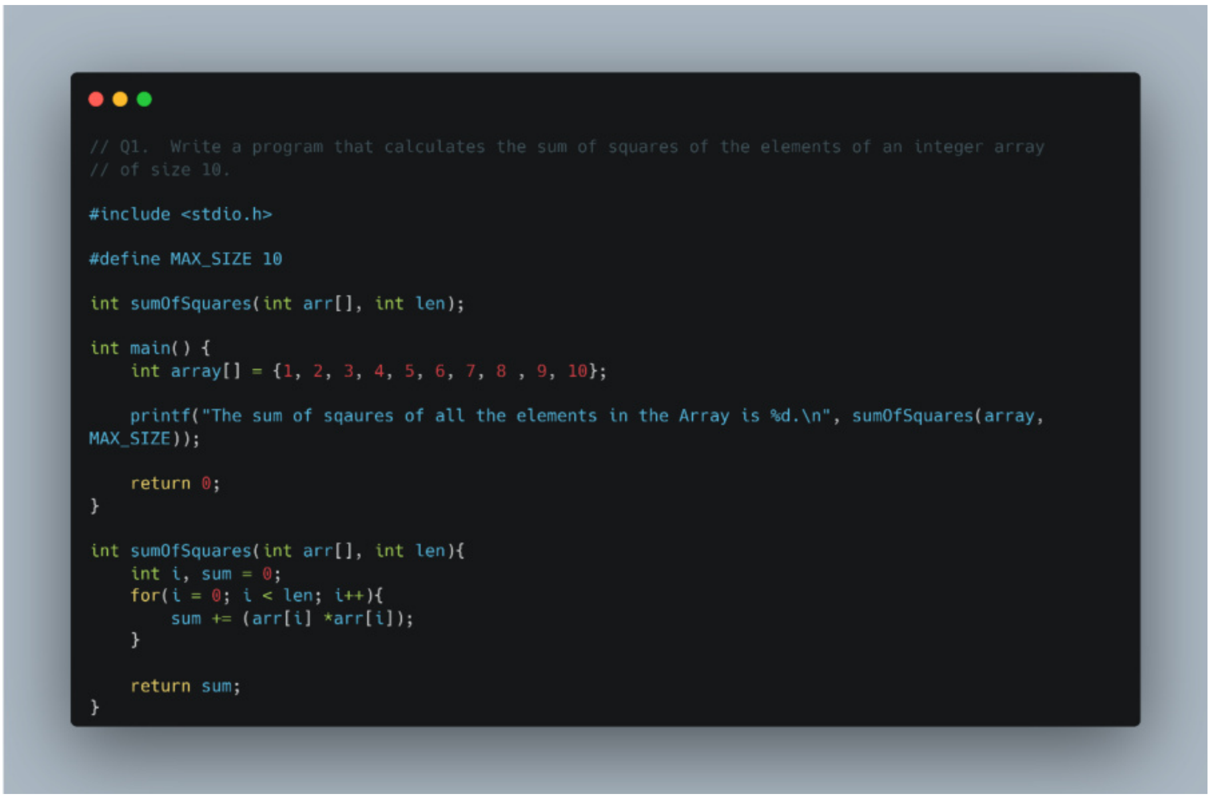
Name - Deshmukh Mehmood Rehan

MIS – 612303050

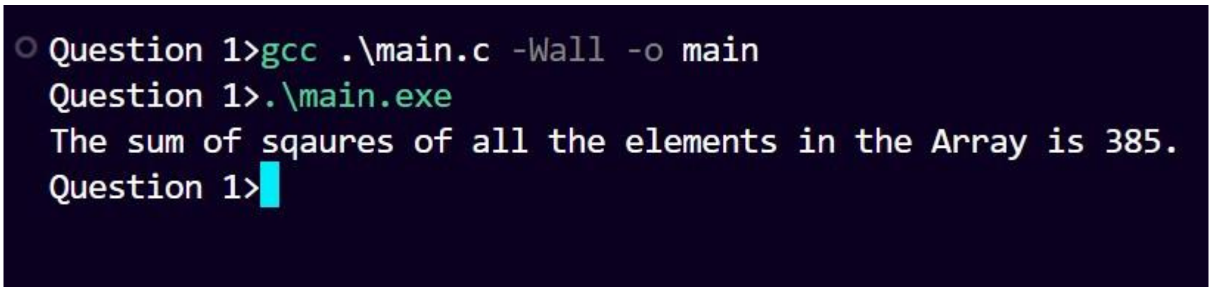
SY-Div 2

Question 1: Write a program that calculates the sum of squares of the elements of an integer array of size 10.



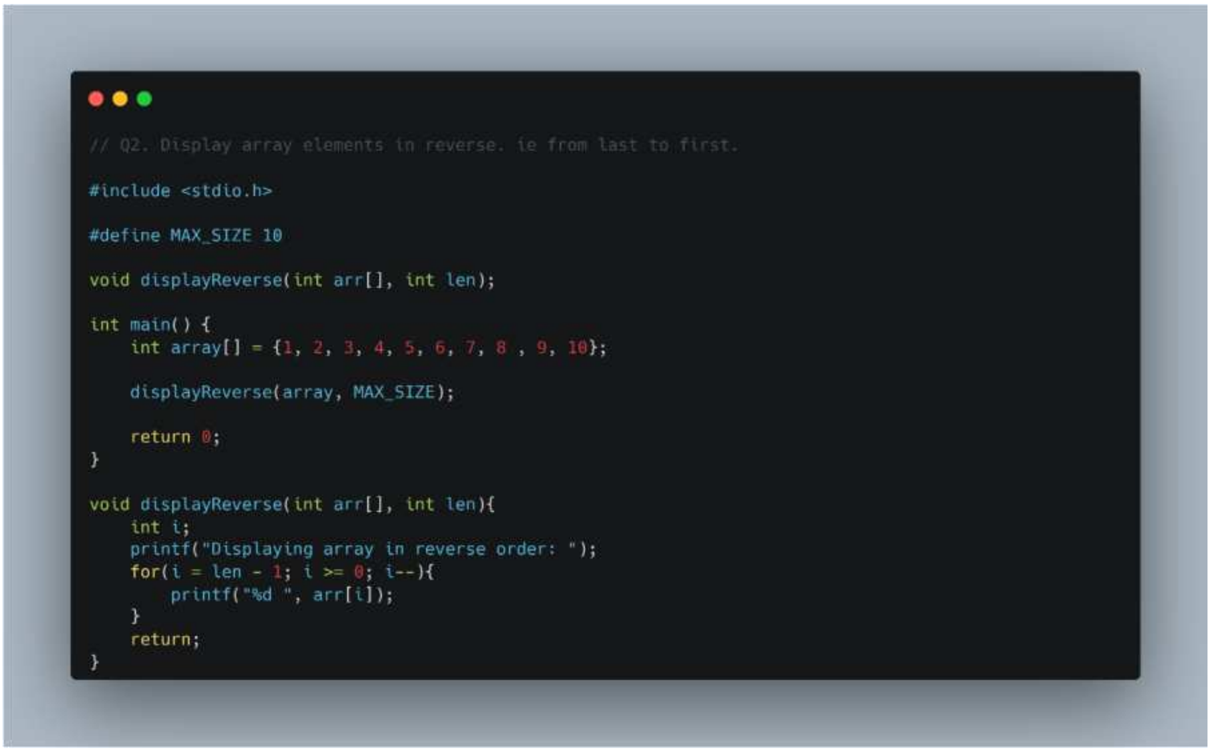
```c
// Q1.  Write a program that calculates the sum of squares of the elements of an integer array
// of size 10.

#include <stdio.h>

#define MAX_SIZE 10

int sumOfSquares(int arr[], int len);

int main() {
    int array[] = {1, 2, 3, 4, 5, 6, 7, 8 , 9, 10};

    printf("The sum of sqaures of all the elements in the Array is %d.\n", sumOfSquares(array,
MAX_SIZE));

    return 0;
}

int sumOfSquares(int arr[], int len){
    int i, sum = 0;
    for(i = 0; i < len; i++){
        sum += (arr[i] *arr[i]);
    }

    return sum;
}
```

Output:



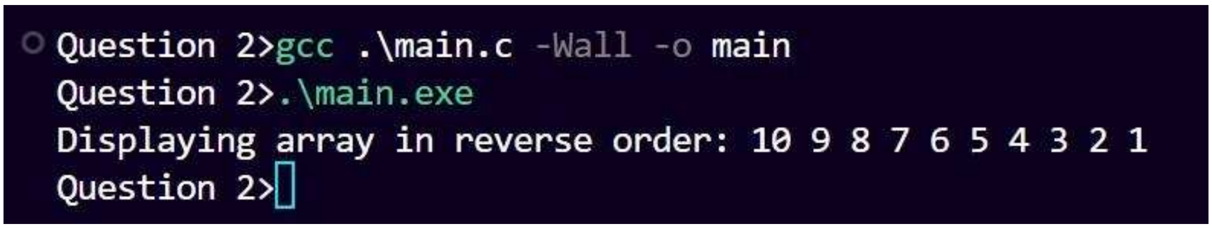
```
Question 1>gcc .\main.c -Wall -o main
Question 1>.\main.exe
The sum of sqaures of all the elements in the Array is 385.
Question 1>
```

Question 2: Display array elements in reverse. ie from last to first.



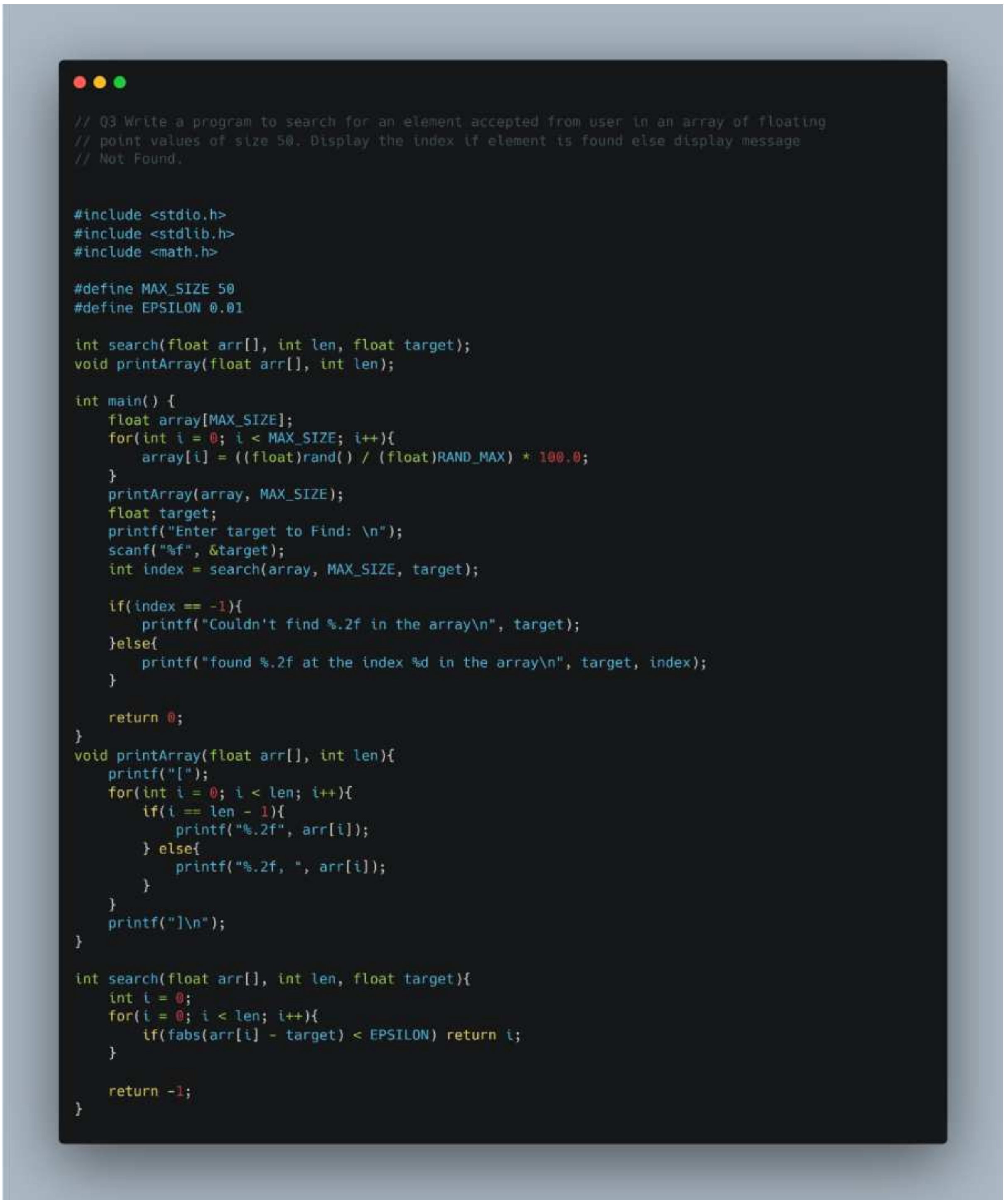
```c
// Q2. Display array elements in reverse. ie from last to first.

#include <stdio.h>

#define MAX_SIZE 10

void displayReverse(int arr[], int len);

int main() {
    int array[] = {1, 2, 3, 4, 5, 6, 7, 8 , 9, 10};

    displayReverse(array, MAX_SIZE);

    return 0;
}

void displayReverse(int arr[], int len){
    int i;
    printf("Displaying array in reverse order: ");
    for(i = len - 1; i >= 0; i--){
        printf("%d ", arr[i]);
    }
    return;
}
```

