ASSIGNMENT 1 SOLUTIONS

1) Develop a Menu driven program to demonstrate the following operations of Arrays

——MENU——-

1. CREATE

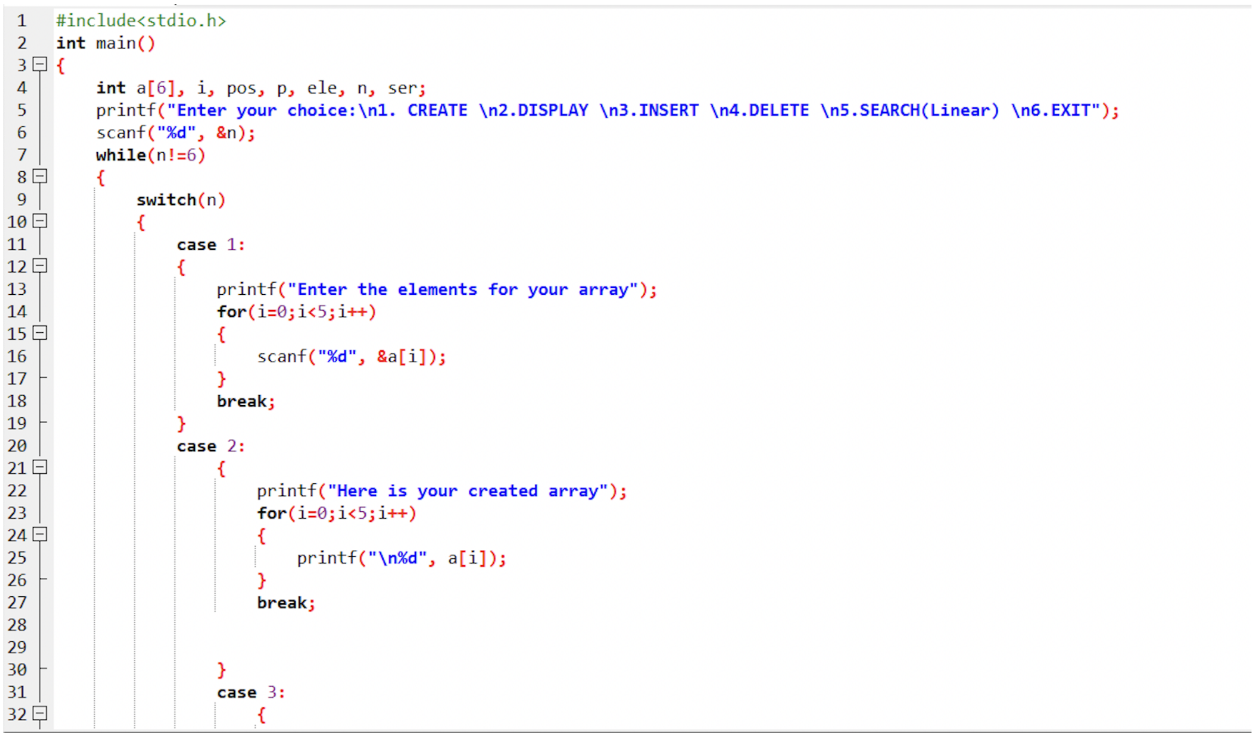
2. DISPLAY

3. INSERT

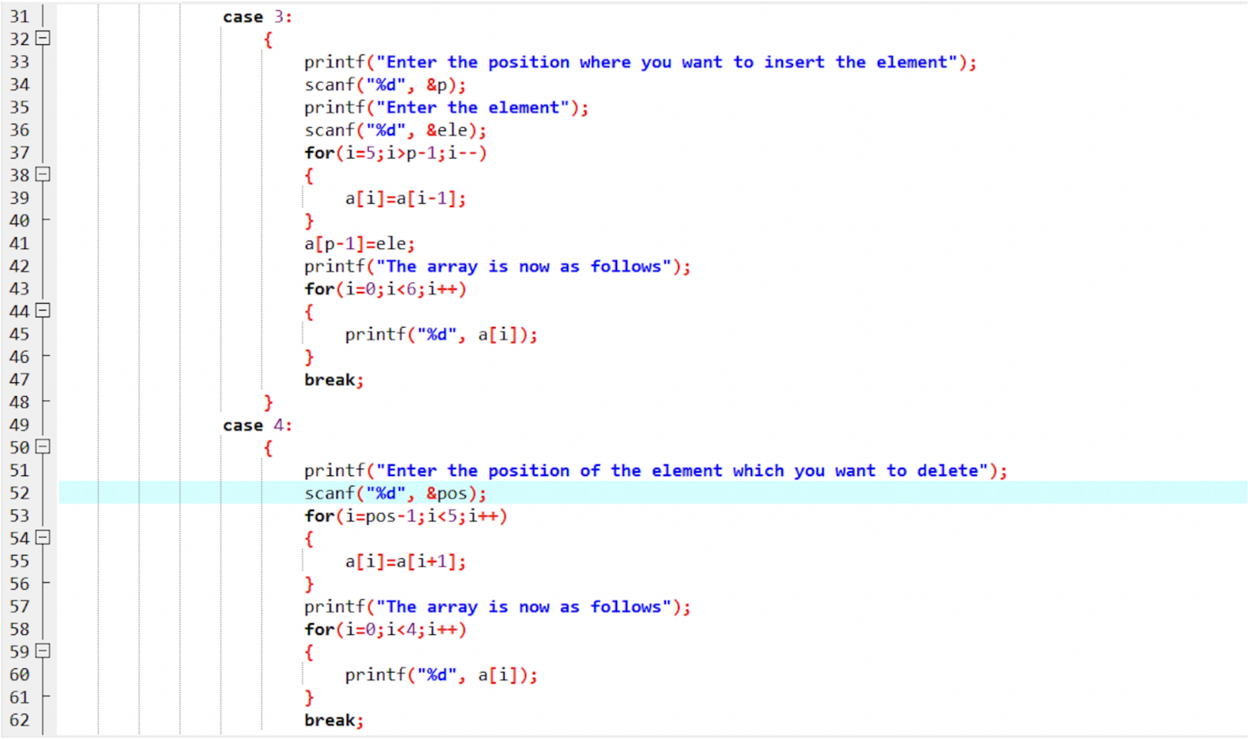
4. DELETE

5. SEARCH (Linear)

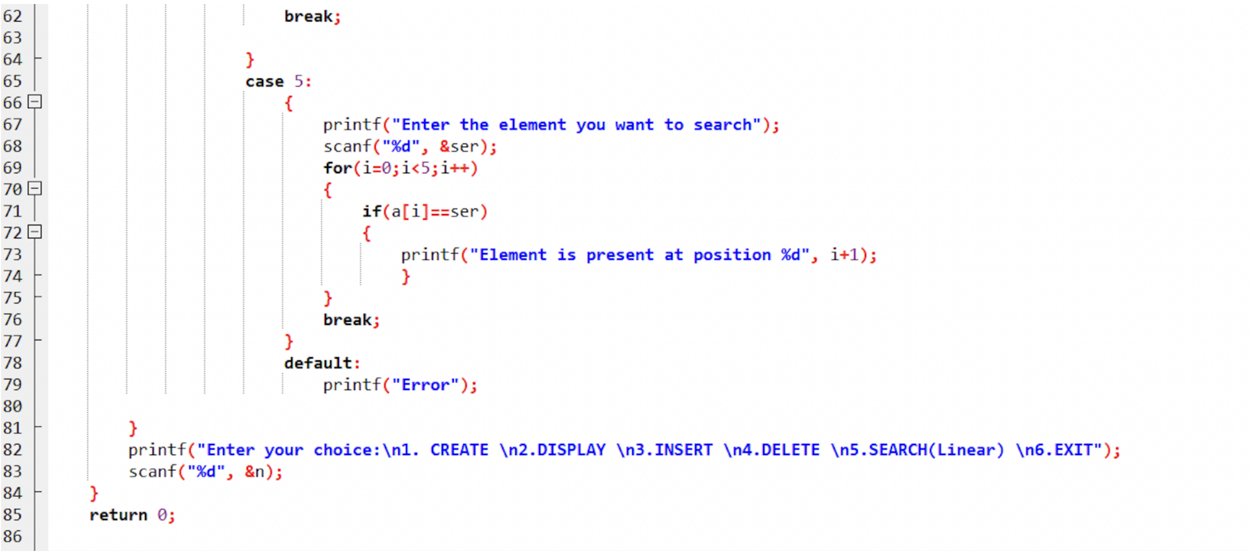
6. EXIT



```
#include<stdio.h>
int main()
{
    int a[6], i, pos, p, ele, n, ser;
    printf("Enter your choice:\n1. CREATE \n2.DISPLAY \n3.INSERT \n4.DELETE \n5.SEARCH(Linear) \n6.EXIT");
    scanf("%d", &n);
    while(n!=6)
    {
        switch(n)
        {
            case 1:
            {
                printf("Enter the elements for your array");
                for(i=0;i<5;i++)
                {
                    scanf("%d", &a[i]);
                }
                break;
            }
            case 2:
                {
                    printf("Here is your created array");
                    for(i=0;i<5;i++)
                    {
                        printf("\n%d", a[i]);
                    }
                    break;


                }
                case 3:
                    {
```

