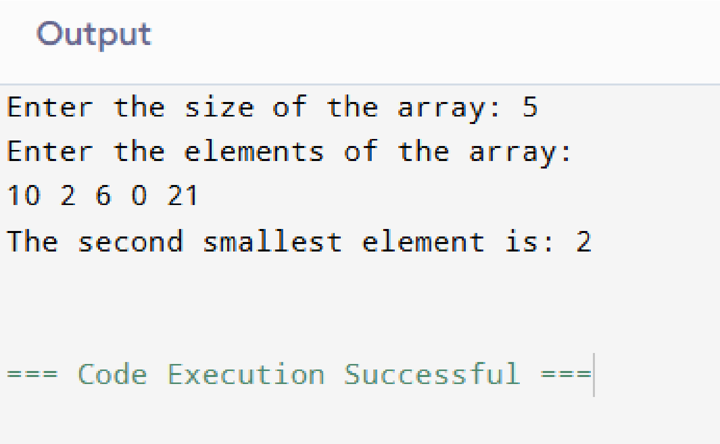
2. SquareRoots

Code:

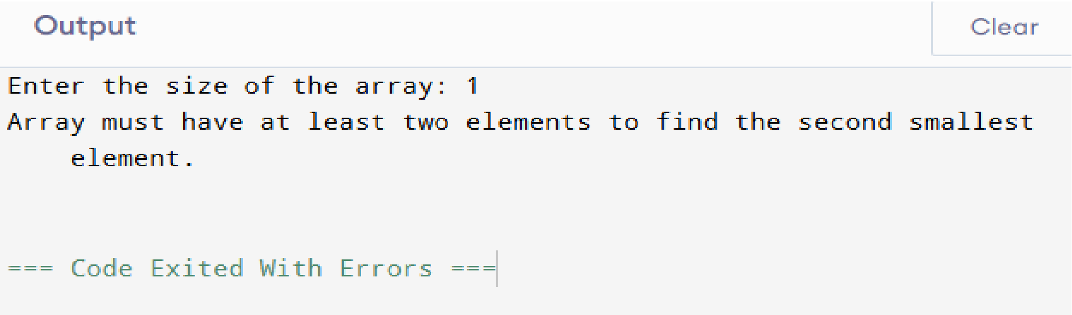


```c
#include <stdio.h>
int main() {
    int n;
    printf("Enter the size of the array: ");
    scanf("%d", &n);
    if (n < 2) {
        printf("Array must have at least two elements to find the second smallest element.\n");
        return 1;
    }
    int arr[n];
    printf("Enter the elements of the array:\n");
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    int smallest = arr[0];
    int second_smallest = arr[1];
    if (smallest > second_smallest) {
        smallest = arr[1];
        second_smallest = arr[0];
    }
    for (int i = 2; i < n; i++) {
        if (arr[i] < smallest) {
            second_smallest = smallest;
            smallest = arr[i];
        } else if (arr[i] < second_smallest && arr[i] != smallest) {
            second_smallest = arr[i];
        }
    }
    printf("The second smallest element is: %d\n", second_smallest);
    return 0;
}

```

Input/Output:



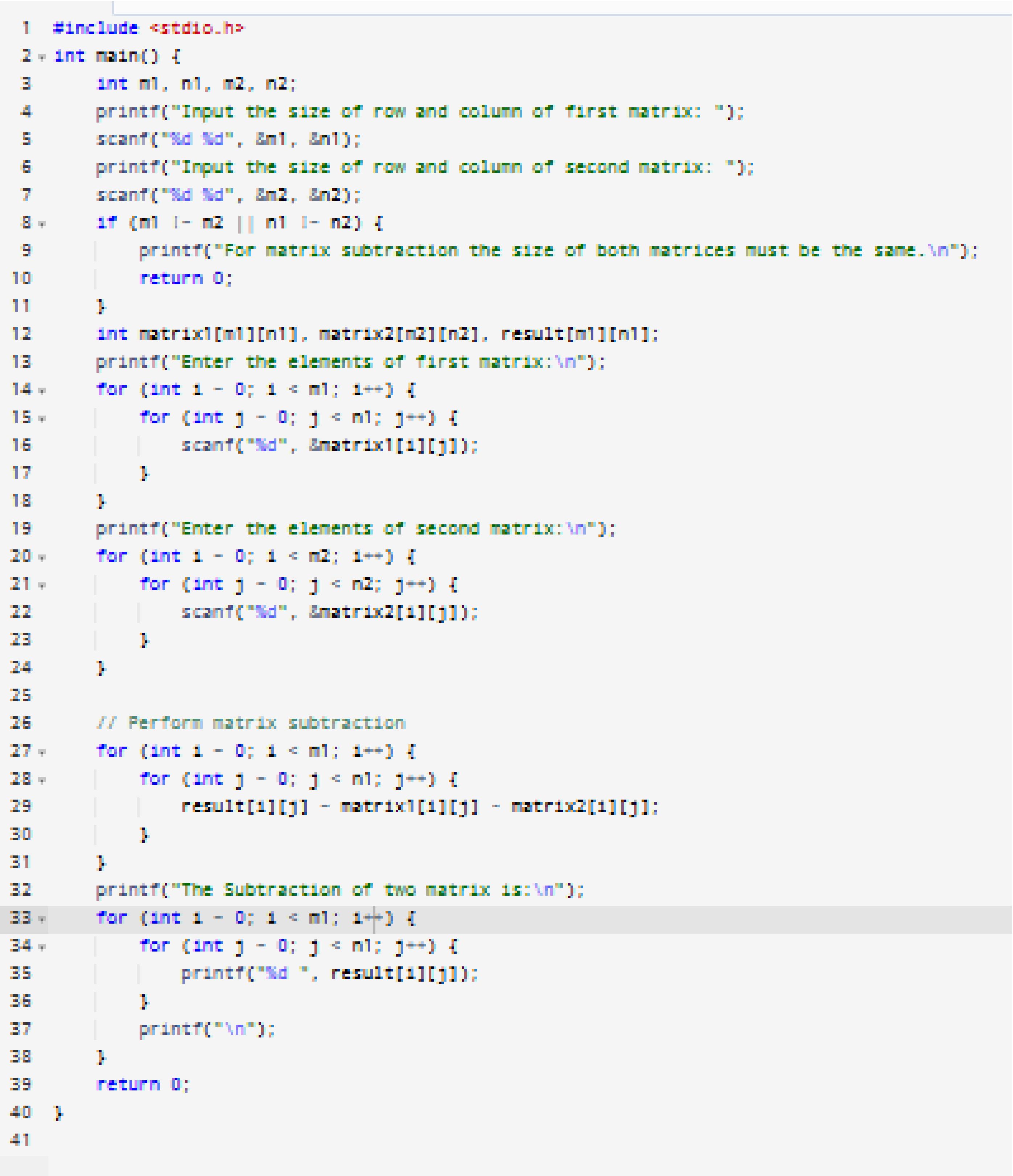
```
Enter the size of the array: 5
Enter the elements of the array:
10 2 6 0 21
The second smallest element is: 2


=== Code Execution Successful ===
```



```
Enter the size of the array: 1
Array must have at least two elements to find the second smallest
    element.


=== Code Exited With Errors ===
```