Output:



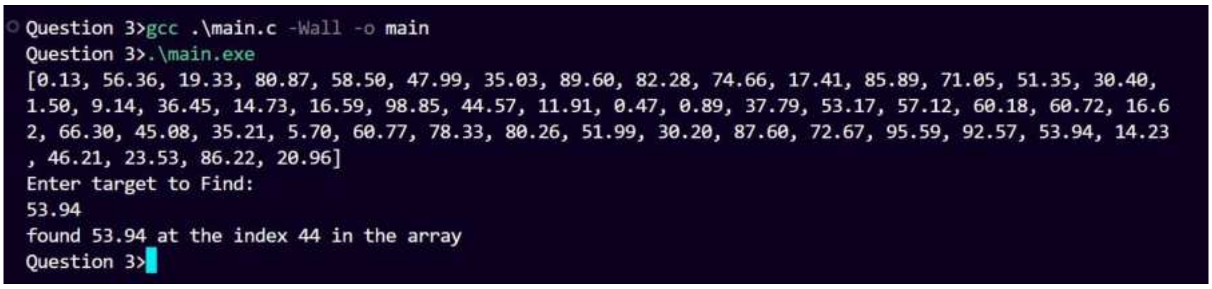
```
Question 2>gcc .\main.c -Wall -o main
Question 2>.\main.exe
Displaying array in reverse order: 10 9 8 7 6 5 4 3 2 1
Question 2>
```

Question 3: Write a program to search for an element accepted from user in an array of floatingpoint values of size 50. Display the index if element is found else display message Not Found



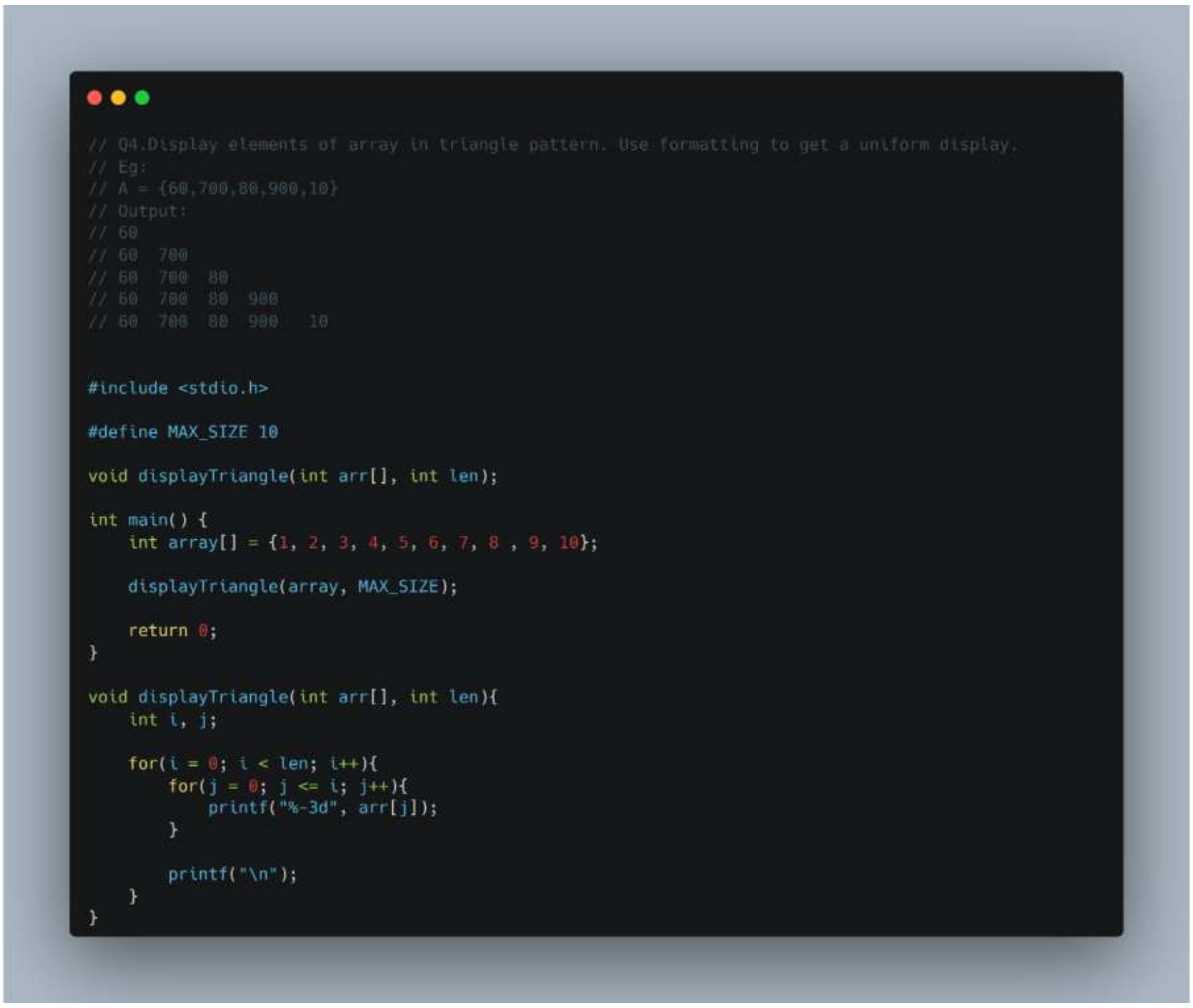
```c
// Q3 Write a program to search for an element accepted from user in an array of floating
// point values of size 50. Display the index if element is found else display message
// Not Found.


#include <stdio.h>
#include <stdlib.h>
#include <math.h>

#define MAX_SIZE 50
#define EPSILON 0.01

int search(float arr[], int len, float target);
void printArray(float arr[], int len);

int main() {
    float array[MAX_SIZE];
    for(int i = 0; i < MAX_SIZE; i++){
        array[i] = ((float)rand() / (float)RAND_MAX) * 100.0;
    }
    printArray(array, MAX_SIZE);
    float target;
    printf("Enter target to Find: \n");
    scanf("%f", &target);
    int index = search(array, MAX_SIZE, target);

    if(index == -1){
        printf("Couldn't find %.2f in the array\n", target);
    }else{
        printf("found %.2f at the index %d in the array\n", target, index);
    }

    return 0;
}
void printArray(float arr[], int len){
    printf("[");
    for(int i = 0; i < len; i++){
        if(i == len - 1){
            printf("%.2f", arr[i]);
        } else{
            printf("%.2f, ", arr[i]);
        }
    }
    printf("]\n");
}

int search(float arr[], int len, float target){
    int i = 0;
    for(i = 0; i < len; i++){
        if(fabs(arr[i] - target) < EPSILON) return i;
    }

    return -1;
}
```

Output



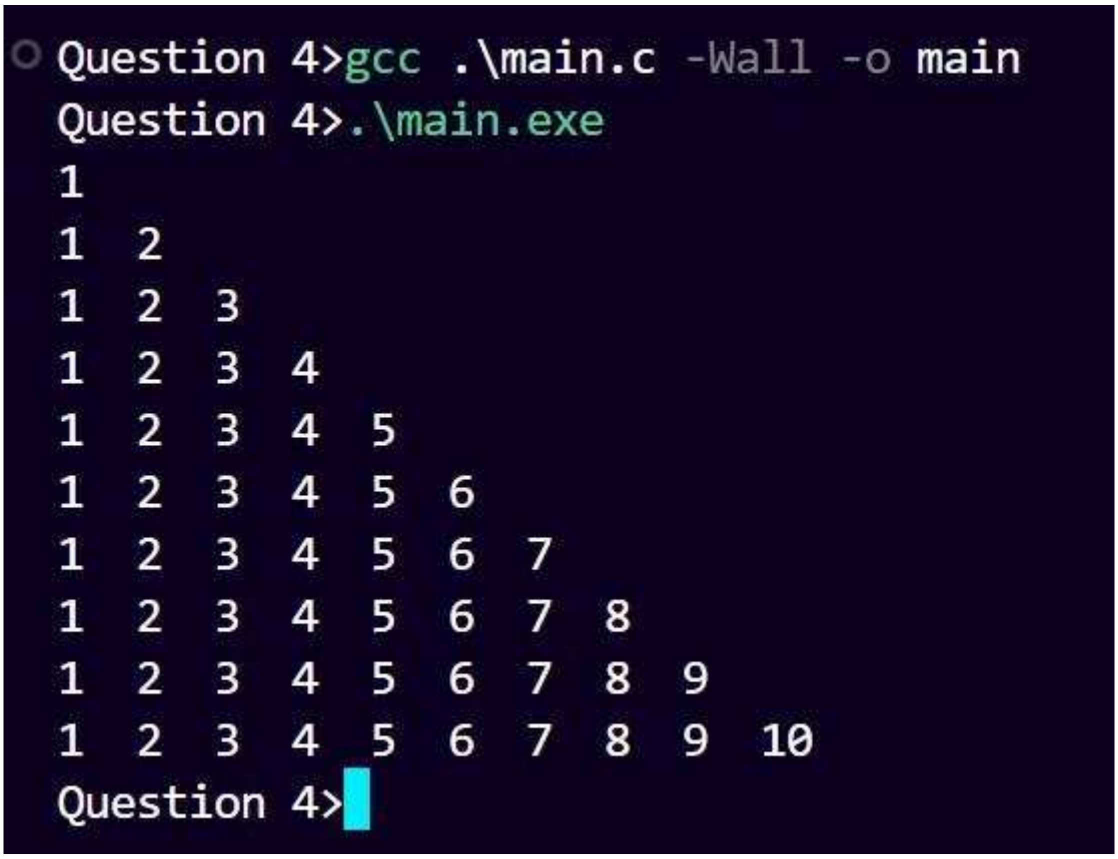
```
Question 3>gcc .\main.c -Wall -o main
Question 3>.\main.exe
[0.13, 56.36, 19.33, 80.87, 58.50, 47.99, 35.03, 89.60, 82.28, 74.66, 17.41, 85.89, 71.05, 51.35, 30.40,
1.50, 9.14, 36.45, 14.73, 16.59, 98.85, 44.57, 11.91, 0.47, 0.89, 37.79, 53.17, 57.12, 60.18, 60.72, 16.6
2, 66.30, 45.08, 35.21, 5.70, 60.77, 78.33, 80.26, 51.99, 30.20, 87.60, 72.67, 95.59, 92.57, 53.94, 14.23
, 46.21, 23.53, 86.22, 20.96]
Enter target to Find:
53.94
found 53.94 at the index 44 in the array
Question 3>
```

Question 4: Display elements of array in triangle pattern. Use formatting to get a uniform display. Eg: A = {60,700,80,900,10} Output: 60 60 700 60 700 80 60 700 80 900 60 700 80 900 10



