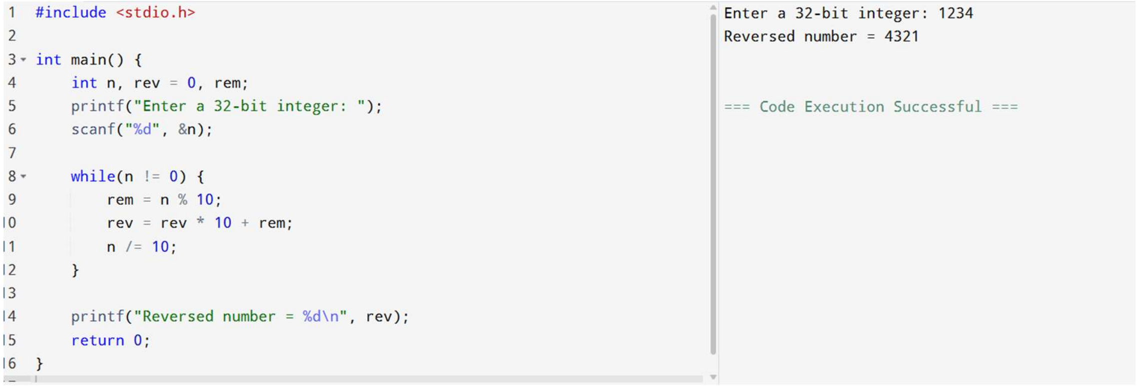
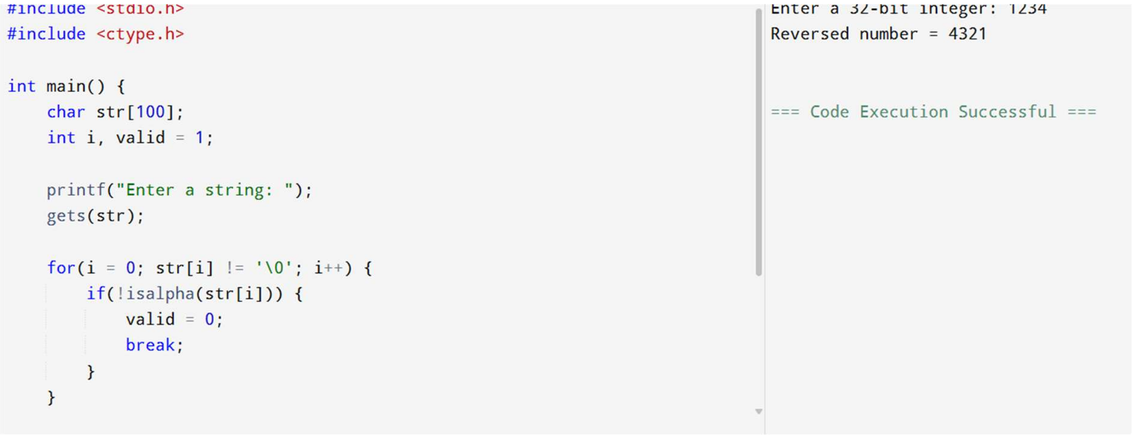
EX:1



```c
#include <stdio.h>

int main() {
    int n, rev = 0, rem;
    printf("Enter a 32-bit integer: ");
    scanf("%d", &n);

    while(n != 0) {
        rem = n % 10;
        rev = rev * 10 + rem;
        n /= 10;
    }

    printf("Reversed number = %d\n", rev);
    return 0;
}
```

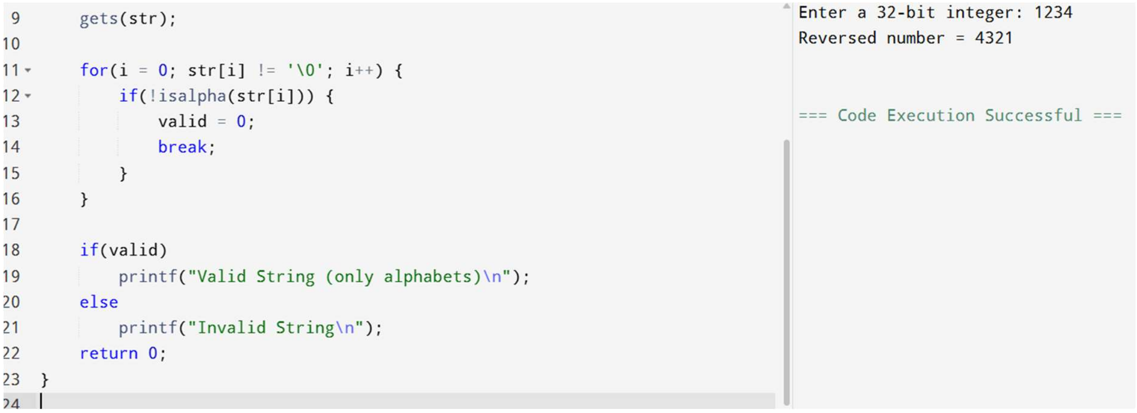
```
Enter a 32-bit integer: 1234
Reversed number = 4321


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

EX: 2



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

int main() {
    char str[100];
    int i, valid = 1;

    printf("Enter a string: ");
    gets(str);

    for(i = 0; str[i] != '\0'; i++) {
        if(!isalpha(str[i])) {
            valid = 0;
            break;
        }
    }
```

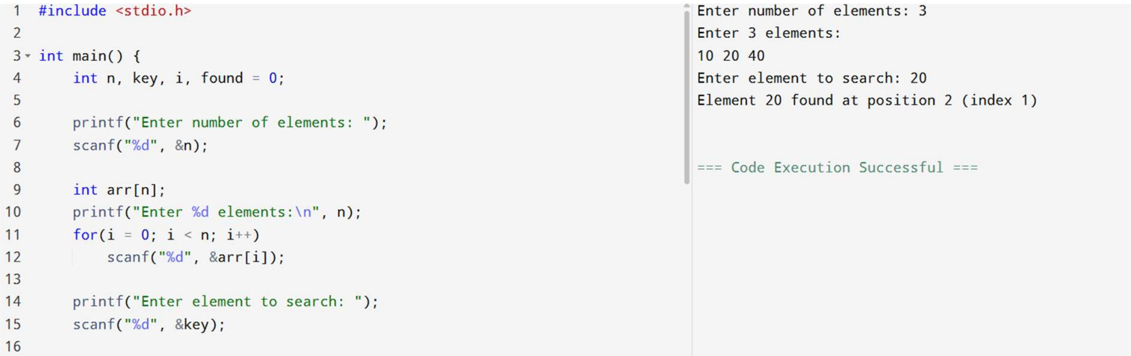
```
Enter a 32-bit integer: 1234
Reversed number = 4321


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