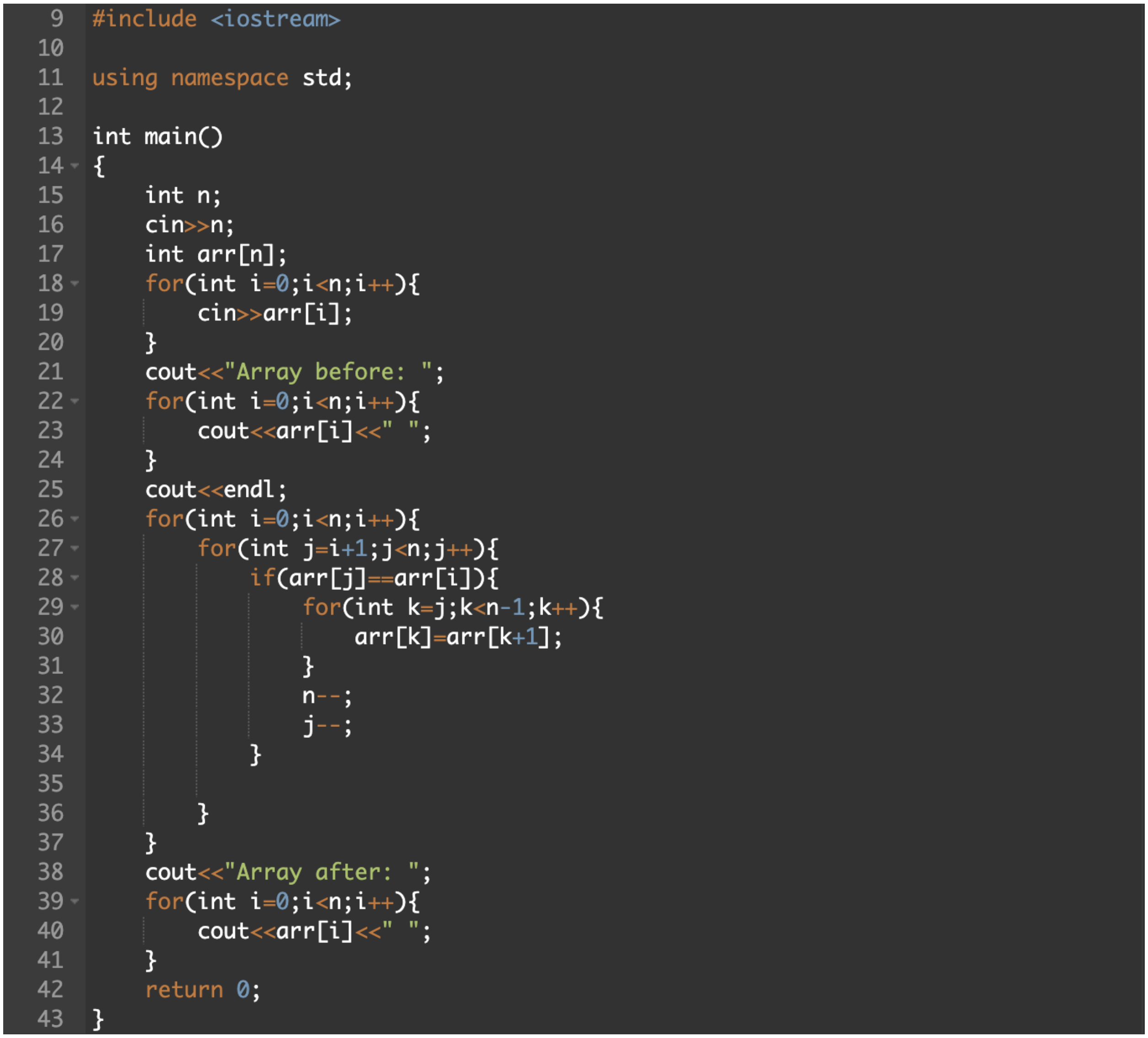


```
                    case 3:
                        {
                            printf("Enter the position where you want to insert the element");
                            scanf("%d", &p);
                            printf("Enter the element");
                            scanf("%d", &ele);
                            for(i=5;i>p-1;i--)
                            {
                                a[i]=a[i-1];
                            }
                            a[p-1]=ele;
                            printf("The array is now as follows");
                            for(i=0;i<6;i++)
                            {
                                printf("%d", a[i]);
                            }
                            break;
                        }
                    case 4:
                        {
                            printf("Enter the position of the element which you want to delete");
                            scanf("%d", &pos);
                            for(i=pos-1;i<5;i++)
                            {
                                a[i]=a[i+1];
                            }
                            printf("The array is now as follows");
                            for(i=0;i<4;i++)
                            {
                                printf("%d", a[i]);
                            }
                            break;
```