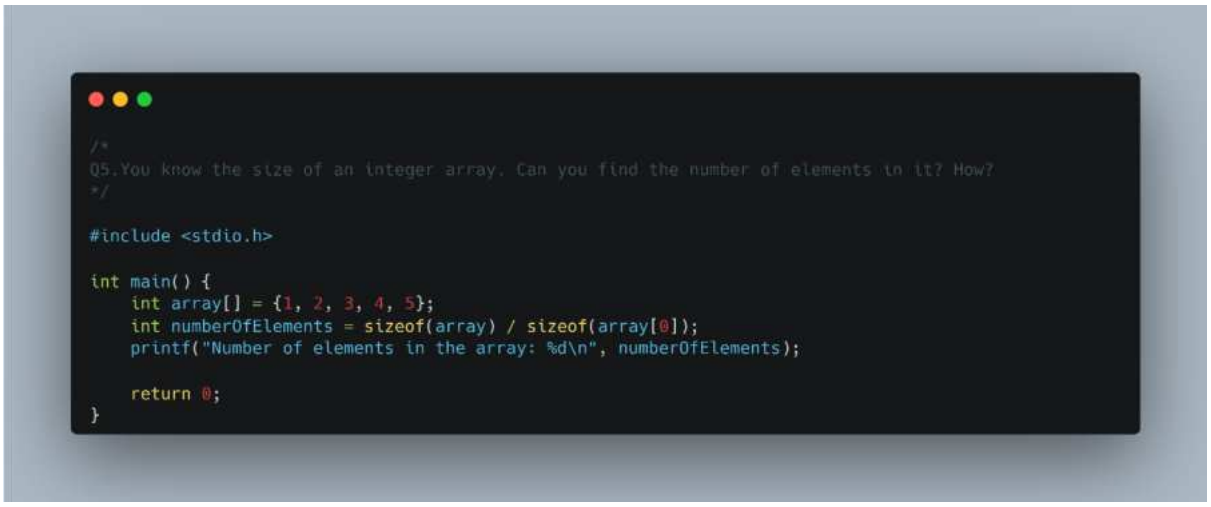
```c
// Q4.Display elements of array in triangle pattern. Use formatting to get a uniform display.
// Eg:
// A = {60,700,80,900,10}
// Output:
// 60
// 60  700
// 60  700  80
// 60  700  80  900
// 60  700  80  900   10


#include <stdio.h>

#define MAX_SIZE 10

void displayTriangle(int arr[], int len);

int main() {
    int array[] = {1, 2, 3, 4, 5, 6, 7, 8 , 9, 10};

    displayTriangle(array, MAX_SIZE);

    return 0;
}

void displayTriangle(int arr[], int len){
    int i, j;

    for(i = 0; i < len; i++){
        for(j = 0; j <= i; j++){
            printf("%-3d", arr[j]);
        }

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

Output



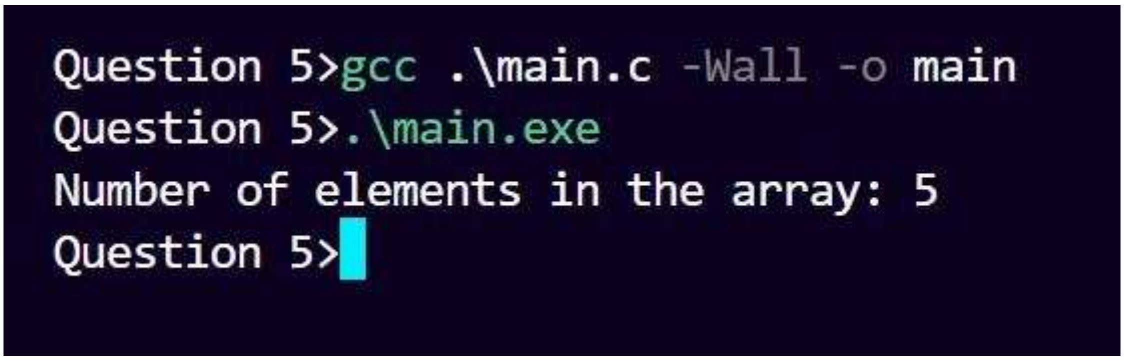
```
Question 4>gcc .\main.c -Wall -o main
Question 4>.\main.exe
1
1  2
1  2  3
1  2  3  4
1  2  3  4  5
1  2  3  4  5  6
1  2  3  4  5  6  7
1  2  3  4  5  6  7  8
1  2  3  4  5  6  7  8  9
1  2  3  4  5  6  7  8  9  10
Question 4>
```

Question 5: You know size of integer array. Can you find number of elements in it? How?



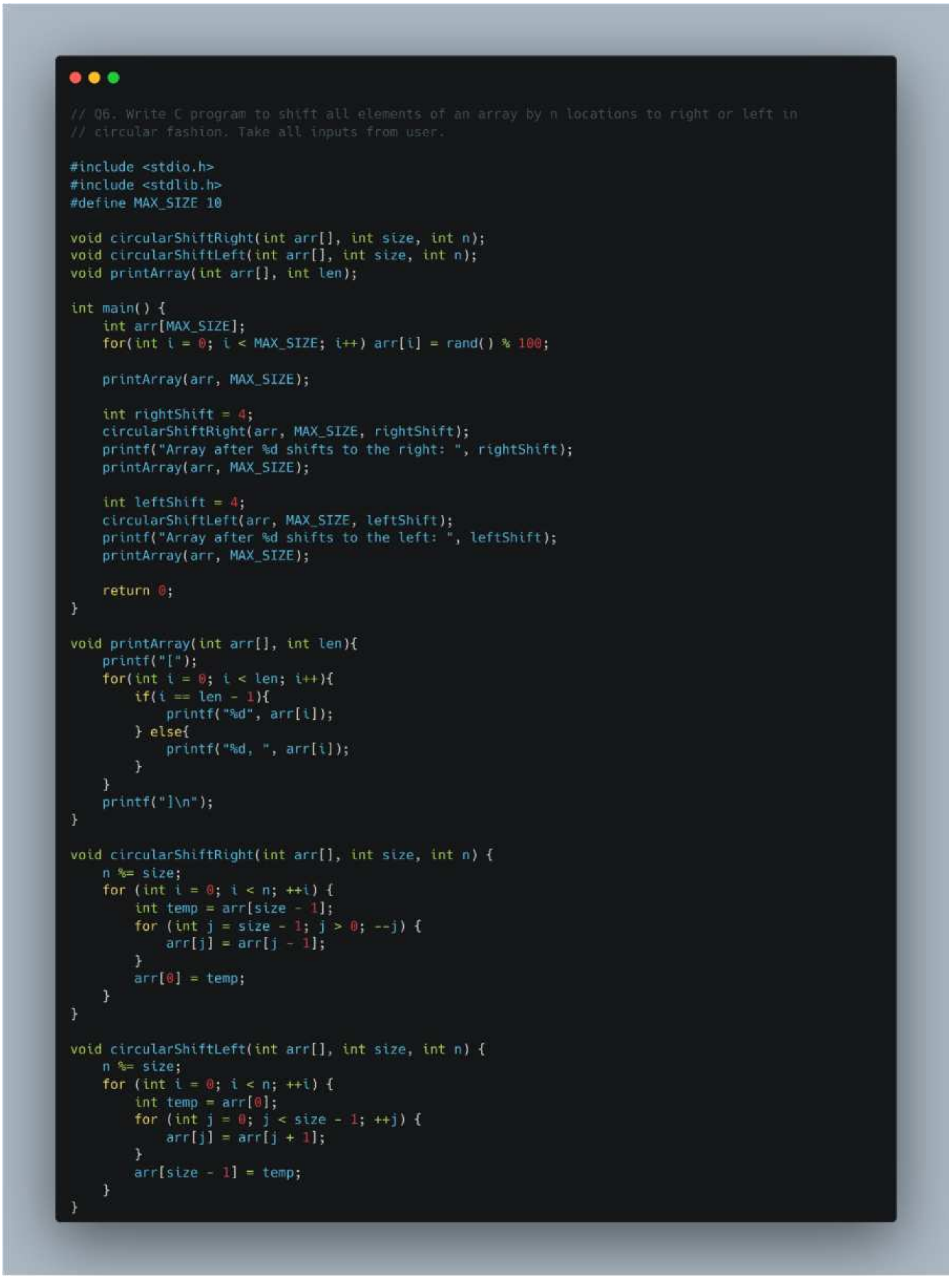
```c
/*
Q5.You know the size of an integer array. Can you find the number of elements in it? How?
*/

#include <stdio.h>

int main() {
    int array[] = {1, 2, 3, 4, 5};
    int numberOfElements = sizeof(array) / sizeof(array[0]);
    printf("Number of elements in the array: %d\n", numberOfElements);

    return 0;
}
```

Output



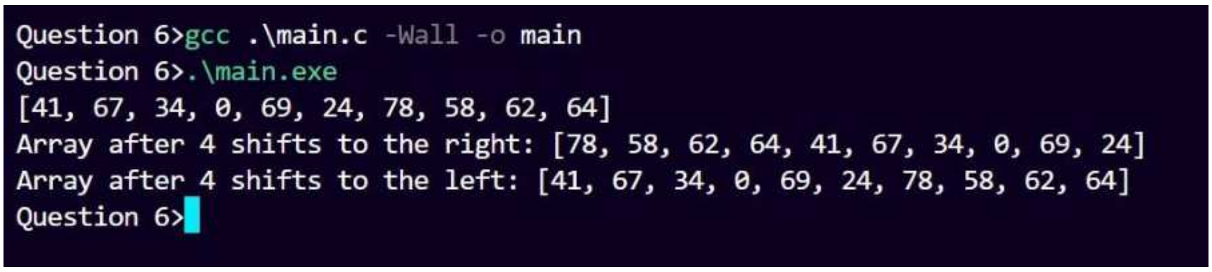
```
Question 5>gcc .\main.c -Wall -o main
Question 5>.\main.exe
Number of elements in the array: 5
Question 5>
```

Question 6: Write C program to shift all elements of an array by n locations to right or left in circular fashion. Take all inputs from user.



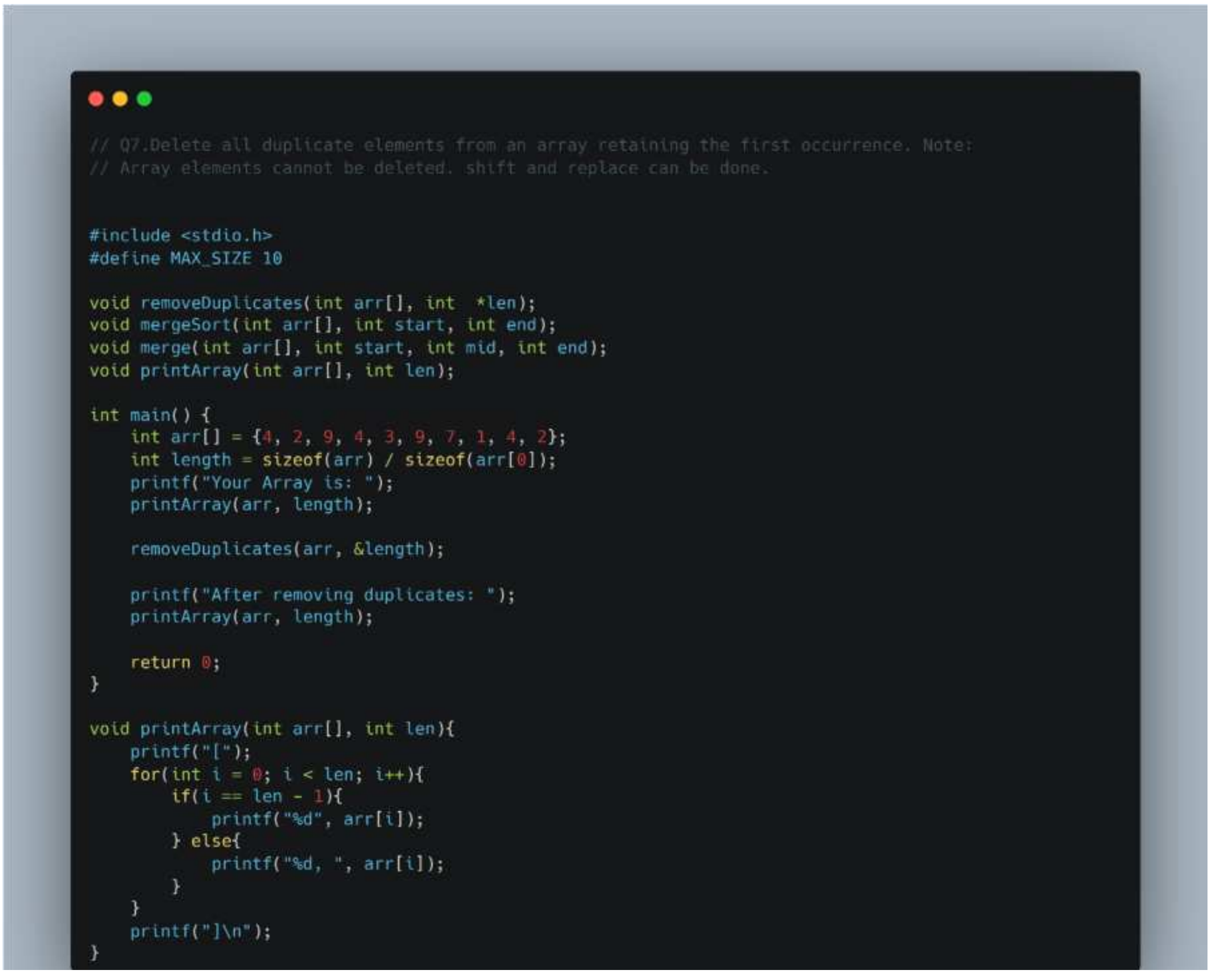
```c
// Q6. Write C program to shift all elements of an array by n locations to right or left in
// circular fashion. Take all inputs from user.