```c
    gets(str);

    for(i = 0; str[i] != '\0'; i++) {
        if(!isalpha(str[i])) {
            valid = 0;
            break;
        }
    }

    if(valid)
        printf("Valid String (only alphabets)\n");
    else
        printf("Invalid String\n");
    return 0;
}
```

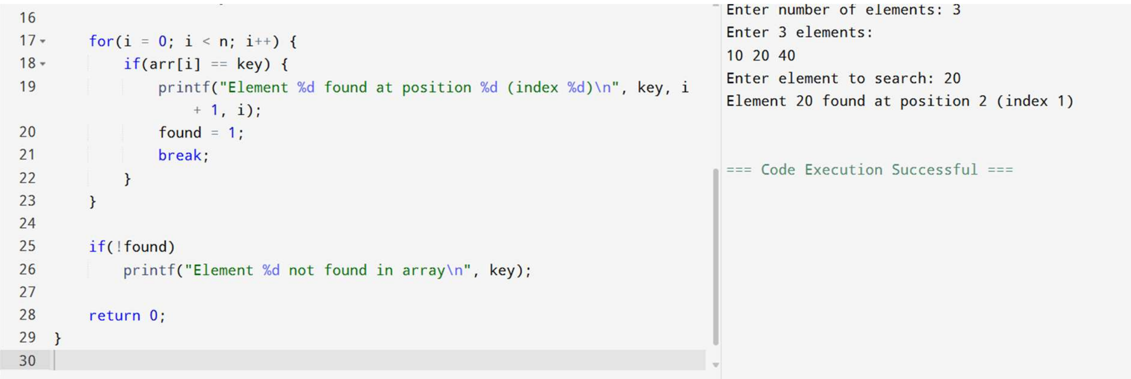
```
Enter a 32-bit integer: 1234
Reversed number = 4321


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

EX:6 A



```c
#include <stdio.h>

int main() {
    int n, key, i, found = 0;

    printf("Enter number of elements: ");
    scanf("%d", &n);

    int arr[n];
    printf("Enter %d elements:\n", n);
    for(i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    printf("Enter element to search: ");
    scanf("%d", &key);

```

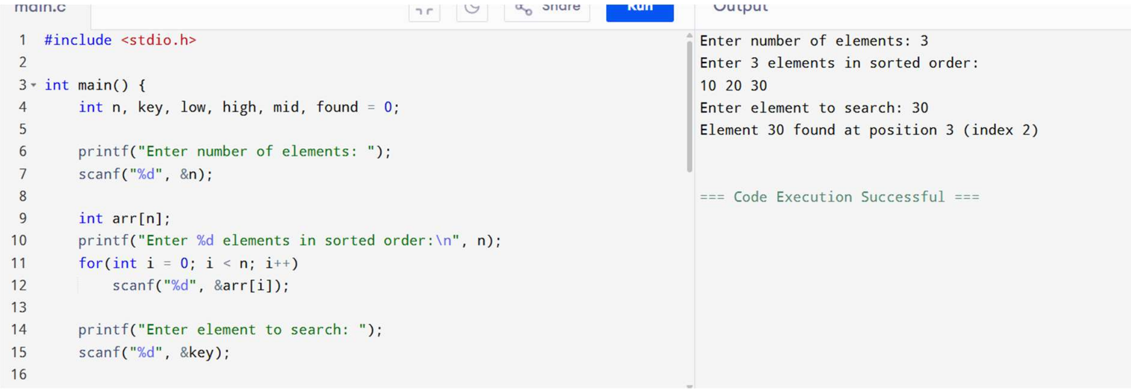
```
Enter number of elements: 3
Enter 3 elements:
10 20 40
Enter element to search: 20
Element 20 found at position 2 (index 1)


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



```c

    for(i = 0; i < n; i++) {
        if(arr[i] == key) {
            printf("Element %d found at position %d (index %d)\n", key, i
                + 1, i);
            found = 1;
            break;
        }
    }

    if(!found)
        printf("Element %d not found in array\n", key);

    return 0;
}

```

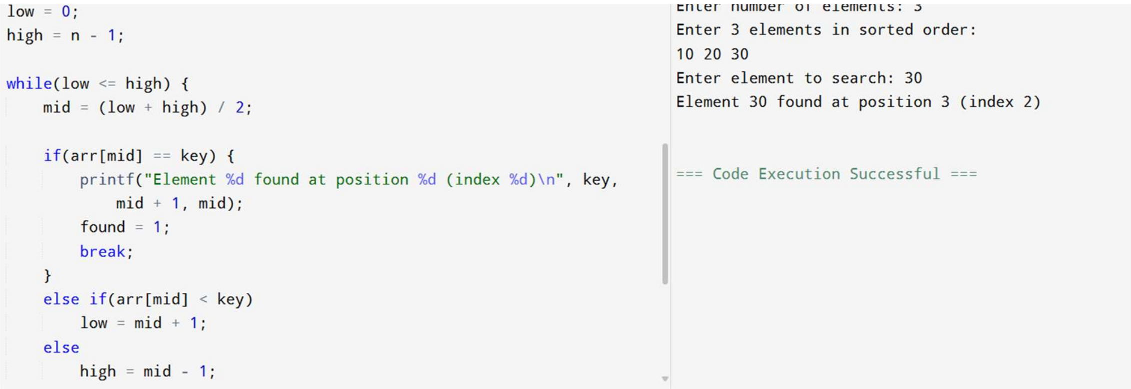
```
Enter number of elements: 3
Enter 3 elements:
10 20 40
Enter element to search: 20
Element 20 found at position 2 (index 1)


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

EX 6 B



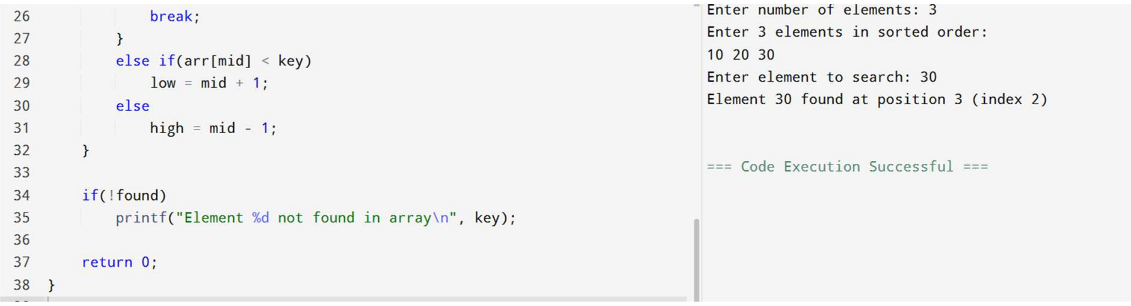
```c
#include <stdio.h>

int main() {
    int n, key, low, high, mid, found = 0;

    printf("Enter number of elements: ");
    scanf("%d", &n);

    int arr[n];
    printf("Enter %d elements in sorted order:\n", n);
    for(int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    printf("Enter element to search: ");
    scanf("%d", &key);

```

```
Enter number of elements: 3
Enter 3 elements in sorted order:
10 20 30
Enter element to search: 30
Element 30 found at position 3 (index 2)


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