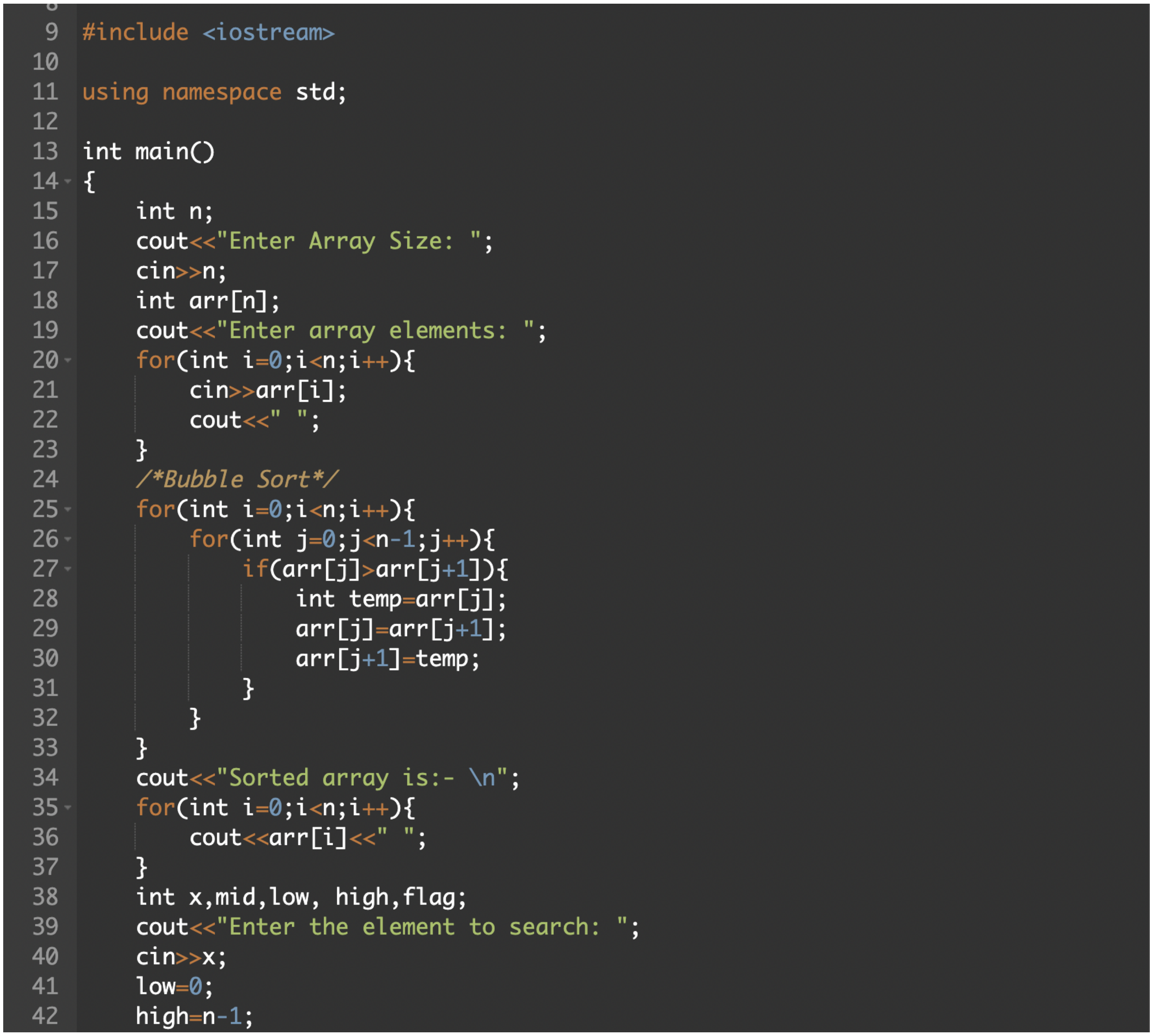
```
                            break;

                        }
                    case 5:
                        {
                            printf("Enter the element you want to search");
                            scanf("%d", &ser);
                            for(i=0;i<5;i++)
                            {
                                if(a[i]==ser)
                                {
                                    printf("Element is present at position %d", i+1);
                                }
                            }
                            break;
                        }
                        default:
                            printf("Error");

        }
        printf("Enter your choice:\n1. CREATE \n2.DISPLAY \n3.INSERT \n4.DELETE \n5.SEARCH(Linear) \n6.EXIT");
        scanf("%d", &n);
    }
    return 0;

```

2) Design the logic to remove the duplicate elements from an Array and after the deletion, the array should contain the unique elements

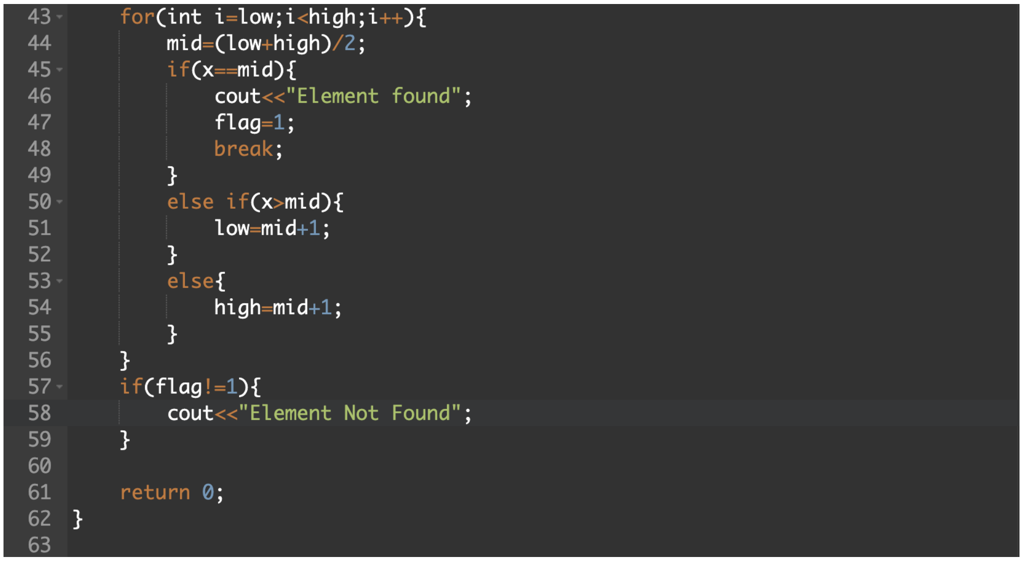


```cpp
#include <iostream>

using namespace std;

int main()
{
    int n;
    cin>>n;
    int arr[n];
    for(int i=0;i<n;i++){
        cin>>arr[i];
    }
    cout<<"Array before: ";
    for(int i=0;i<n;i++){
        cout<<arr[i]<<" ";
    }
    cout<<endl;
    for(int i=0;i<n;i++){
        for(int j=i+1;j<n;j++){
            if(arr[j]==arr[i]){
                for(int k=j;k<n-1;k++){
                    arr[k]=arr[k+1];
                }
                n--;
                j--;
            }

        }
    }
    cout<<"Array after: ";
    for(int i=0;i<n;i++){
        cout<<arr[i]<<" ";
    }
    return 0;
}
```

3) Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in wrong order. Code the Bubble sort with the following elements:

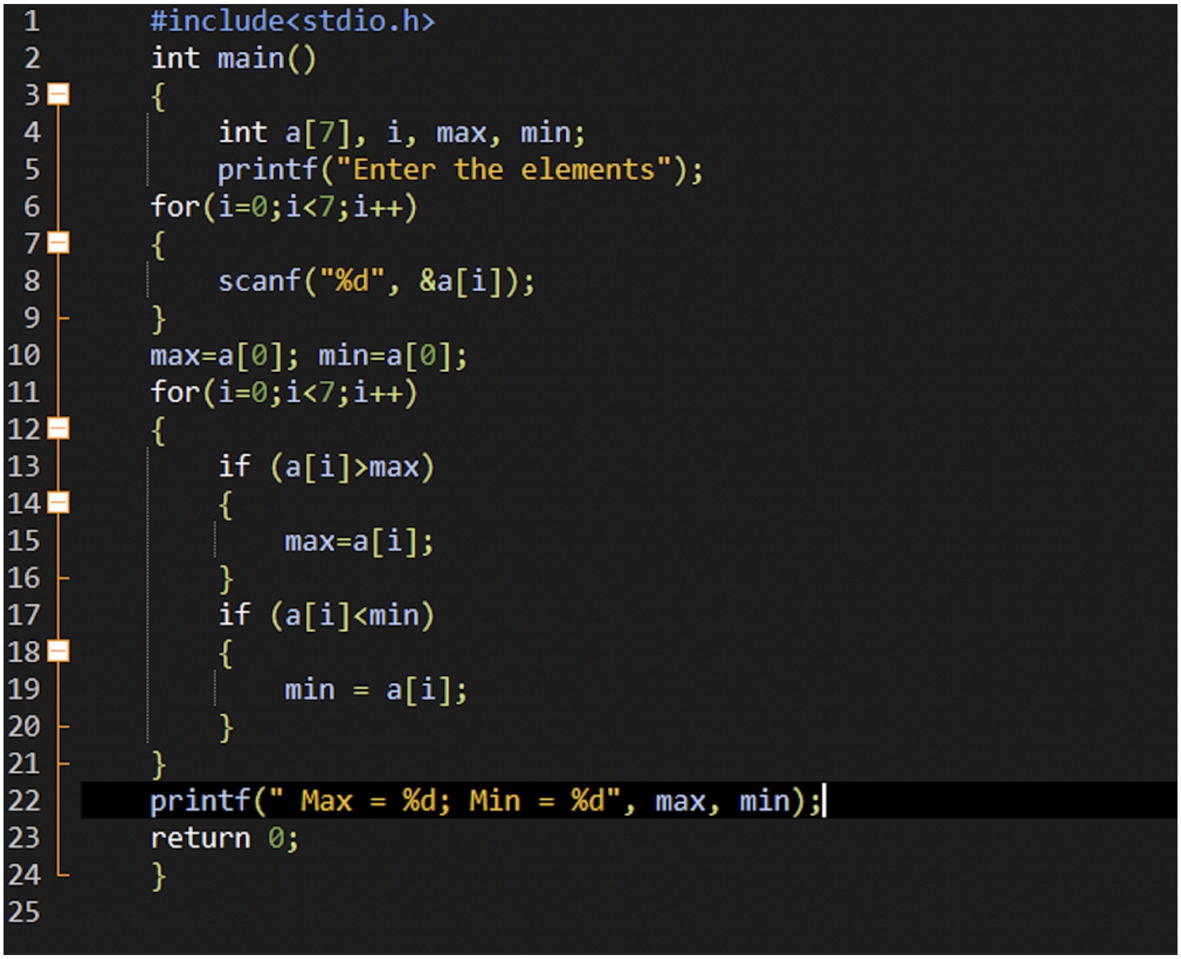


```cpp
#include <iostream>

using namespace std;

int main()
{
    int n;
    cout<<"Enter Array Size: ";
    cin>>n;
    int arr[n];
    cout<<"Enter array elements: ";
    for(int i=0;i<n;i++){
        cin>>arr[i];
        cout<<" ";
    }
    /*Bubble Sort*/
    for(int i=0;i<n;i++){
        for(int j=0;j<n-1;j++){
            if(arr[j]>arr[j+1]){
                int temp=arr[j];
                arr[j]=arr[j+1];
                arr[j+1]=temp;
            }
        }
    }
    cout<<"Sorted array is:- \n";
    for(int i=0;i<n;i++){
        cout<<arr[i]<<" ";
    }
    int x,mid,low, high,flag;
    cout<<"Enter the element to search: ";
    cin>>x;
    low=0;
    high=n-1;
```



```cpp
    for(int i=low;i<high;i++){
        mid=(low+high)/2;
        if(x==mid){
            cout<<"Element found";
            flag=1;
            break;
        }
        else if(x>mid){
            low=mid+1;
        }
        else{
            high=mid+1;
        }
    }
    if(flag!=1){
        cout<<"Element Not Found";
    }

    return 0;
}
```