#include <stdio.h>
#include <stdlib.h>
#define MAX_SIZE 10

void circularShiftRight(int arr[], int size, int n);
void circularShiftLeft(int arr[], int size, int n);
void printArray(int arr[], int len);

int main() {
    int arr[MAX_SIZE];
    for(int i = 0; i < MAX_SIZE; i++) arr[i] = rand() % 100;

    printArray(arr, MAX_SIZE);

    int rightShift = 4;
    circularShiftRight(arr, MAX_SIZE, rightShift);
    printf("Array after %d shifts to the right: ", rightShift);
    printArray(arr, MAX_SIZE);

    int leftShift = 4;
    circularShiftLeft(arr, MAX_SIZE, leftShift);
    printf("Array after %d shifts to the left: ", leftShift);
    printArray(arr, MAX_SIZE);

    return 0;
}

void printArray(int arr[], int len){
    printf("[");
    for(int i = 0; i < len; i++){
        if(i == len - 1){
            printf("%d", arr[i]);
        } else{
            printf("%d, ", arr[i]);
        }
    }
    printf("]\n");
}

void circularShiftRight(int arr[], int size, int n) {
    n %= size;
    for (int i = 0; i < n; ++i) {
        int temp = arr[size - 1];
        for (int j = size - 1; j > 0; --j) {
            arr[j] = arr[j - 1];
        }
        arr[0] = temp;
    }
}

void circularShiftLeft(int arr[], int size, int n) {
    n %= size;
    for (int i = 0; i < n; ++i) {
        int temp = arr[0];
        for (int j = 0; j < size - 1; ++j) {
            arr[j] = arr[j + 1];
        }
        arr[size - 1] = temp;
    }
}
```

Output

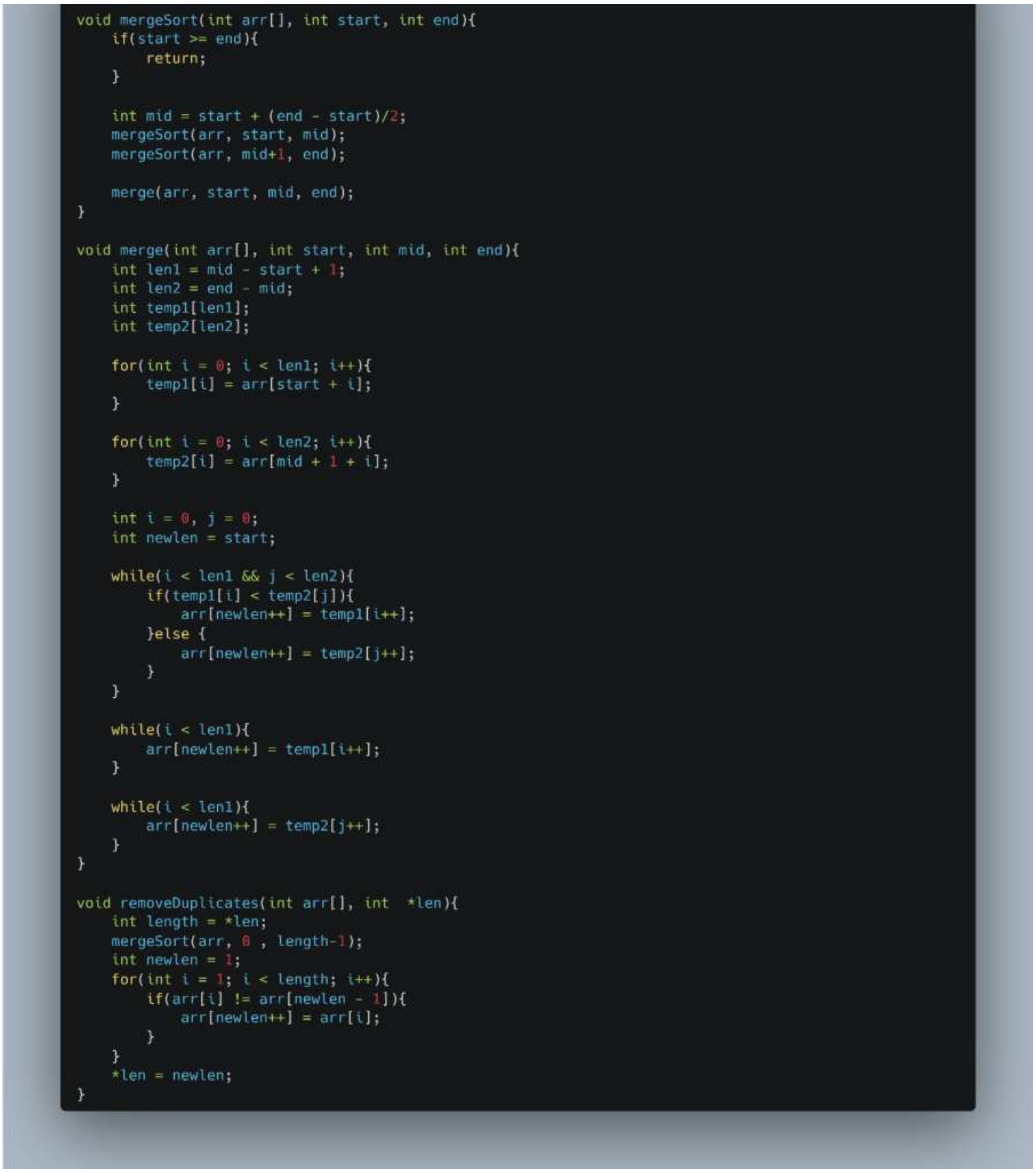


```
Question 6>gcc .\main.c -Wall -o main
Question 6>.\main.exe
[41, 67, 34, 0, 69, 24, 78, 58, 62, 64]
Array after 4 shifts to the right: [78, 58, 62, 64, 41, 67, 34, 0, 69, 24]
Array after 4 shifts to the left: [41, 67, 34, 0, 69, 24, 78, 58, 62, 64]
Question 6>
```

Question 7: Delete all duplicate elements from an array retaining the first occurrence. Note: Array elements cannot be deleted. shift and replace can be done.

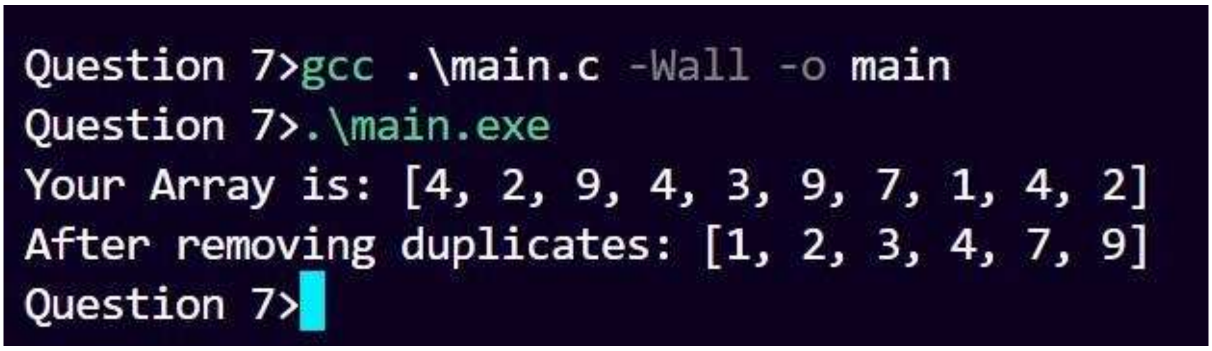


```c
// Q7.Delete all duplicate elements from an array retaining the first occurrence. Note:
// Array elements cannot be deleted. shift and replace can be done.


#include <stdio.h>
#define MAX_SIZE 10

void removeDuplicates(int arr[], int  *len);
void mergeSort(int arr[], int start, int end);
void merge(int arr[], int start, int mid, int end);
void printArray(int arr[], int len);

int main() {
    int arr[] = {4, 2, 9, 4, 3, 9, 7, 1, 4, 2};
    int length = sizeof(arr) / sizeof(arr[0]);
    printf("Your Array is: ");
    printArray(arr, length);

    removeDuplicates(arr, &length);

    printf("After removing duplicates: ");
    printArray(arr, length);

    return 0;
}

void printArray(int arr[], int len){
    printf("[");
    for(int i = 0; i < len; i++){
        if(i == len - 1){
            printf("%d", arr[i]);
        } else{
            printf("%d, ", arr[i]);
        }
    }
    printf("]\n");
}
```



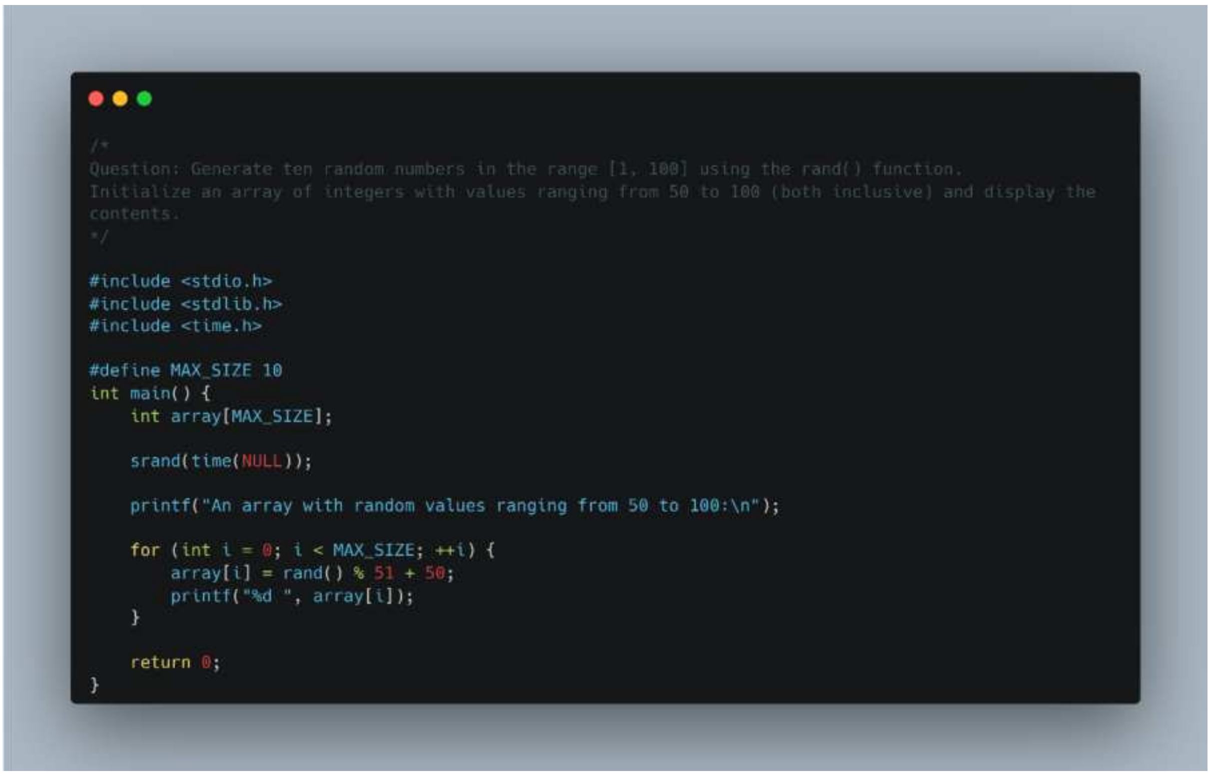
```c
void mergeSort(int arr[], int start, int end){
    if(start >= end){
        return;
    }