```c
low = 0;
high = n - 1;

while(low <= high) {
    mid = (low + high) / 2;

    if(arr[mid] == key) {
        printf("Element %d found at position %d (index %d)\n", key,
            mid + 1, mid);
        found = 1;
        break;
    }
    else if(arr[mid] < key)
        low = mid + 1;
    else
        high = mid - 1;
```

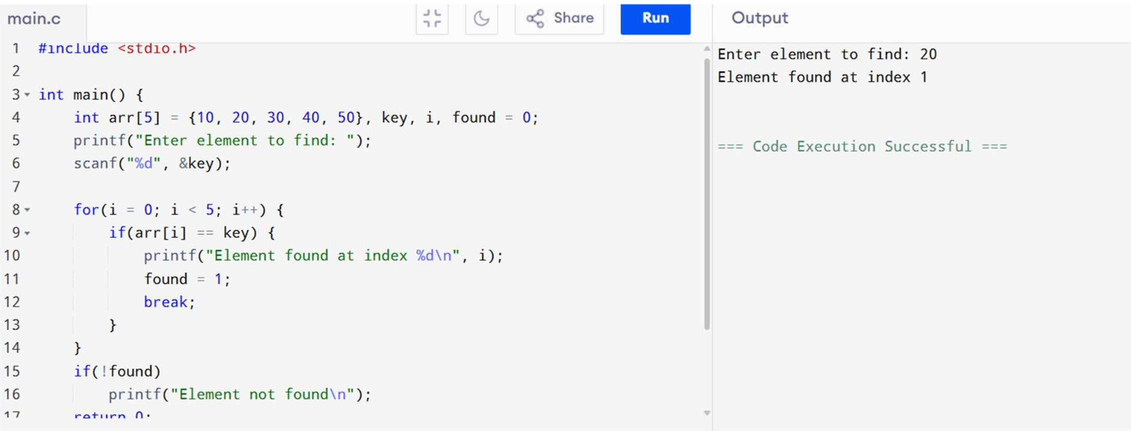
```
Enter number of elements: 3
Enter 3 elements in sorted order:
10 20 30
Enter element to search: 30
Element 30 found at position 3 (index 2)


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



```c
            break;
        }
        else if(arr[mid] < key)
            low = mid + 1;
        else
            high = mid - 1;
    }

    if(!found)
        printf("Element %d not found in array\n", key);

    return 0;
}
```

```
Enter number of elements: 3
Enter 3 elements in sorted order:
10 20 30
Enter element to search: 30
Element 30 found at position 3 (index 2)

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

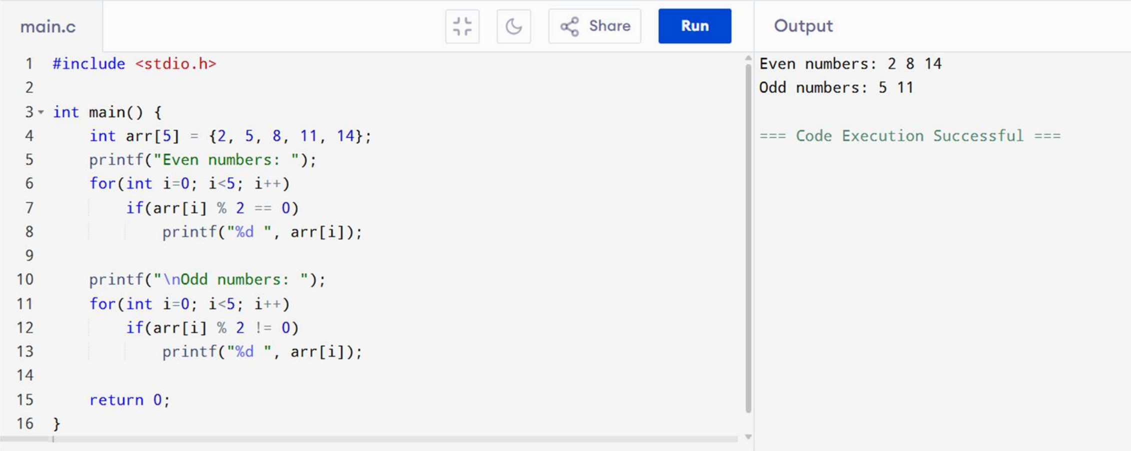
EX:7



```c
#include <stdio.h>

int main() {
    int arr[5] = {10, 20, 30, 40, 50}, key, i, found = 0;
    printf("Enter element to find: ");
    scanf("%d", &key);

    for(i = 0; i < 5; i++) {
        if(arr[i] == key) {
            printf("Element found at index %d\n", i);
            found = 1;
            break;
        }
    }
    if(!found)
        printf("Element not found\n");
    return 0;
```

```
Enter element to find: 20
Element found at index 1


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

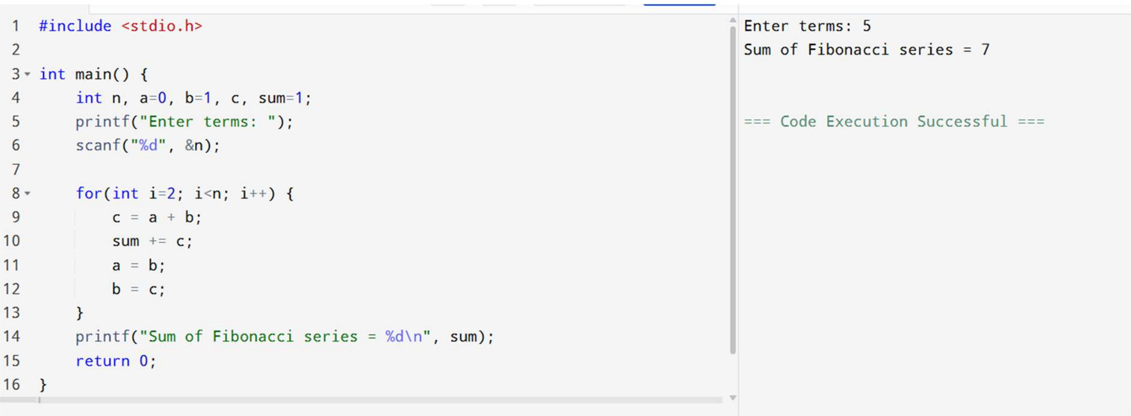
EX:8



```c
#include <stdio.h>

int main() {
    int arr[5] = {2, 5, 8, 11, 14};
    printf("Even numbers: ");
    for(int i=0; i<5; i++)
        if(arr[i] % 2 == 0)
            printf("%d ", arr[i]);

    printf("\nOdd numbers: ");
    for(int i=0; i<5; i++)
        if(arr[i] % 2 != 0)
            printf("%d ", arr[i]);

    return 0;
}
```

```
Even numbers: 2 8 14
Odd numbers: 5 11

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

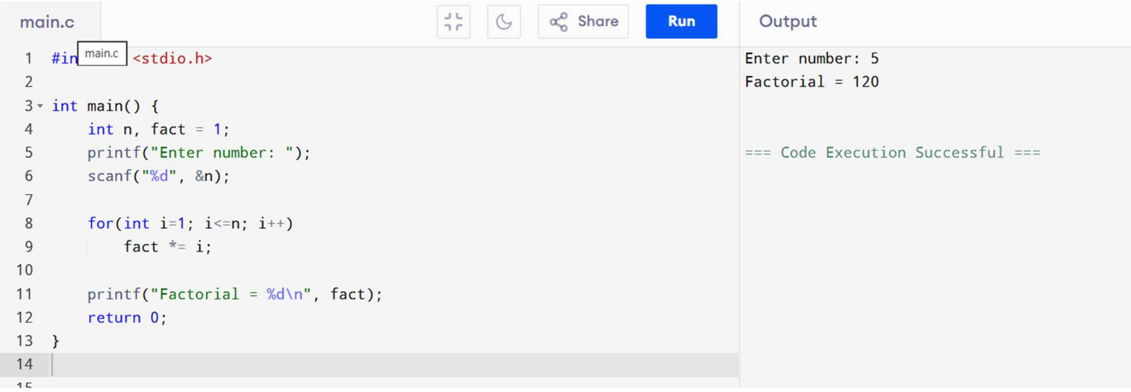
EX:9