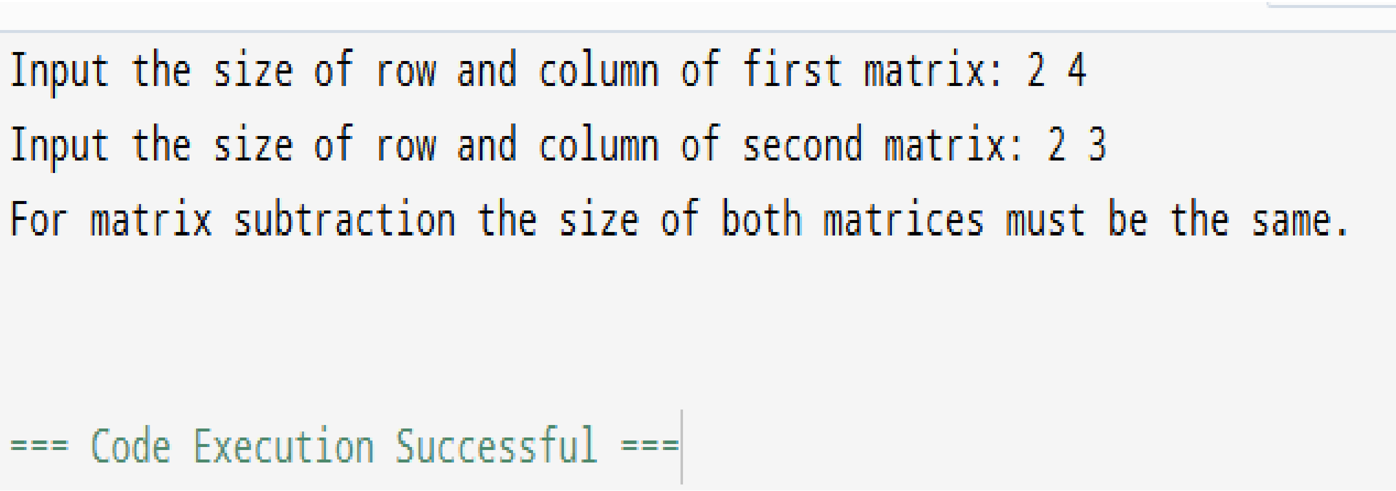
3. Subtracting Grids

Code:

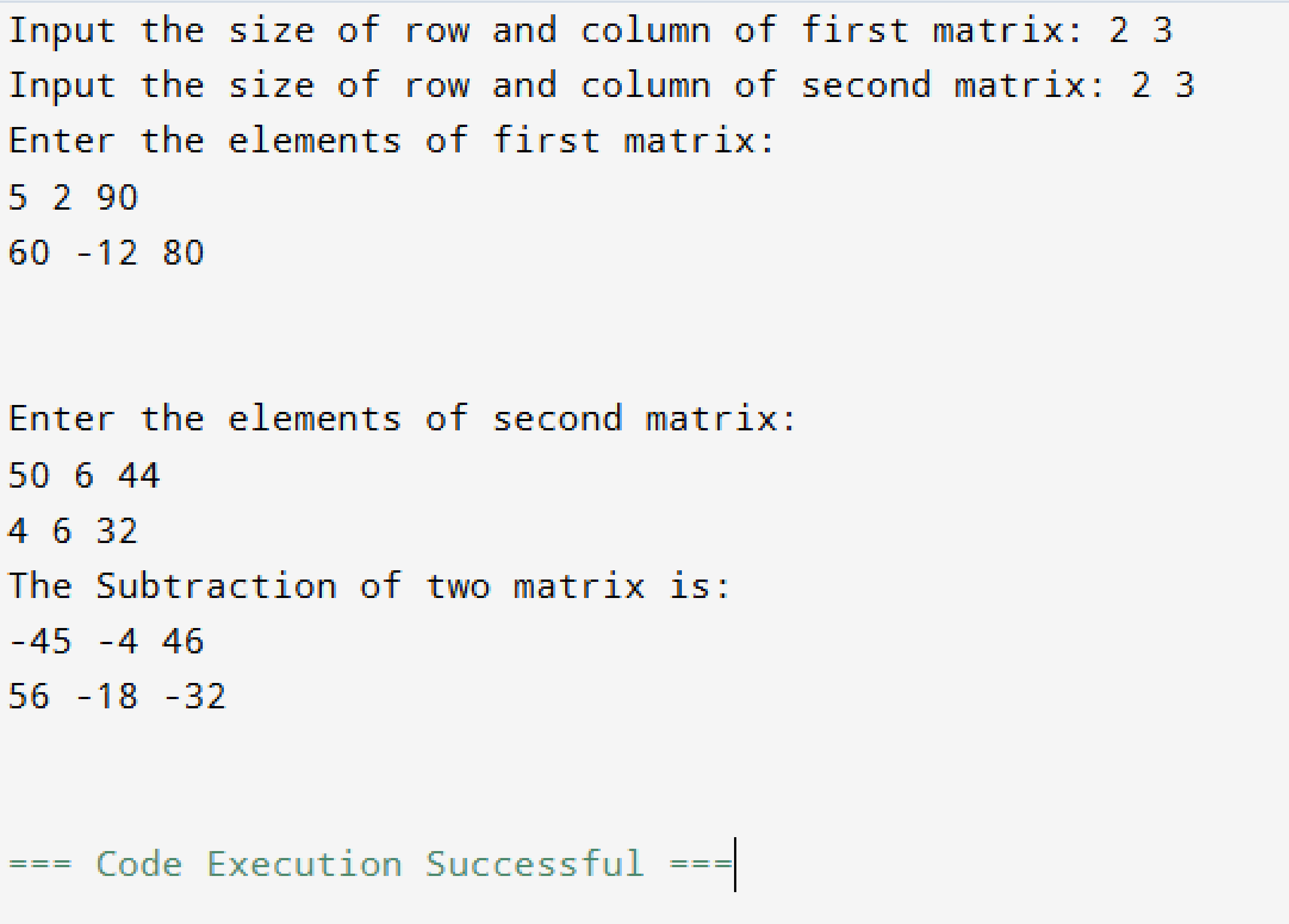


```
#include <stdio.h>
int main() {
    int m1, n1, m2, n2;
    printf("Input the size of row and column of first matrix: ");
    scanf("%d %d", &m1, &n1);
    printf("Input the size of row and column of second matrix: ");
    scanf("%d %d", &m2, &n2);
    if (m1 != m2 || n1 != n2) {
        printf("For matrix subtraction the size of both matrices must be the same.\n");
        return 0;
    }
    int matrix1[m1][n1], matrix2[m2][n2], result[m1][n1];
    printf("Enter the elements of first matrix:\n");
    for (int i = 0; i < m1; i++) {
        for (int j = 0; j < n1; j++) {
            scanf("%d", &matrix1[i][j]);
        }
    }
    printf("Enter the elements of second matrix:\n");
    for (int i = 0; i < m2; i++) {
        for (int j = 0; j < n2; j++) {
            scanf("%d", &matrix2[i][j]);
        }
    }

    // Perform matrix subtraction
    for (int i = 0; i < m1; i++) {
        for (int j = 0; j < n1; j++) {
            result[i][j] = matrix1[i][j] - matrix2[i][j];
        }
    }
    printf("The Subtraction of two matrix is:\n");
    for (int i = 0; i < m1; i++) {
        for (int j = 0; j < n1; j++) {
            printf("%d ", result[i][j]);
        }
        printf("\n");
    }
    return 0;
}

```

Input/Output:



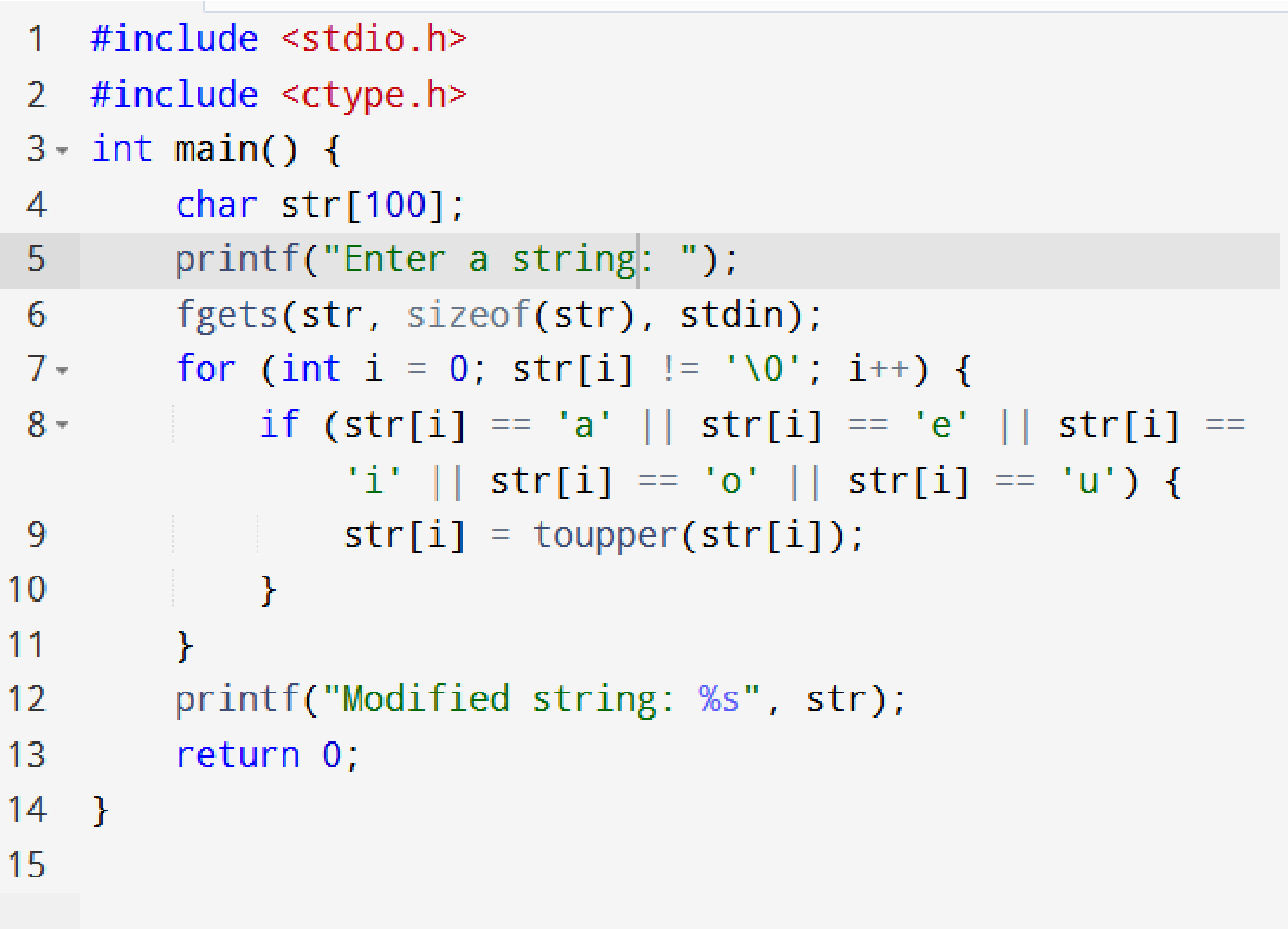
```
Input the size of row and column of first matrix: 2 4
Input the size of row and column of second matrix: 2 3
For matrix subtraction the size of both matrices must be the same.



=== Code Execution Successful ===
```



```
Input the size of row and column of first matrix: 2 3
Input the size of row and column of second matrix: 2 3
Enter the elements of first matrix:
5 2 90
60 -12 80


Enter the elements of second matrix:
50 6 44
4 6 32
The Subtraction of two matrix is:
-45 -4 46
56 -18 -32


=== Code Execution Successful ===
```