    int mid = start + (end - start)/2;
    mergeSort(arr, start, mid);
    mergeSort(arr, mid+1, end);

    merge(arr, start, mid, end);
}

void merge(int arr[], int start, int mid, int end){
    int len1 = mid - start + 1;
    int len2 = end - mid;
    int temp1[len1];
    int temp2[len2];

    for(int i = 0; i < len1; i++){
        temp1[i] = arr[start + i];
    }

    for(int i = 0; i < len2; i++){
        temp2[i] = arr[mid + 1 + i];
    }

    int i = 0, j = 0;
    int newlen = start;

    while(i < len1 && j < len2){
        if(temp1[i] < temp2[j]){
            arr[newlen++] = temp1[i++];
        }else {
            arr[newlen++] = temp2[j++];
        }
    }

    while(i < len1){
        arr[newlen++] = temp1[i++];
    }

    while(i < len1){
        arr[newlen++] = temp2[j++];
    }
}

void removeDuplicates(int arr[], int  *len){
    int length = *len;
    mergeSort(arr, 0 , length-1);
    int newlen = 1;
    for(int i = 1; i < length; i++){
        if(arr[i] != arr[newlen - 1]){
            arr[newlen++] = arr[i];
        }
    }
    *len = newlen;
}
```

Output



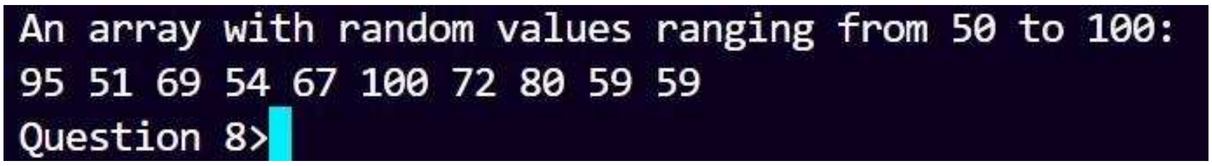
```
Question 7>gcc .\main.c -Wall -o main
Question 7>.\main.exe
Your Array is: [4, 2, 9, 4, 3, 9, 7, 1, 4, 2]
After removing duplicates: [1, 2, 3, 4, 7, 9]
Question 7>
```

Question 8: Initialize array of integers with values ranging 50 – 100 both inclusive and display the contents.



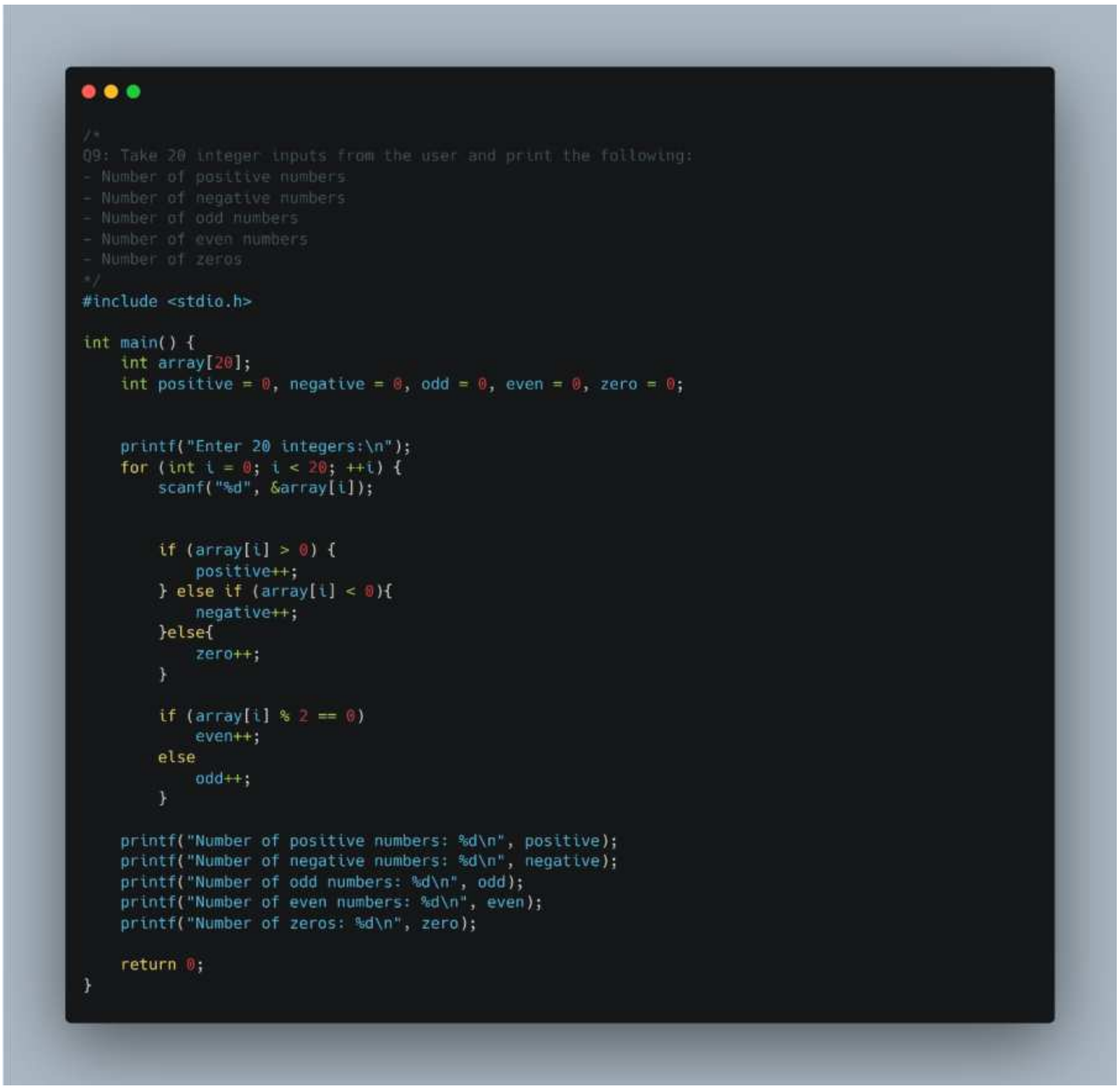
```c
/*
Question: Generate ten random numbers in the range [1, 100] using the rand() function.
Initialize an array of integers with values ranging from 50 to 100 (both inclusive) and display the
contents.
*/

#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define MAX_SIZE 10
int main() {
    int array[MAX_SIZE];

    srand(time(NULL));

    printf("An array with random values ranging from 50 to 100:\n");

    for (int i = 0; i < MAX_SIZE; ++i) {
        array[i] = rand() % 51 + 50;
        printf("%d ", array[i]);
    }

    return 0;
}
```

Output



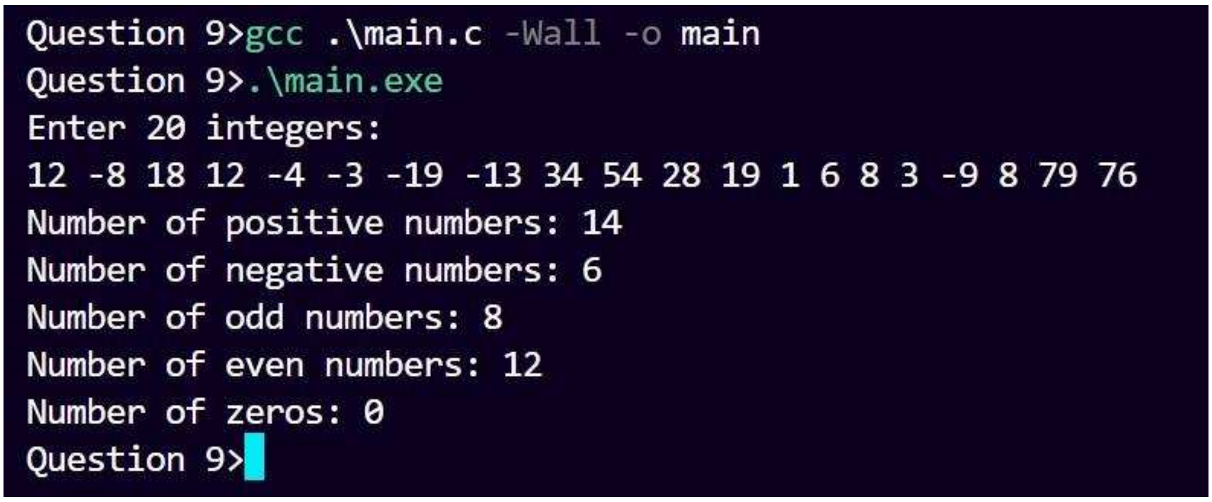
```
An array with random values ranging from 50 to 100:
95 51 69 54 67 100 72 80 59 59
Question 8>
```

Question 9: Take 20 integer inputs from user and print the following: number of positive numbers, number of negative numbers, number of odd numbers, number of even numbers, number of 0



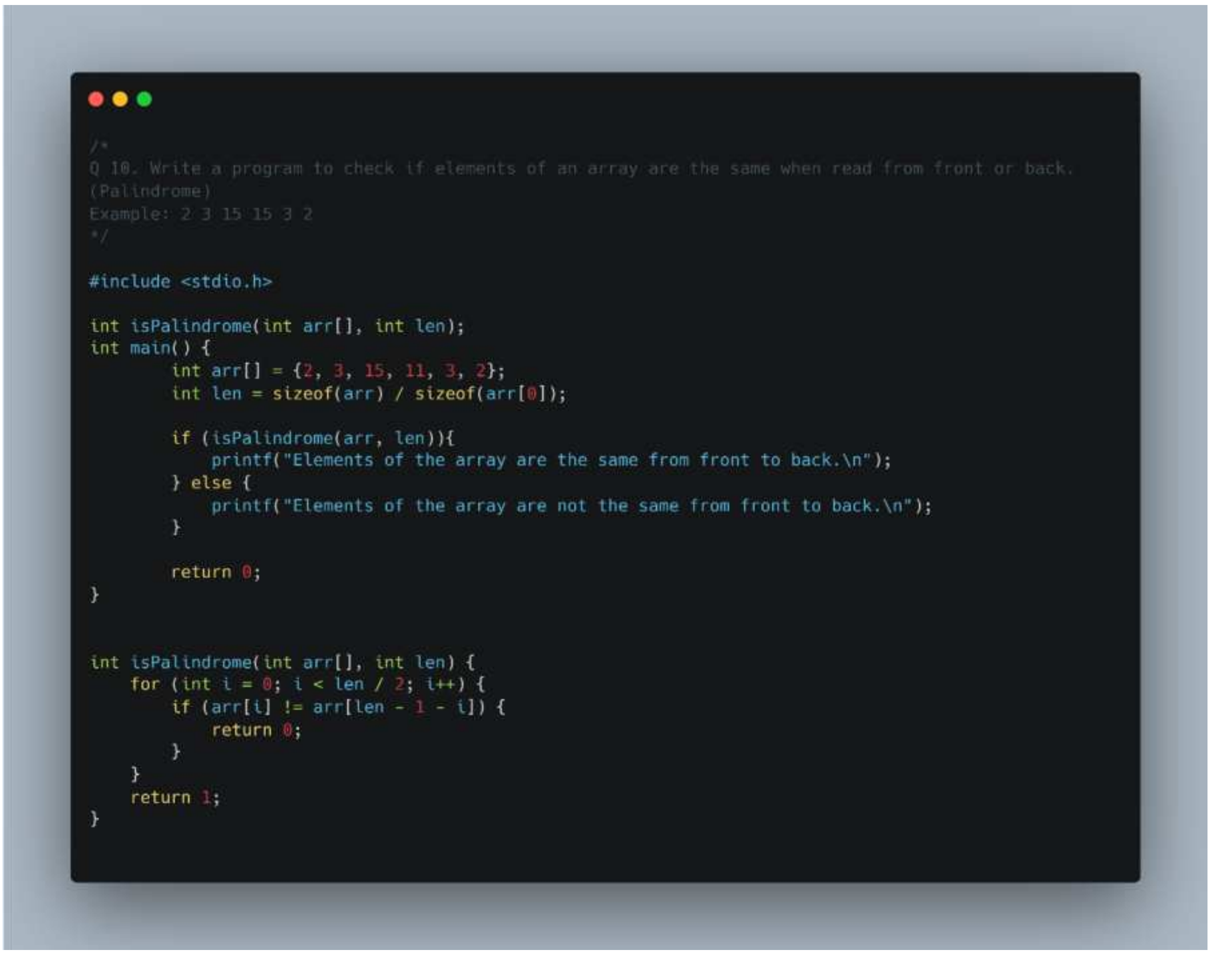
```c
/*
Q9: Take 20 integer inputs from the user and print the following:
- Number of positive numbers
- Number of negative numbers
- Number of odd numbers
- Number of even numbers
- Number of zeros
*/
#include <stdio.h>