```c
#include <stdio.h>

int main() {
    int n, a=0, b=1, c, sum=1;
    printf("Enter terms: ");
    scanf("%d", &n);

    for(int i=2; i<n; i++) {
        c = a + b;
        sum += c;
        a = b;
        b = c;
    }
    printf("Sum of Fibonacci series = %d\n", sum);
    return 0;
}
```

```
Enter terms: 5
Sum of Fibonacci series = 7


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

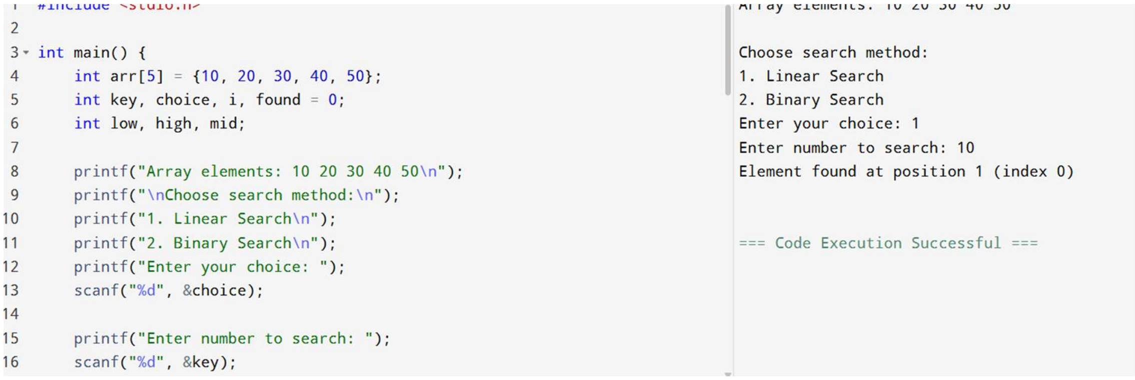
EX:10



```c
#in<stdio.h>

int main() {
    int n, fact = 1;
    printf("Enter number: ");
    scanf("%d", &n);

    for(int i=1; i<=n; i++)
        fact *= i;

    printf("Factorial = %d\n", fact);
    return 0;
}

```

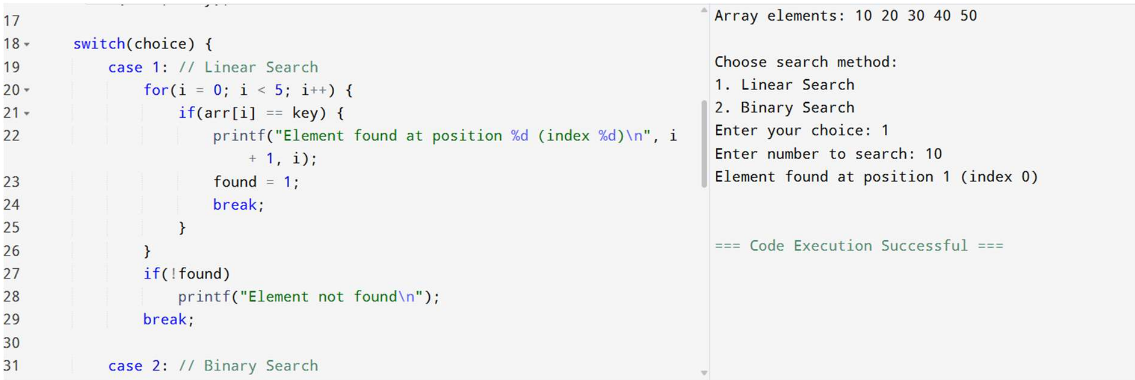
```
Enter number: 5
Factorial = 120


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

EX:17



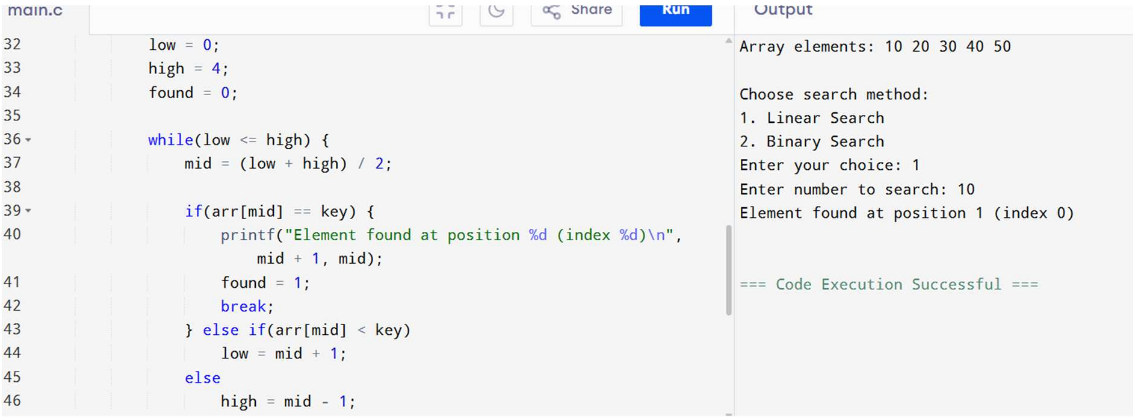
```c
#include <stdio.h>

int main() {
    int arr[5] = {10, 20, 30, 40, 50};
    int key, choice, i, found = 0;
    int low, high, mid;

    printf("Array elements: 10 20 30 40 50\n");
    printf("\nChoose search method:\n");
    printf("1. Linear Search\n");
    printf("2. Binary Search\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);

    printf("Enter number to search: ");
    scanf("%d", &key);
```

```
Array elements: 10 20 30 40 50

Choose search method:
1. Linear Search
2. Binary Search
Enter your choice: 1
Enter number to search: 10
Element found at position 1 (index 0)