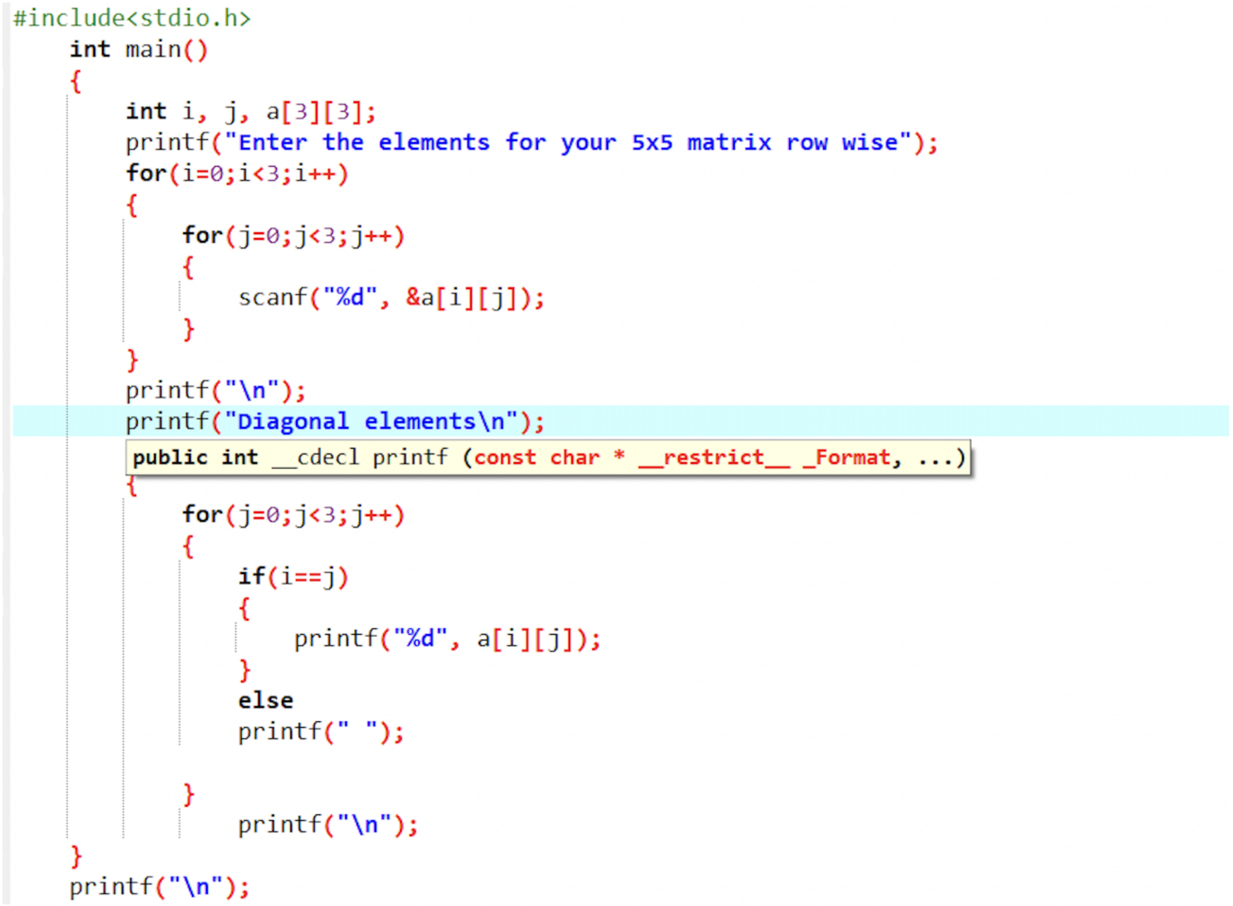
4) **Develop a program to find the minimum and maximum elements in an array.**