int main() {
    int array[20];
    int positive = 0, negative = 0, odd = 0, even = 0, zero = 0;


    printf("Enter 20 integers:\n");
    for (int i = 0; i < 20; ++i) {
        scanf("%d", &array[i]);


        if (array[i] > 0) {
            positive++;
        } else if (array[i] < 0){
            negative++;
        }else{
            zero++;
        }

        if (array[i] % 2 == 0)
            even++;
        else
            odd++;
        }

    printf("Number of positive numbers: %d\n", positive);
    printf("Number of negative numbers: %d\n", negative);
    printf("Number of odd numbers: %d\n", odd);
    printf("Number of even numbers: %d\n", even);
    printf("Number of zeros: %d\n", zero);

    return 0;
}
```

Output:



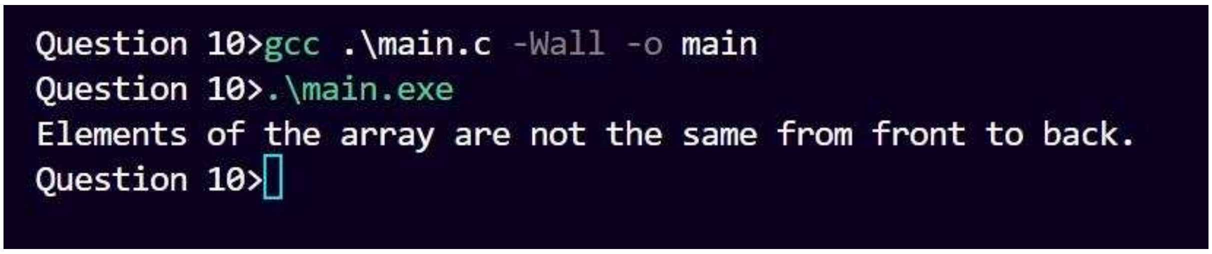
```
Question 9>gcc .\main.c -Wall -o main
Question 9>.\main.exe
Enter 20 integers:
12 -8 18 12 -4 -3 -19 -13 34 54 28 19 1 6 8 3 -9 8 79 76
Number of positive numbers: 14
Number of negative numbers: 6
Number of odd numbers: 8
Number of even numbers: 12
Number of zeros: 0
Question 9>
```

Question 10: Write a program to check if elements of an array are same or not it read from front or back.



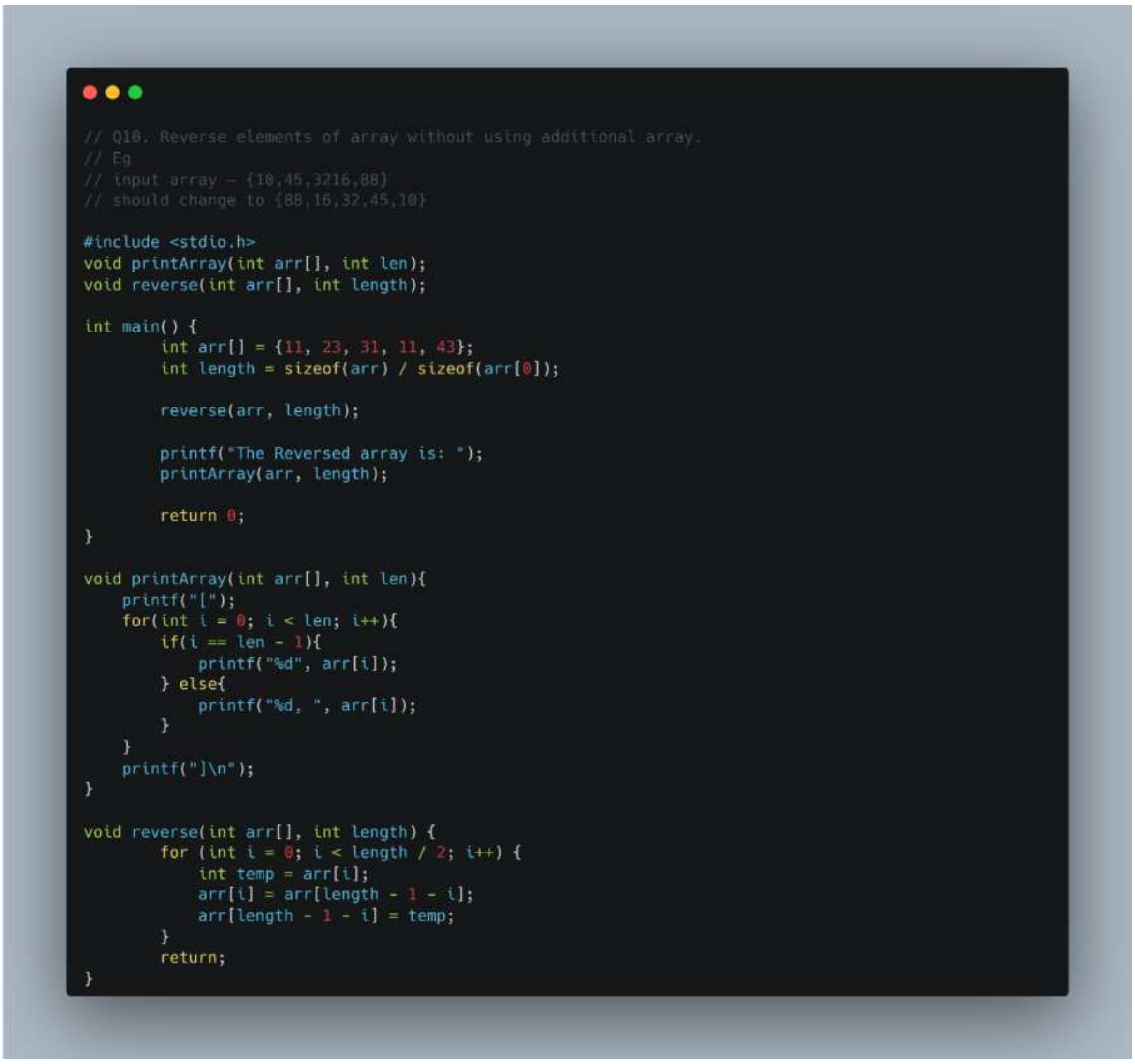
```c
/*
Q 10. Write a program to check if elements of an array are the same when read from front or back.
(Palindrome)
Example: 2 3 15 15 3 2
*/

#include <stdio.h>

int isPalindrome(int arr[], int len);
int main() {
        int arr[] = {2, 3, 15, 11, 3, 2};
        int len = sizeof(arr) / sizeof(arr[0]);

        if (isPalindrome(arr, len)){
            printf("Elements of the array are the same from front to back.\n");
        } else {
            printf("Elements of the array are not the same from front to back.\n");
        }

        return 0;
}


int isPalindrome(int arr[], int len) {
    for (int i = 0; i < len / 2; i++) {
        if (arr[i] != arr[len - 1 - i]) {
            return 0;
        }
    }
    return 1;
}
```

Output:



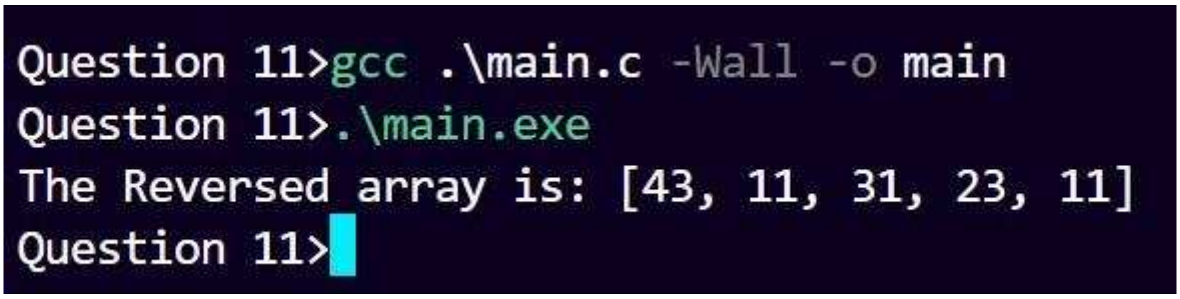
```
Question 10>gcc .\main.c -Wall -o main
Question 10>.\main.exe
Elements of the array are not the same from front to back.
Question 10>
```

Question 11: Reverse elements of array without using additional array



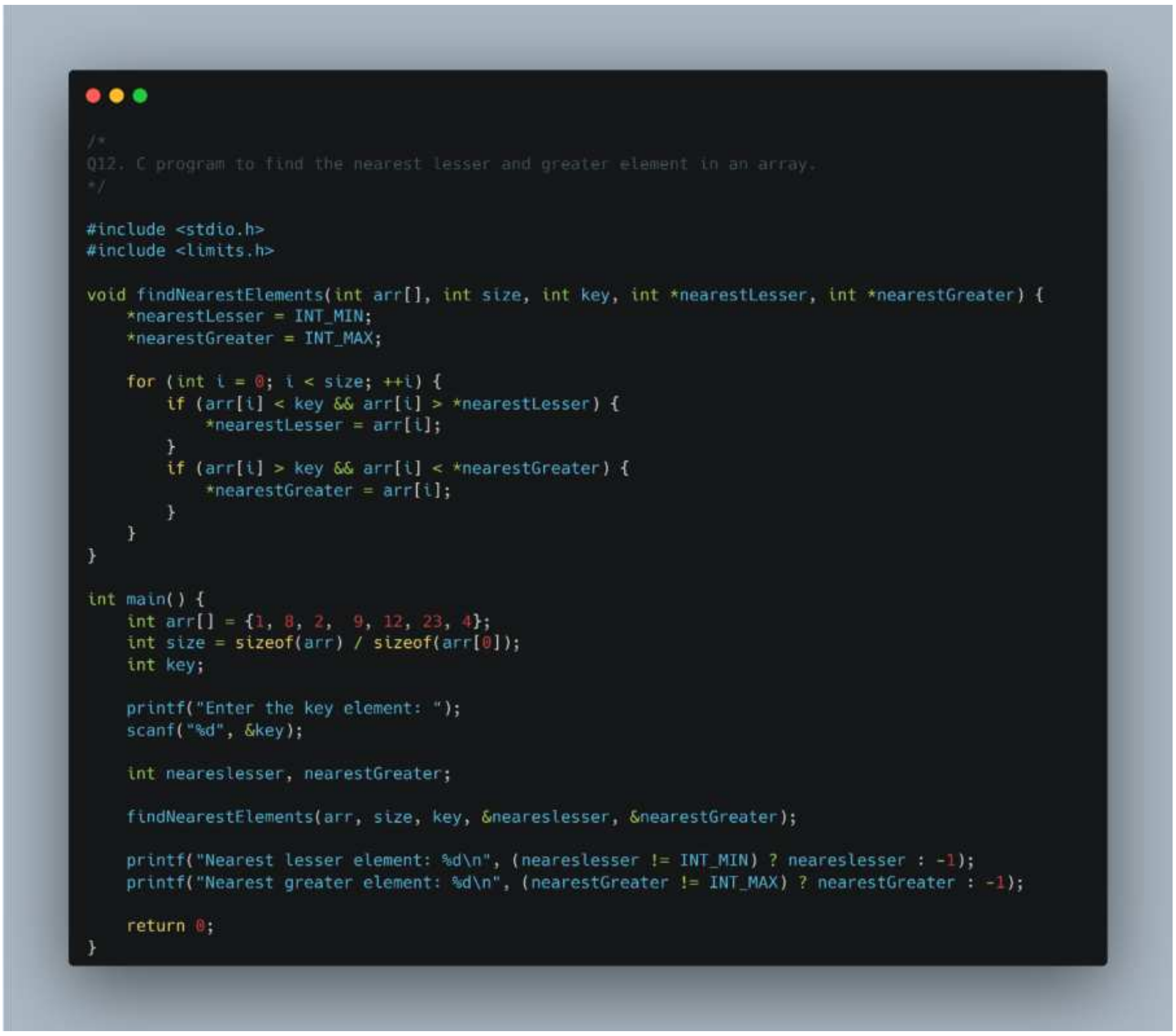
```c
// Q10. Reverse elements of array without using additional array.
// Eg
// input array - {10,45,3216,88}
// should change to {88,16,32,45,10}