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



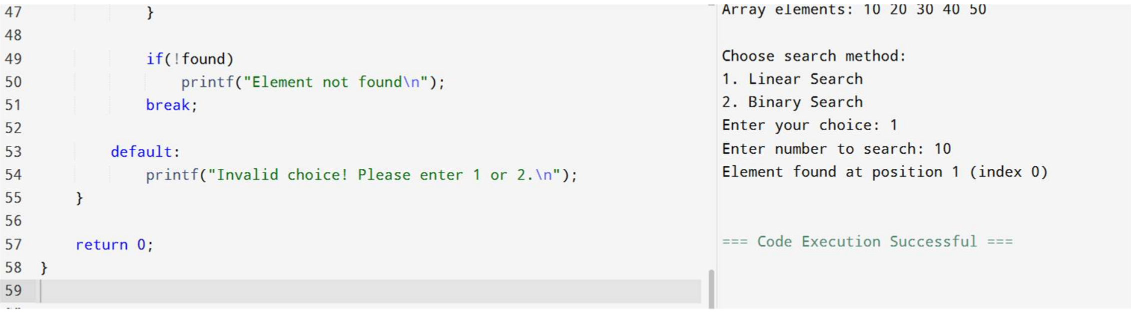
```c
    switch(choice) {
        case 1: // Linear Search
            for(i = 0; i < 5; i++) {
                if(arr[i] == key) {
                    printf("Element found at position %d (index %d)\n", i
                        + 1, i);
                    found = 1;
                    break;
                }
            }
            if(!found)
                printf("Element not found\n");
            break;

        case 2: // Binary Search
```

```
Array elements: 10 20 30 40 50

Choose search method:
1. Linear Search
2. Binary Search
Enter your choice: 1
Enter number to search: 10
Element found at position 1 (index 0)


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



```c
            low = 0;
            high = 4;
            found = 0;

            while(low <= high) {
                mid = (low + high) / 2;

                if(arr[mid] == key) {
                    printf("Element found at position %d (index %d)\n",
                        mid + 1, mid);
                    found = 1;
                    break;
                } else if(arr[mid] < key)
                    low = mid + 1;
                else
                    high = mid - 1;
```

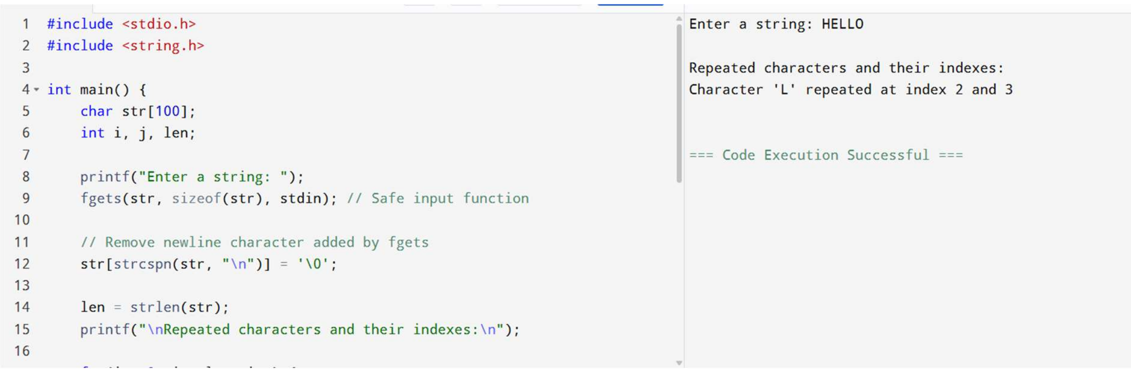
```
Array elements: 10 20 30 40 50

Choose search method:
1. Linear Search
2. Binary Search
Enter your choice: 1
Enter number to search: 10
Element found at position 1 (index 0)


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



```c
            }

            if(!found)
                printf("Element not found\n");
            break;

        default:
            printf("Invalid choice! Please enter 1 or 2.\n");
    }

    return 0;
}

```

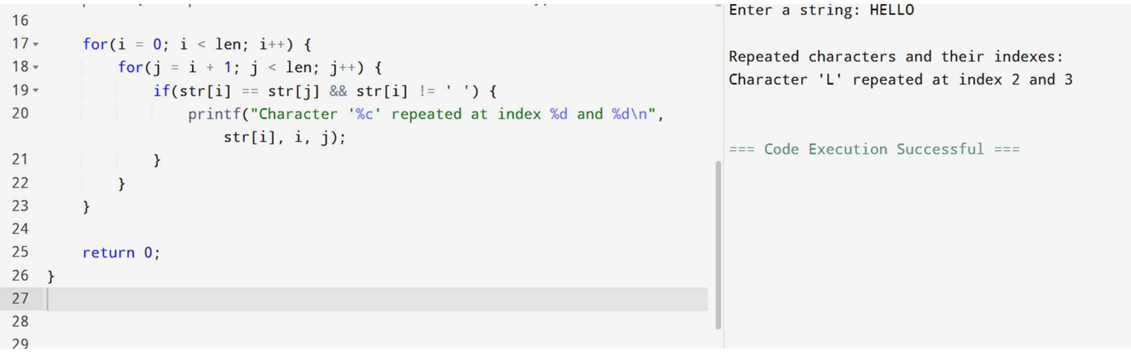
```
Array elements: 10 20 30 40 50

Choose search method:
1. Linear Search
2. Binary Search
Enter your choice: 1
Enter number to search: 10
Element found at position 1 (index 0)


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

EX:31



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

int main() {
    char str[100];
    int i, j, len;

    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin); // Safe input function

    // Remove newline character added by fgets
    str[strcspn(str, "\n")] = '\0';

    len = strlen(str);
    printf("\nRepeated characters and their indexes:\n");

```

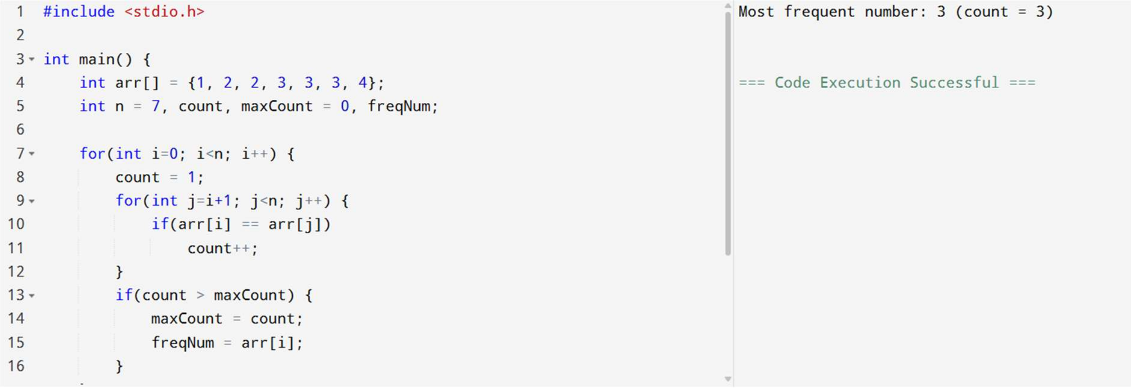
```
Enter a string: HELLO

Repeated characters and their indexes:
Character 'L' repeated at index 2 and 3


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