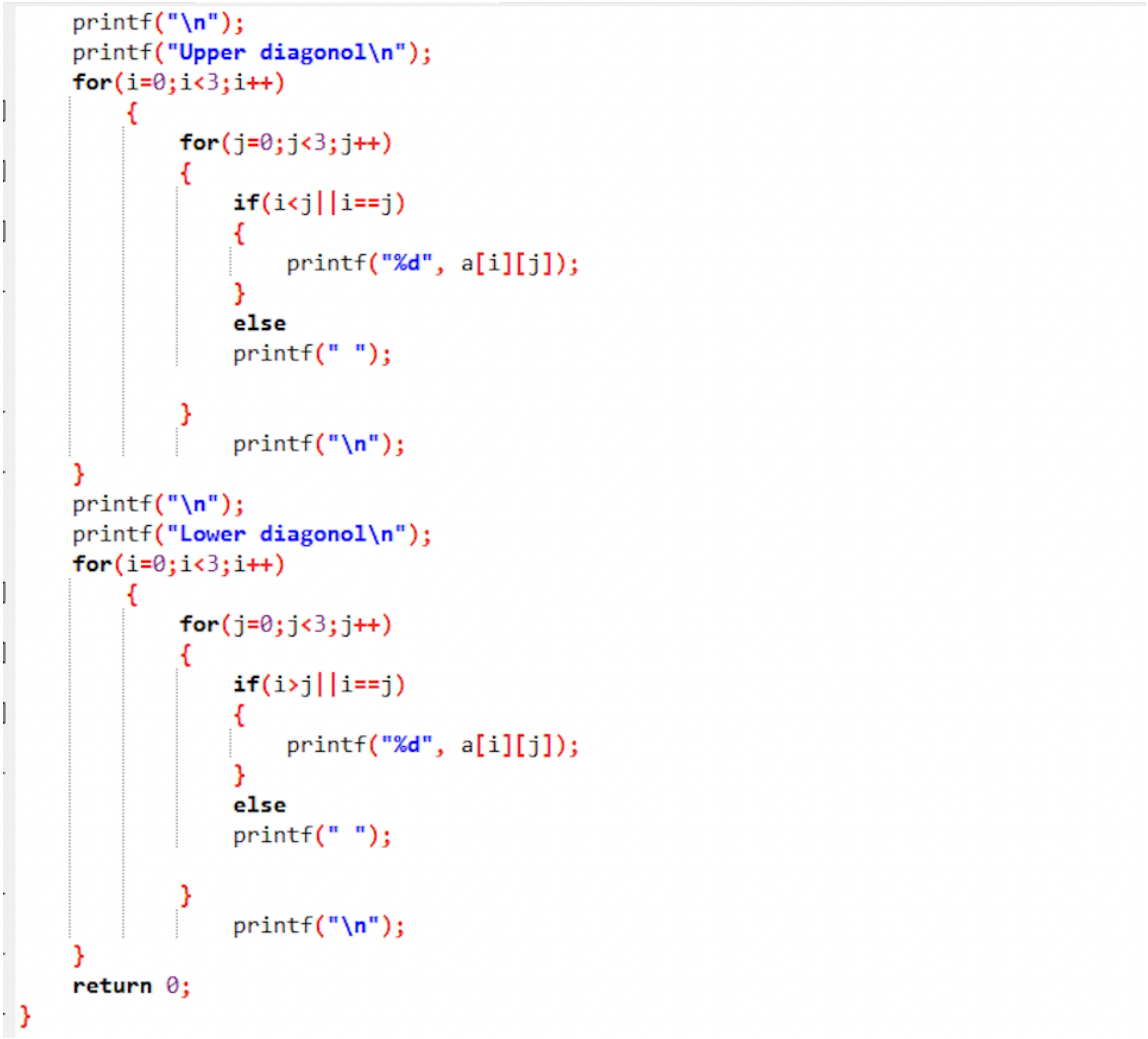
```
#include<stdio.h>
int main()
{
    int a[7], i, max, min;
    printf("Enter the elements");
for(i=0;i<7;i++)
{
    scanf("%d", &a[i]);
}
max=a[0]; min=a[0];
for(i=0;i<7;i++)
{
    if (a[i]>max)
    {
        max=a[i];
    }
    if (a[i]<min)
    {
        min = a[i];
    }
}
printf(" Max = %d; Min = %d", max, min);
return 0;
}
```