#include <stdio.h>
void printArray(int arr[], int len);
void reverse(int arr[], int length);

int main() {
        int arr[] = {11, 23, 31, 11, 43};
        int length = sizeof(arr) / sizeof(arr[0]);

        reverse(arr, length);

        printf("The Reversed array is: ");
        printArray(arr, length);

        return 0;
}

void printArray(int arr[], int len){
    printf("[");
    for(int i = 0; i < len; i++){
        if(i == len - 1){
            printf("%d", arr[i]);
        } else{
            printf("%d, ", arr[i]);
        }
    }
    printf("]\n");
}

void reverse(int arr[], int length) {
        for (int i = 0; i < length / 2; i++) {
            int temp = arr[i];
            arr[i] = arr[length - 1 - i];
            arr[length - 1 - i] = temp;
        }
        return;
}
```

Output:



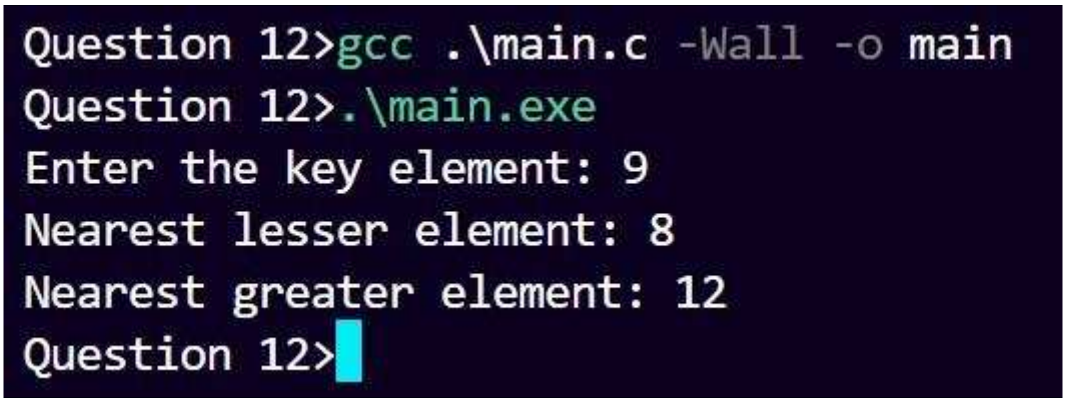
```
Question 11>gcc .\main.c -Wall -o main
Question 11>.\main.exe
The Reversed array is: [43, 11, 31, 23, 11]
Question 11>
```

Question 12: C program to find nearest lesser and greater element in an array.



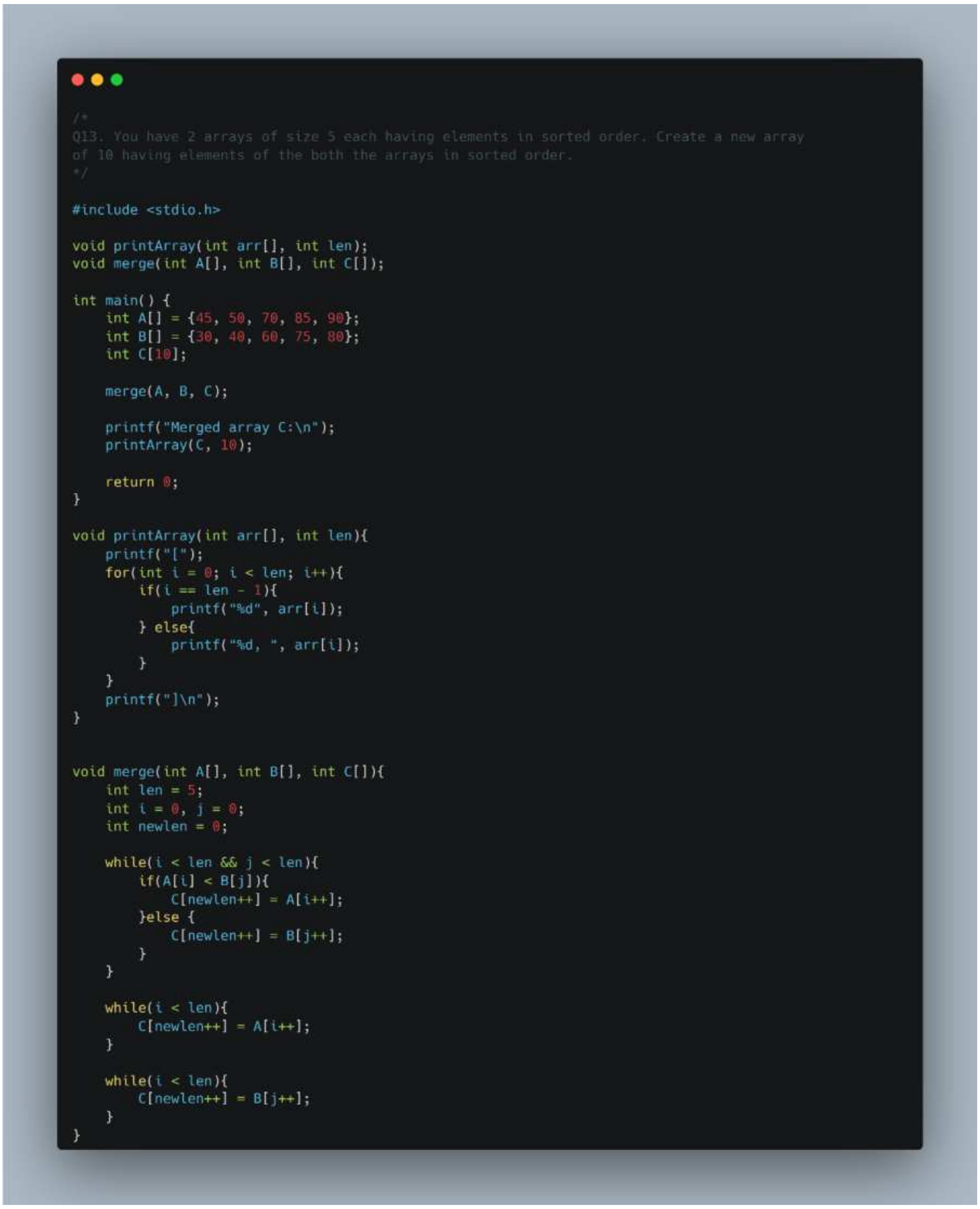
```c
/*
Q12. C program to find the nearest lesser and greater element in an array.
*/

#include <stdio.h>
#include <limits.h>

void findNearestElements(int arr[], int size, int key, int *nearestLesser, int *nearestGreater) {
    *nearestLesser = INT_MIN;
    *nearestGreater = INT_MAX;

    for (int i = 0; i < size; ++i) {
        if (arr[i] < key && arr[i] > *nearestLesser) {
            *nearestLesser = arr[i];
        }
        if (arr[i] > key && arr[i] < *nearestGreater) {
            *nearestGreater = arr[i];
        }
    }
}

int main() {
    int arr[] = {1, 8, 2,  9, 12, 23, 4};
    int size = sizeof(arr) / sizeof(arr[0]);
    int key;

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

    int neareslesser, nearestGreater;

    findNearestElements(arr, size, key, &neareslesser, &nearestGreater);

    printf("Nearest lesser element: %d\n", (neareslesser != INT_MIN) ? neareslesser : -1);
    printf("Nearest greater element: %d\n", (nearestGreater != INT_MAX) ? nearestGreater : -1);

    return 0;
}
```

Output:



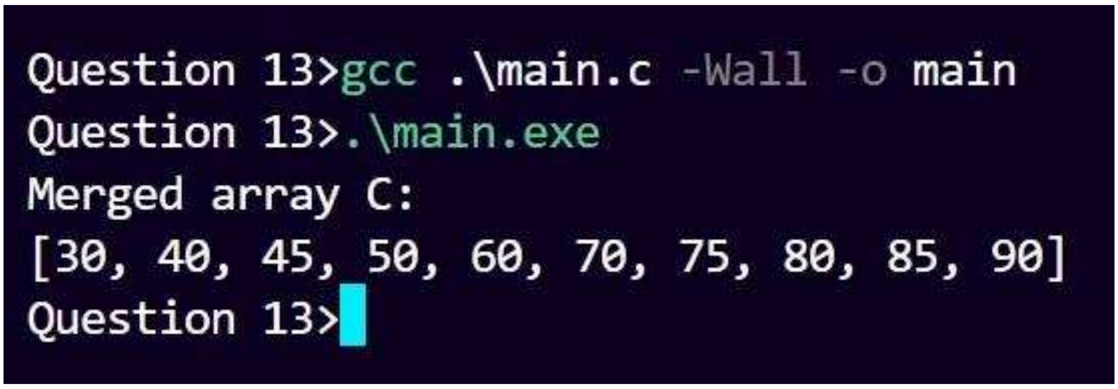
```
Question 12>gcc .\main.c -Wall -o main
Question 12>.\main.exe
Enter the key element: 9
Nearest lesser element: 8
Nearest greater element: 12
Question 12>
```

Question 13: You have 2 arrays of size 5 each having elements in sorted order. Create a new array of 10 having elements of the both the arrays in sorted order.



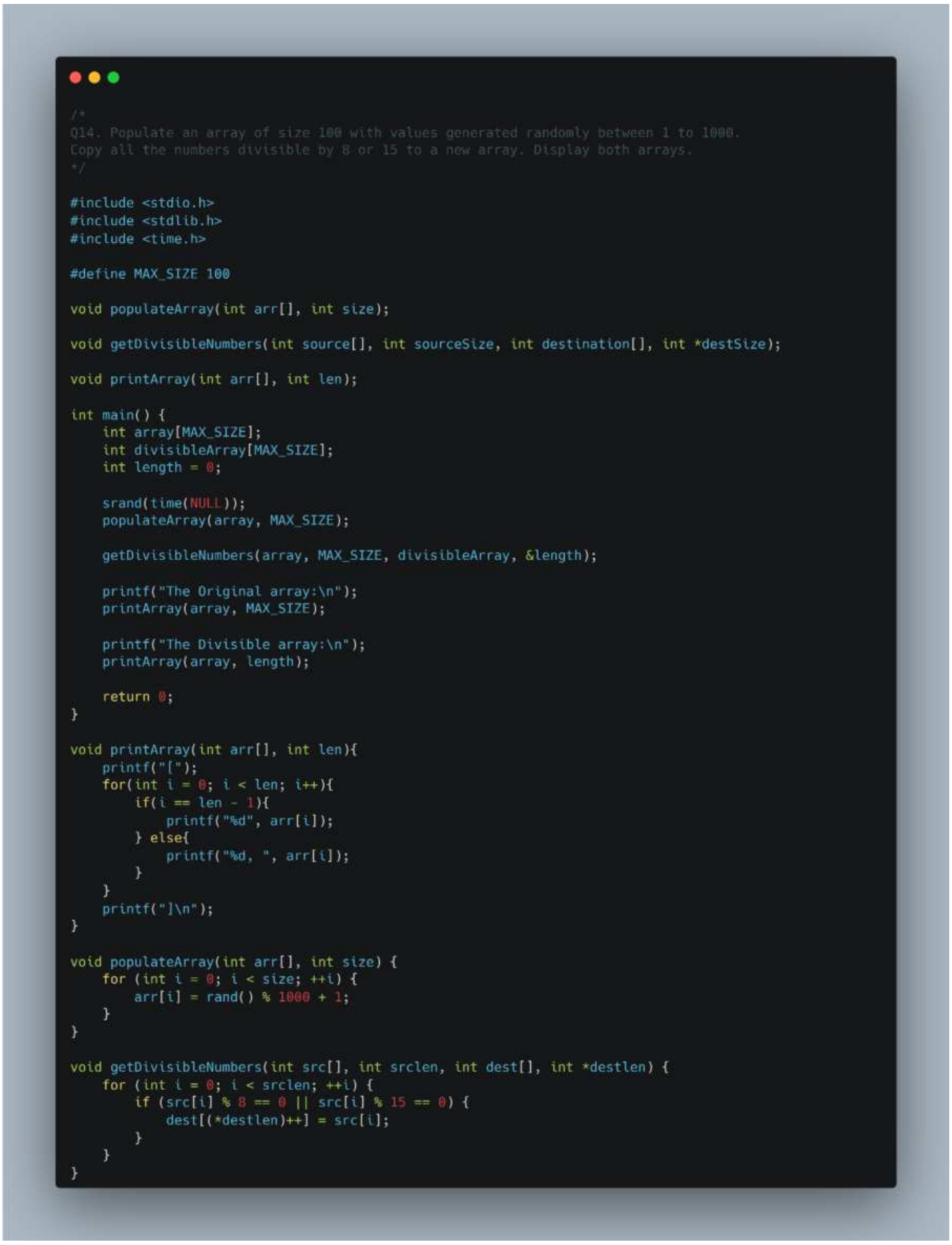
```c
/*
Q13. You have 2 arrays of size 5 each having elements in sorted order. Create a new array
of 10 having elements of the both the arrays in sorted order.
*/