```c

    for(i = 0; i < len; i++) {
        for(j = i + 1; j < len; j++) {
            if(str[i] == str[j] && str[i] != ' ') {
                printf("Character '%c' repeated at index %d and %d\n",
                    str[i], i, j);
            }
        }
    }

    return 0;
}
```

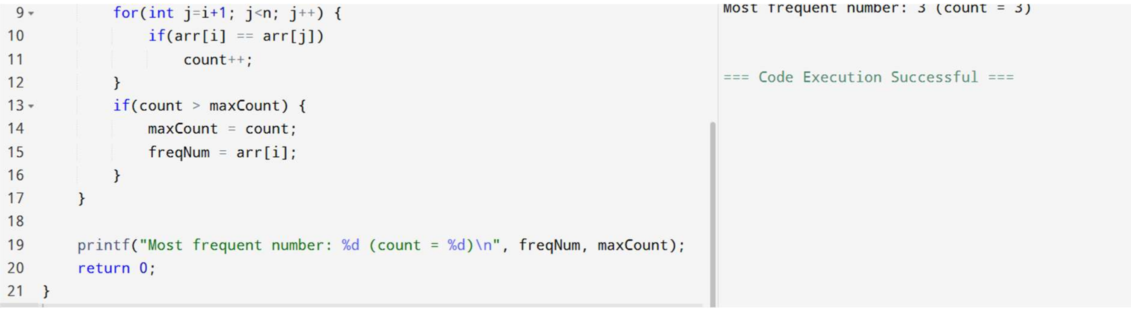
```
Enter a string: HELLO

Repeated characters and their indexes:
Character 'L' repeated at index 2 and 3


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

EX:32



```c
#include <stdio.h>

int main() {
    int arr[] = {1, 2, 2, 3, 3, 3, 4};
    int n = 7, count, maxCount = 0, freqNum;

    for(int i=0; i<n; i++) {
        count = 1;
        for(int j=i+1; j<n; j++) {
            if(arr[i] == arr[j])
                count++;
        }
        if(count > maxCount) {
            maxCount = count;
            freqNum = arr[i];
        }
```

```
Most frequent number: 3 (count = 3)


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

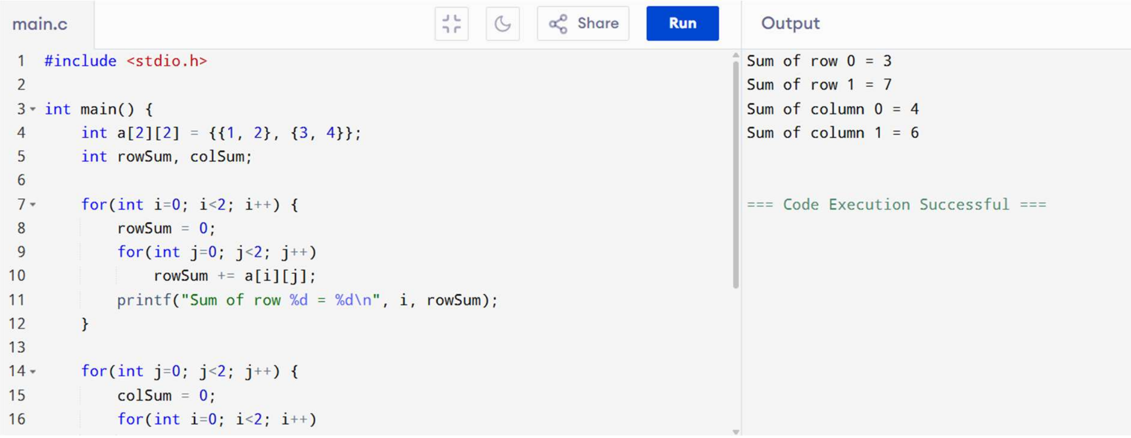


```c
        for(int j=i+1; j<n; j++) {
            if(arr[i] == arr[j])
                count++;
        }
        if(count > maxCount) {
            maxCount = count;
            freqNum = arr[i];
        }
```

```c
    }

    printf("Most frequent number: %d (count = %d)\n", freqNum, maxCount);
    return 0;
}
```

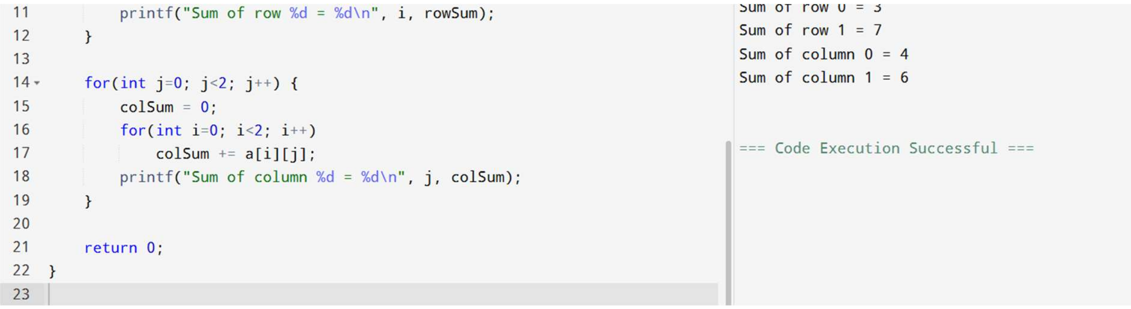
```
Most frequent number: 3 (count = 3)


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

EX:42



```c
#include <stdio.h>

int main() {
    int a[2][2] = {{1, 2}, {3, 4}};
    int rowSum, colSum;

    for(int i=0; i<2; i++) {
        rowSum = 0;
        for(int j=0; j<2; j++)
            rowSum += a[i][j];
        printf("Sum of row %d = %d\n", i, rowSum);
    }

    for(int j=0; j<2; j++) {
        colSum = 0;
        for(int i=0; i<2; i++)
```

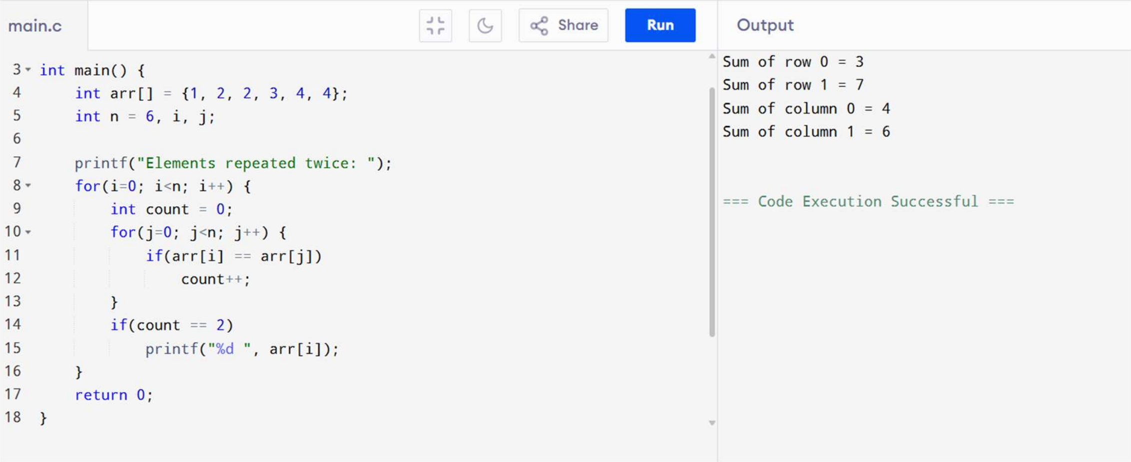
```
Sum of row 0 = 3
Sum of row 1 = 7
Sum of column 0 = 4
Sum of column 1 = 6


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