5) **Develop the following programs**

a) **Printing the diagonal, upper and lower diagonal elements of a matrix.**

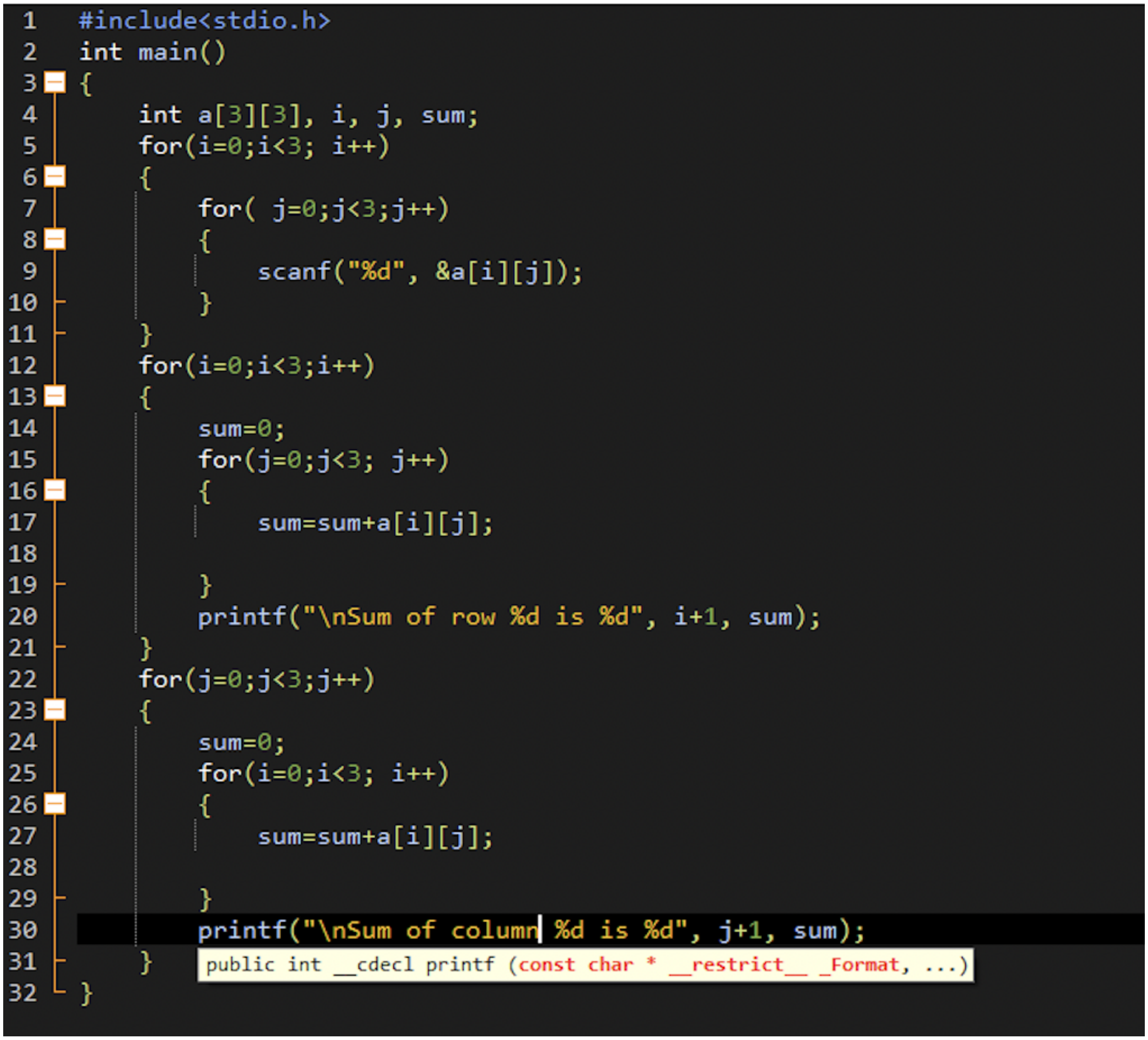


```
#include<stdio.h>
    int main()
    {
        int i, j, a[3][3];
        printf("Enter the elements for your 5x5 matrix row wise");
        for(i=0;i<3;i++)
        {
            for(j=0;j<3;j++)
            {
                scanf("%d", &a[i][j]);
            }
        }
        printf("\n");
        printf("Diagonal elements\n");
        public int __cdecl printf (const char * __restrict__ _Format, ...)
        {
            for(j=0;j<3;j++)
            {
                if(i==j)
                {
                    printf("%d", a[i][j]);
                }
                else
                printf(" ");

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



```
    printf("\n");
    printf("Upper diagonol\n");
    for(i=0;i<3;i++)
        {
            for(j=0;j<3;j++)
            {
                if(i<j||i==j)
                {
                    printf("%d", a[i][j]);
                }
                else
                printf(" ");

            }
                printf("\n");
    }
    printf("\n");
    printf("Lower diagonol\n");
    for(i=0;i<3;i++)
        {
            for(j=0;j<3;j++)
            {
                if(i>j||i==j)
                {
                    printf("%d", a[i][j]);
                }
                else
                printf(" ");

            }
                printf("\n");
    }
    return 0;
}
```

b)Find the row-wise and column-wise sum of all the elements that present in a two-dimensional array.



```c
#include<stdio.h>
int main()
{
    int a[3][3], i, j, sum;
    for(i=0;i<3; i++)
    {
        for( j=0;j<3;j++)
        {
            scanf("%d", &a[i][j]);
        }
    }
    for(i=0;i<3;i++)
    {
        sum=0;
        for(j=0;j<3; j++)
        {
            sum=sum+a[i][j];

        }
        printf("\nSum of row %d is %d", i+1, sum);
    }
    for(j=0;j<3;j++)
    {
        sum=0;
        for(i=0;i<3; i++)
        {
            sum=sum+a[i][j];

        }
        printf("\nSum of column %d is %d", j+1, sum);
    }
}
```

public int __cdecl printf (const char * __restrict__ _Format, ...)