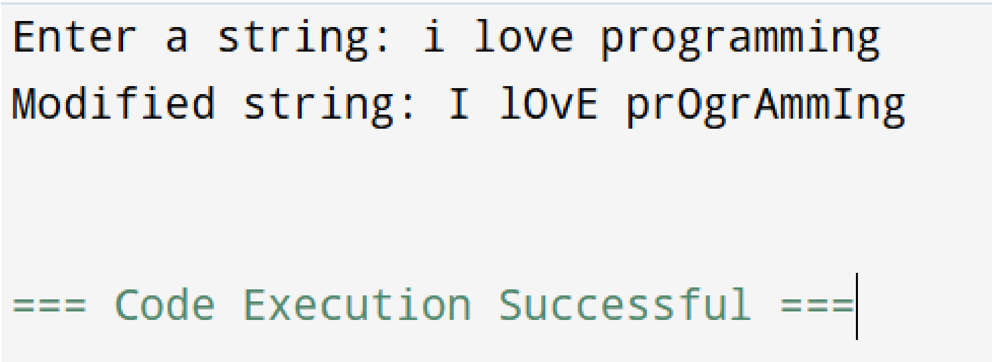
4. Vowel Uplift: A Capital Transformation

Code:



```c
#include <stdio.h>
#include <ctype.h>
int main() {
    char str[100];
    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);
    for (int i = 0; str[i] != '\0'; i++) {
        if (str[i] == 'a' || str[i] == 'e' || str[i] == 'i' || str[i] == 'o' || str[i] == 'u') {
            str[i] = toupper(str[i]);
        }
    }
    printf("Modified string: %s", str);
    return 0;
}
```

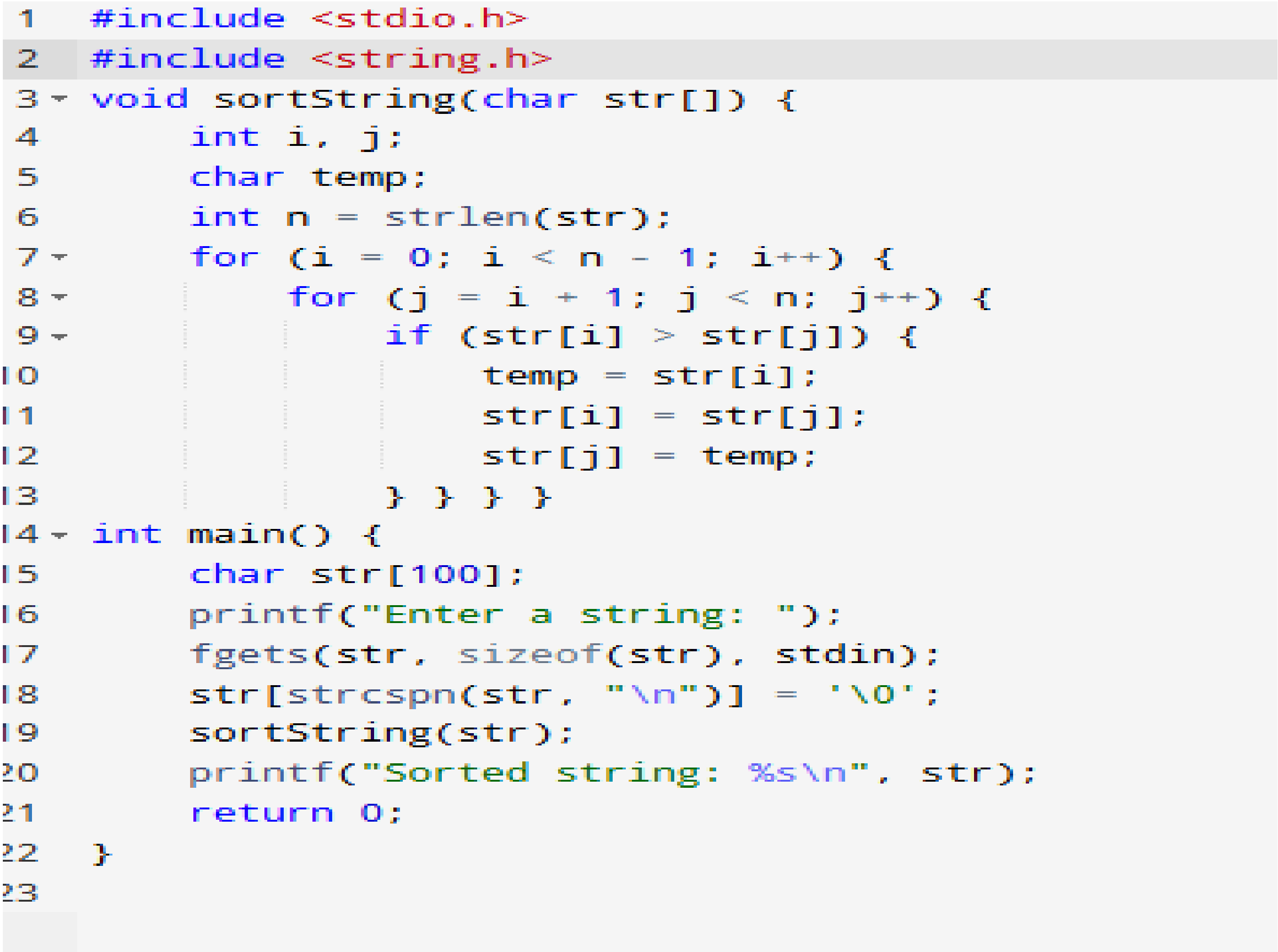
Input/Output:



```
Enter a string: i love programming
Modified string: I lOvE prOgrAmmIng


=== Code Execution Successful ===
```