```c
        printf("Sum of row %d = %d\n", i, rowSum);
    }

    for(int j=0; j<2; j++) {
        colSum = 0;
        for(int i=0; i<2; i++)
            colSum += a[i][j];
        printf("Sum of column %d = %d\n", j, colSum);
    }

    return 0;
}

```

```
Sum of row 0 = 3
Sum of row 1 = 7
Sum of column 0 = 4
Sum of column 1 = 6


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

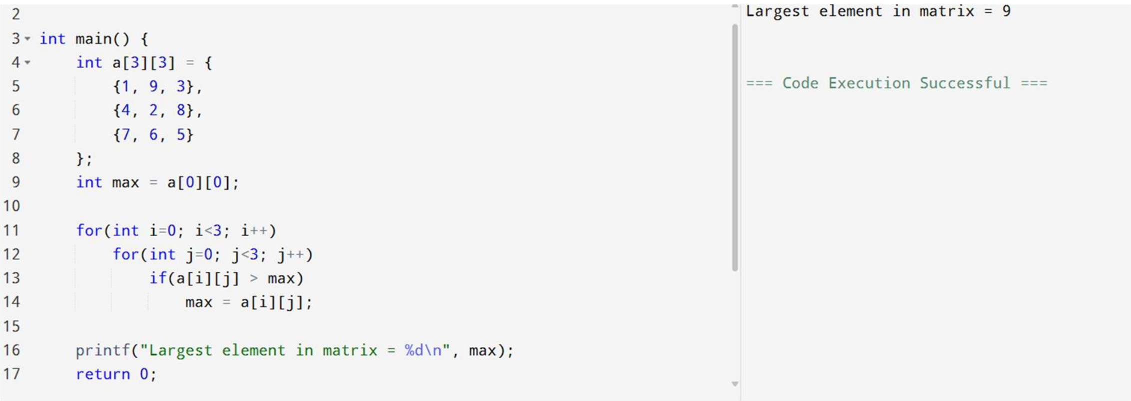
EX:43



```c
int main() {
    int arr[] = {1, 2, 2, 3, 4, 4};
    int n = 6, i, j;

    printf("Elements repeated twice: ");
    for(i=0; i<n; i++) {
        int count = 0;
        for(j=0; j<n; j++) {
            if(arr[i] == arr[j])
                count++;
        }
        if(count == 2)
            printf("%d ", arr[i]);
    }
    return 0;
}
```

```
Sum of row 0 = 3
Sum of row 1 = 7
Sum of column 0 = 4
Sum of column 1 = 6


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

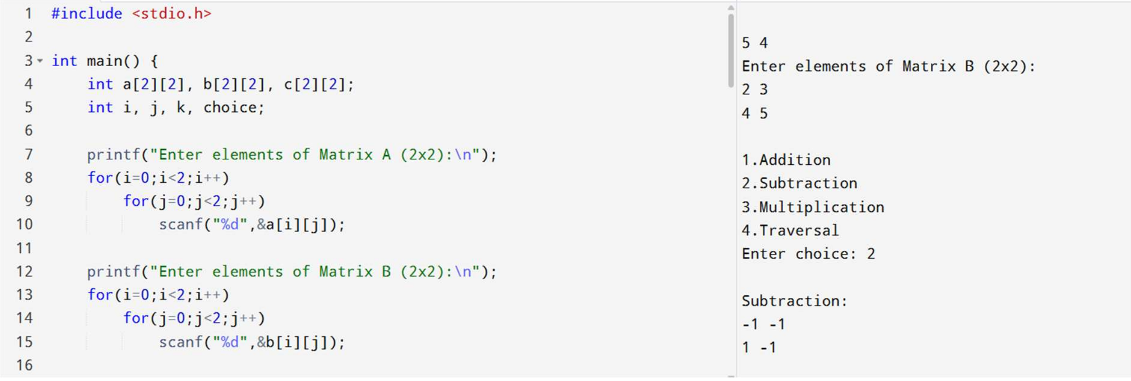
EX:48



```c

int main() {
    int a[3][3] = {
        {1, 9, 3},
        {4, 2, 8},
        {7, 6, 5}
    };
    int max = a[0][0];

    for(int i=0; i<3; i++)
        for(int j=0; j<3; j++)
            if(a[i][j] > max)
                max = a[i][j];

    printf("Largest element in matrix = %d\n", max);
    return 0;
```

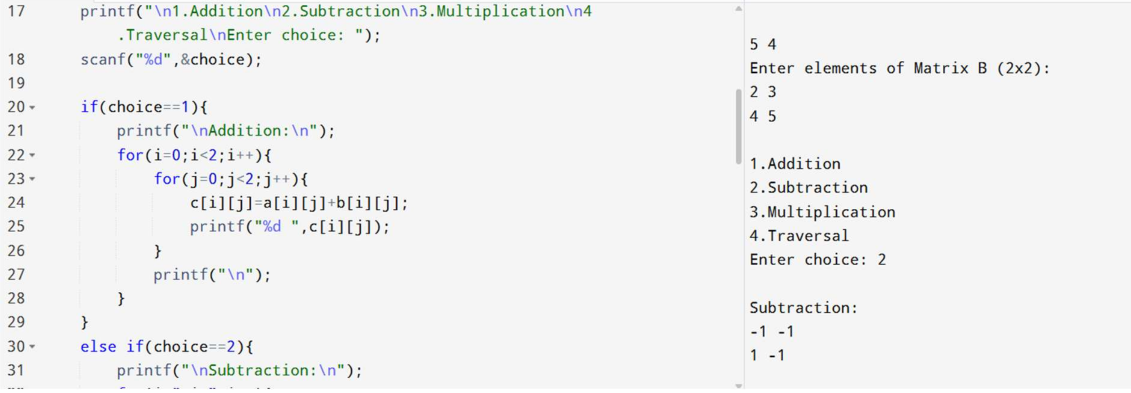
```
Largest element in matrix = 9


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

EX:



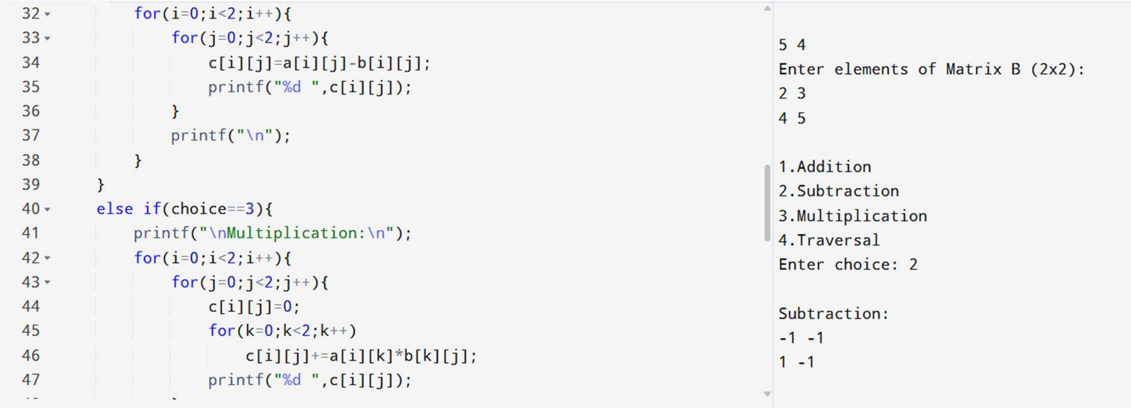
```c
#include <stdio.h>

int main() {
    int a[2][2], b[2][2], c[2][2];
    int i, j, k, choice;

    printf("Enter elements of Matrix A (2x2):\n");
    for(i=0;i<2;i++)
        for(j=0;j<2;j++)
            scanf("%d",&a[i][j]);

    printf("Enter elements of Matrix B (2x2):\n");
    for(i=0;i<2;i++)
        for(j=0;j<2;j++)
            scanf("%d",&b[i][j]);

```

```
5 4
Enter elements of Matrix B (2x2):
2 3
4 5

1.Addition
2.Subtraction
3.Multiplication
4.Traversal
Enter choice: 2

Subtraction:
-1 -1
1 -1
```



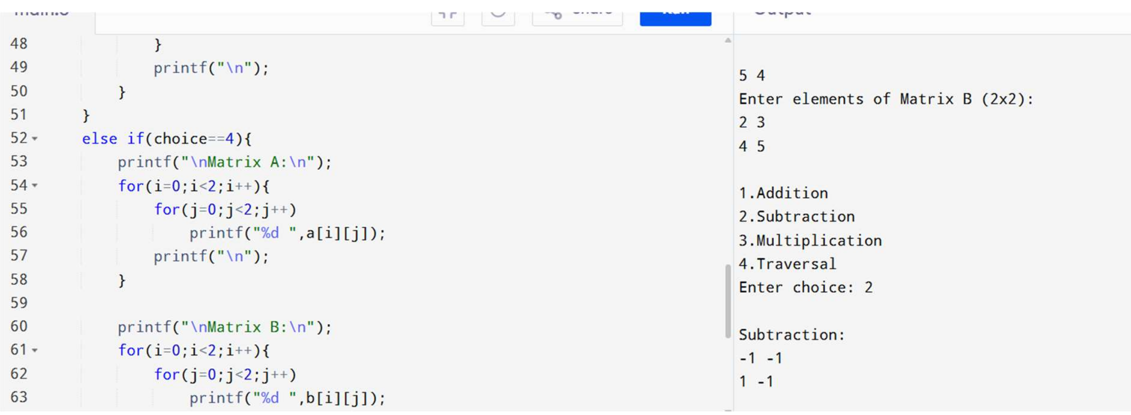
```c
    printf("\n1.Addition\n2.Subtraction\n3.Multiplication\n4.Traversal\nEnter choice: ");
    scanf("%d",&choice);

    if(choice==1){
        printf("\nAddition:\n");
        for(i=0;i<2;i++){
            for(j=0;j<2;j++){
                c[i][j]=a[i][j]+b[i][j];
                printf("%d ",c[i][j]);
            }
            printf("\n");
        }
    }
    else if(choice==2){
        printf("\nSubtraction:\n");
```

```
5 4
Enter elements of Matrix B (2x2):
2 3
4 5

1.Addition
2.Subtraction
3.Multiplication
4.Traversal
Enter choice: 2

Subtraction:
-1 -1
1 -1
```



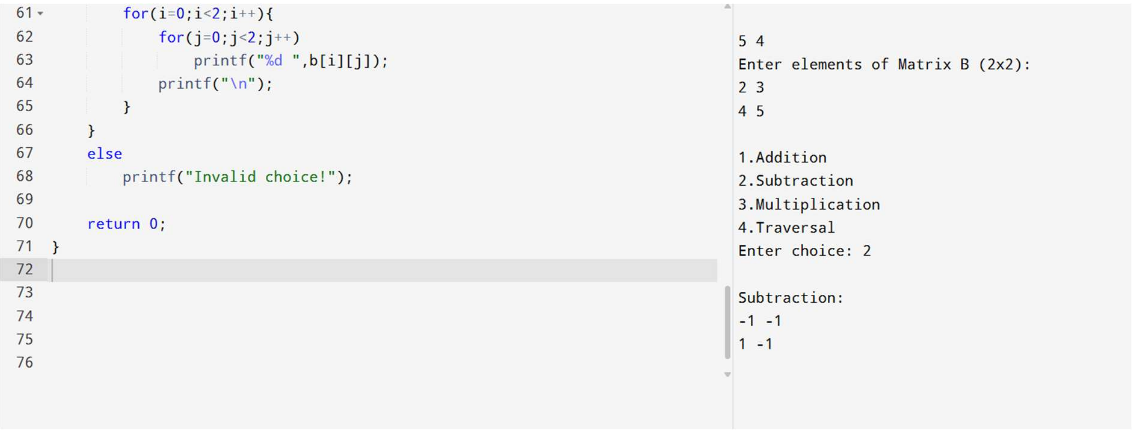
```c
        for(i=0;i<2;i++){
            for(j=0;j<2;j++){
                c[i][j]=a[i][j]-b[i][j];
                printf("%d ",c[i][j]);
            }
            printf("\n");
        }
    }
    else if(choice==3){
        printf("\nMultiplication:\n");
        for(i=0;i<2;i++){
            for(j=0;j<2;j++){
                c[i][j]=0;
                for(k=0;k<2;k++)
                    c[i][j]+=a[i][k]*b[k][j];
                printf("%d ",c[i][j]);
```

```
5 4
Enter elements of Matrix B (2x2):
2 3
4 5

1.Addition
2.Subtraction
3.Multiplication
4.Traversal
Enter choice: 2

Subtraction:
-1 -1
1 -1
```



```c
            }
            printf("\n");
        }
    }
    else if(choice==4){
        printf("\nMatrix A:\n");
        for(i=0;i<2;i++){
            for(j=0;j<2;j++)
                printf("%d ",a[i][j]);
            printf("\n");
        }

        printf("\nMatrix B:\n");
        for(i=0;i<2;i++){
            for(j=0;j<2;j++)
                printf("%d ",b[i][j]);
```

```
5 4
Enter elements of Matrix B (2x2):
2 3
4 5

1.Addition
2.Subtraction
3.Multiplication
4.Traversal
Enter choice: 2

Subtraction:
-1 -1
1 -1
```



```c
            for(i=0;i<2;i++){
                for(j=0;j<2;j++)
                    printf("%d ",b[i][j]);
                printf("\n");
            }
        }
        else
            printf("Invalid choice!");

        return 0;
    }
```

```
5 4
Enter elements of Matrix B (2x2):
2 3
4 5

1.Addition
2.Subtraction
3.Multiplication
4.Traversal
Enter choice: 2

Subtraction:
-1 -1
1 -1
```