#include <stdio.h>

void printArray(int arr[], int len);
void merge(int A[], int B[], int C[]);

int main() {
    int A[] = {45, 50, 70, 85, 90};
    int B[] = {30, 40, 60, 75, 80};
    int C[10];

    merge(A, B, C);

    printf("Merged array C:\n");
    printArray(C, 10);

    return 0;
}

void printArray(int arr[], int len){
    printf("[");
    for(int i = 0; i < len; i++){
        if(i == len - 1){
            printf("%d", arr[i]);
        } else{
            printf("%d, ", arr[i]);
        }
    }
    printf("]\n");
}


void merge(int A[], int B[], int C[]){
    int len = 5;
    int i = 0, j = 0;
    int newlen = 0;

    while(i < len && j < len){
        if(A[i] < B[j]){
            C[newlen++] = A[i++];
        }else {
            C[newlen++] = B[j++];
        }
    }

    while(i < len){
        C[newlen++] = A[i++];
    }

    while(i < len){
        C[newlen++] = B[j++];
    }
}
```

Output



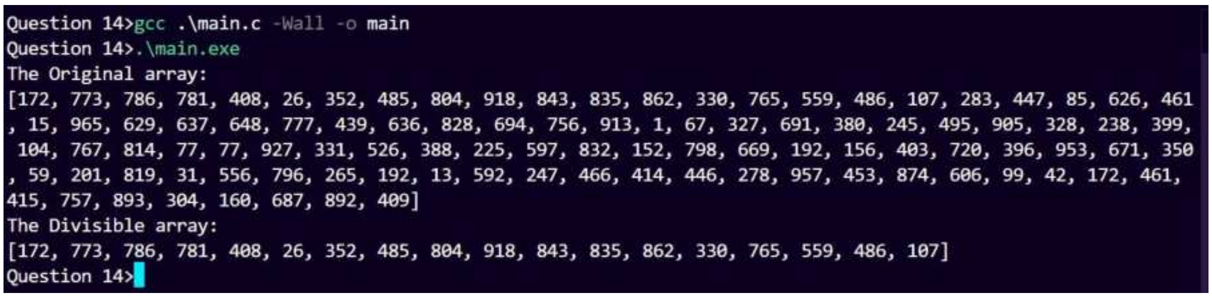
```
Question 13>gcc .\main.c -Wall -o main
Question 13>.\main.exe
Merged array C:
[30, 40, 45, 50, 60, 70, 75, 80, 85, 90]
Question 13>
```

Question 14: Populate an array of size 100 with values generated randomly between 1 to 1000. Copy all the numbers divisible by 8 or 15 to a new array. Display both arrays.



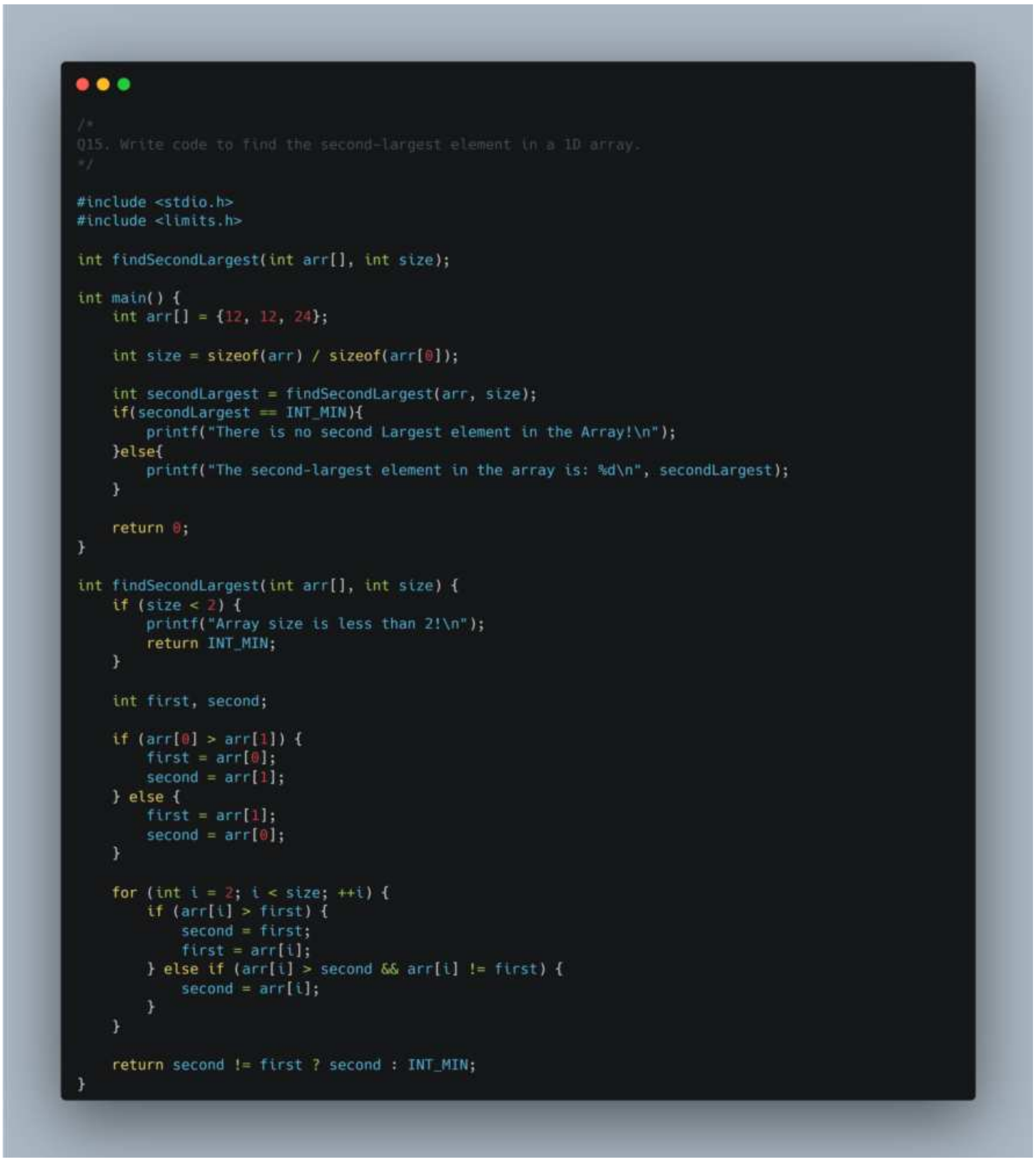
```c
/*
Q14. Populate an array of size 100 with values generated randomly between 1 to 1000.
Copy all the numbers divisible by 8 or 15 to a new array. Display both arrays.
*/

#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define MAX_SIZE 100

void populateArray(int arr[], int size);

void getDivisibleNumbers(int source[], int sourceSize, int destination[], int *destSize);

void printArray(int arr[], int len);

int main() {
    int array[MAX_SIZE];
    int divisibleArray[MAX_SIZE];
    int length = 0;

    srand(time(NULL));
    populateArray(array, MAX_SIZE);

    getDivisibleNumbers(array, MAX_SIZE, divisibleArray, &length);

    printf("The Original array:\n");
    printArray(array, MAX_SIZE);

    printf("The Divisible array:\n");
    printArray(array, length);

    return 0;
}

void printArray(int arr[], int len){
    printf("[");
    for(int i = 0; i < len; i++){
        if(i == len - 1){
            printf("%d", arr[i]);
        } else{
            printf("%d, ", arr[i]);
        }
    }
    printf("]\n");
}

void populateArray(int arr[], int size) {
    for (int i = 0; i < size; ++i) {
        arr[i] = rand() % 1000 + 1;
    }
}

void getDivisibleNumbers(int src[], int srclen, int dest[], int *destlen) {
    for (int i = 0; i < srclen; ++i) {
        if (src[i] % 8 == 0 || src[i] % 15 == 0) {
            dest[(*destlen)++] = src[i];
        }
    }
}
```

Output



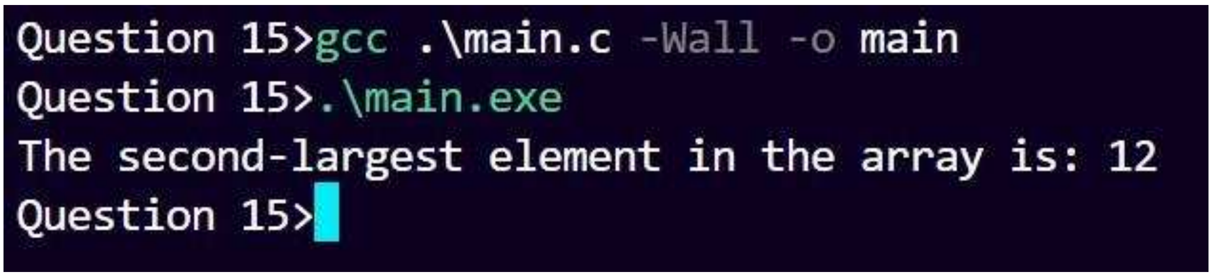
```
Question 14>gcc .\main.c -Wall -o main
Question 14>.\main.exe
The Original array:
[172, 773, 786, 781, 408, 26, 352, 485, 804, 918, 843, 835, 862, 330, 765, 559, 486, 107, 283, 447, 85, 626, 461
, 15, 965, 629, 637, 648, 777, 439, 636, 828, 694, 756, 913, 1, 67, 327, 691, 380, 245, 495, 905, 328, 238, 399,
 104, 767, 814, 77, 77, 927, 331, 526, 388, 225, 597, 832, 152, 798, 669, 192, 156, 403, 720, 396, 953, 671, 350
, 59, 201, 819, 31, 556, 796, 265, 192, 13, 592, 247, 466, 414, 446, 278, 957, 453, 874, 606, 99, 42, 172, 461,
415, 757, 893, 304, 160, 687, 892, 409]
The Divisible array:
[172, 773, 786, 781, 408, 26, 352, 485, 804, 918, 843, 835, 862, 330, 765, 559, 486, 107]
Question 14>
```

Question 15: Write code to find second largest element in a 1D Array.



```c
/*
Q15. Write code to find the second-largest element in a 1D array.
*/

#include <stdio.h>
#include <limits.h>

int findSecondLargest(int arr[], int size);

int main() {
    int arr[] = {12, 12, 24};

    int size = sizeof(arr) / sizeof(arr[0]);

    int secondLargest = findSecondLargest(arr, size);
    if(secondLargest == INT_MIN){
        printf("There is no second Largest element in the Array!\n");
    }else{
        printf("The second-largest element in the array is: %d\n", secondLargest);
    }

    return 0;
}

int findSecondLargest(int arr[], int size) {
    if (size < 2) {
        printf("Array size is less than 2!\n");
        return INT_MIN;
    }

    int first, second;

    if (arr[0] > arr[1]) {
        first = arr[0];
        second = arr[1];
    } else {
        first = arr[1];
        second = arr[0];
    }

    for (int i = 2; i < size; ++i) {
        if (arr[i] > first) {
            second = first;
            first = arr[i];
        } else if (arr[i] > second && arr[i] != first) {
            second = arr[i];
        }
    }

    return second != first ? second : INT_MIN;
}
```

Output



```
Question 15>gcc .\main.c -Wall -o main
Question 15>.\main.exe
The second-largest element in the array is: 12
Question 15>
```