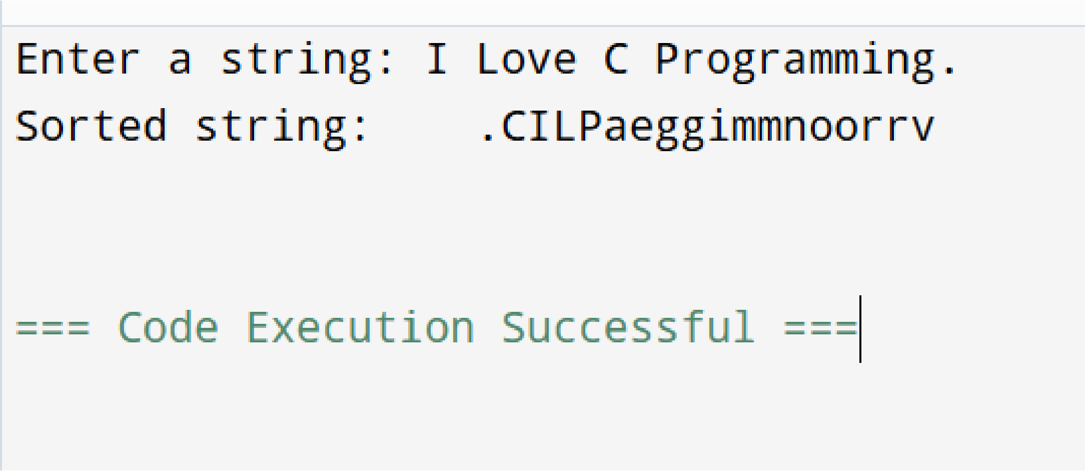
5. Strings in Order: From Chaos to Harmony

Code:



```c
#include <stdio.h>
#include <string.h>
void sortString(char str[]) {
    int i, j;
    char temp;
    int n = strlen(str);
    for (i = 0; i < n - 1; i++) {
        for (j = i + 1; j < n; j++) {
            if (str[i] > str[j]) {
                temp = str[i];
                str[i] = str[j];
                str[j] = temp;
            } } } }
int main() {
    char str[100];
    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);
    str[strcspn(str, "\n")] = '\0';
    sortString(str);
    printf("Sorted string: %s\n", str);
    return 0;
}
```

Input/Output:



```
Enter a string: I Love C Programming.
Sorted string:    .CILPaeggimmnoorrv



=== Code Execution Successful ===
```