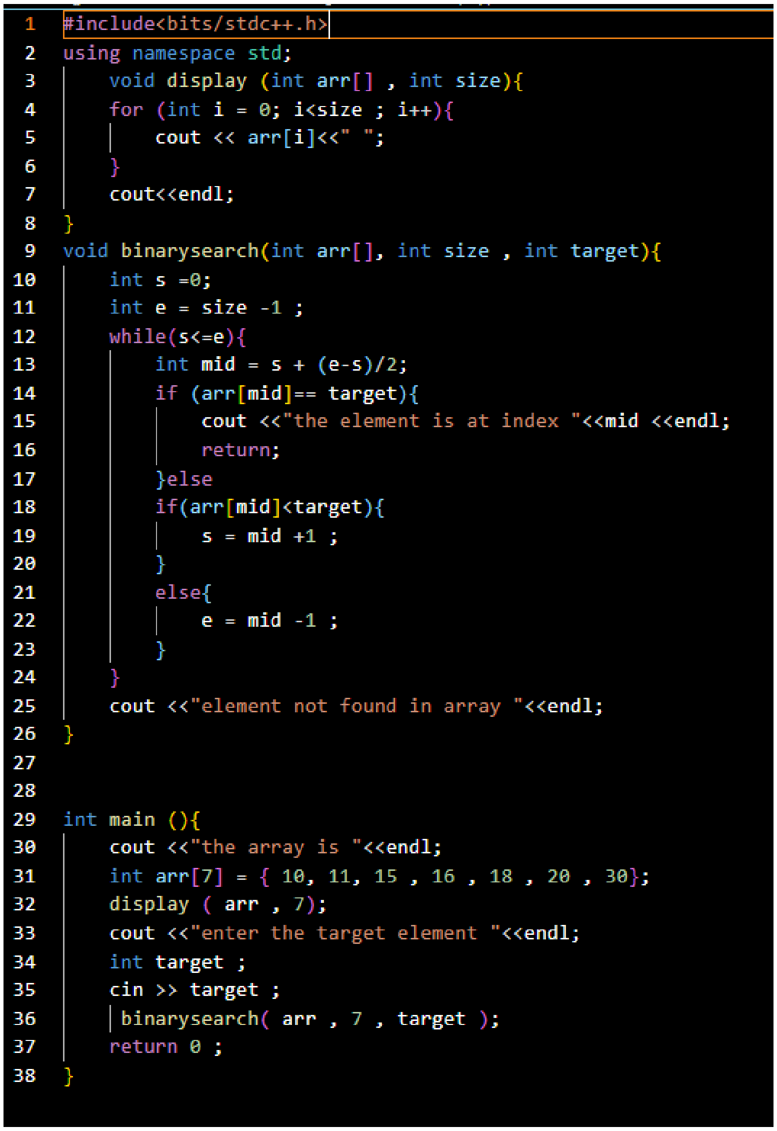
# ASSIGNMENT 2:

NAME: Ujjwal Pant

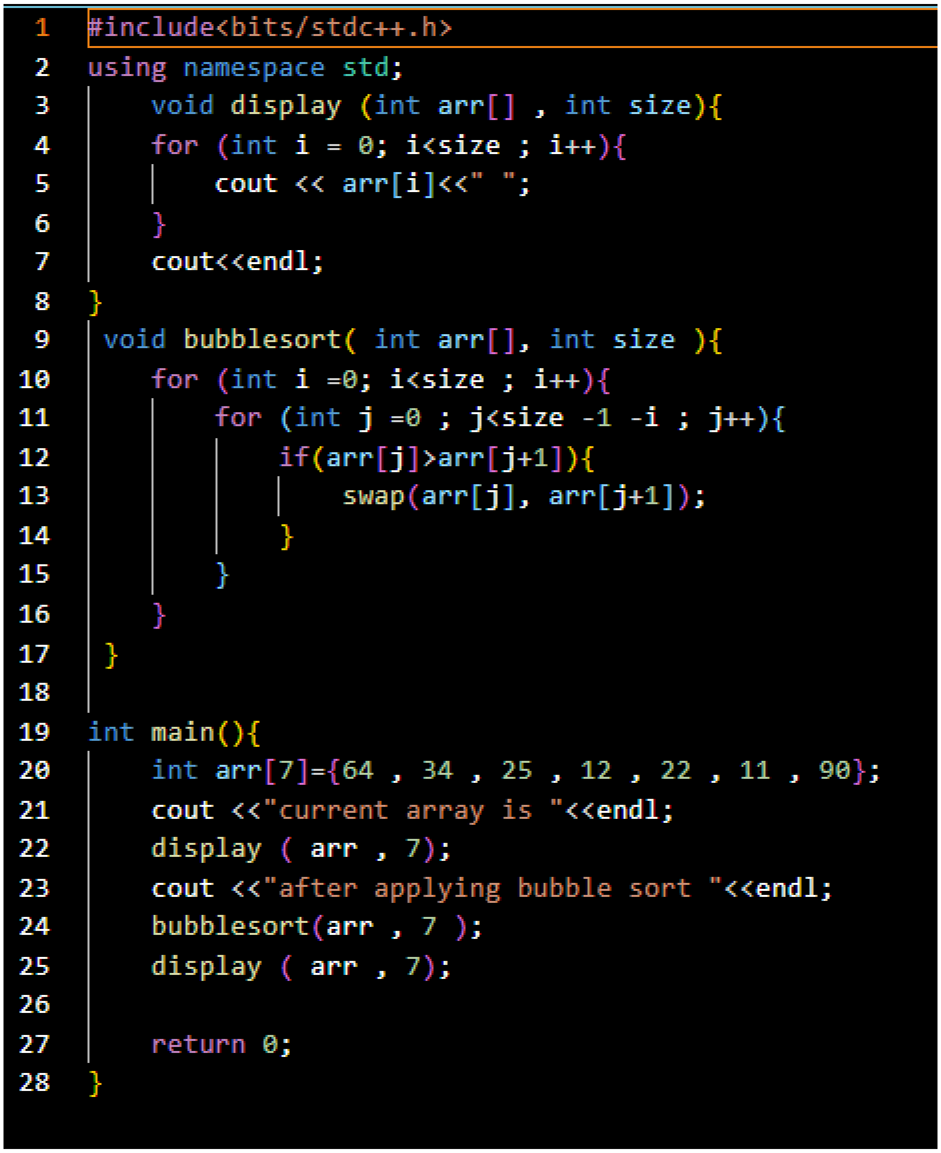
ROLL NUMBER: 1024030370

QUE1: Implement the Binary search algorithm regarded as a fast search algorithm with run-time complexity of O(log n) in comparison to the Linear Search.



```
#include<bits/stdc++.h>
using namespace std;
    void display (int arr[] , int size){
    for (int i = 0; i<size ; i++){
        cout << arr[i]<<" ";
    }
    cout<<endl;
}
void binarysearch(int arr[], int size , int target){
    int s =0;
    int e = size -1 ;
    while(s<=e){
        int mid = s + (e-s)/2;
        if (arr[mid]== target){
            cout <<"the element is at index "<<mid <<endl;
            return;
        }else
        if(arr[mid]<target){
            s = mid +1 ;
        }
        else{
            e = mid -1 ;
        }
    }
    cout <<"element not found in array "<<endl;
}


int main (){
    cout <<"the array is "<<endl;
    int arr[7] = { 10, 11, 15 , 16 , 18 , 20 , 30};
    display ( arr , 7);
    cout <<"enter the target element "<<endl;
    int target ;
    cin >> target ;
    binarysearch( arr , 7 , target );
    return 0 ;
}
```

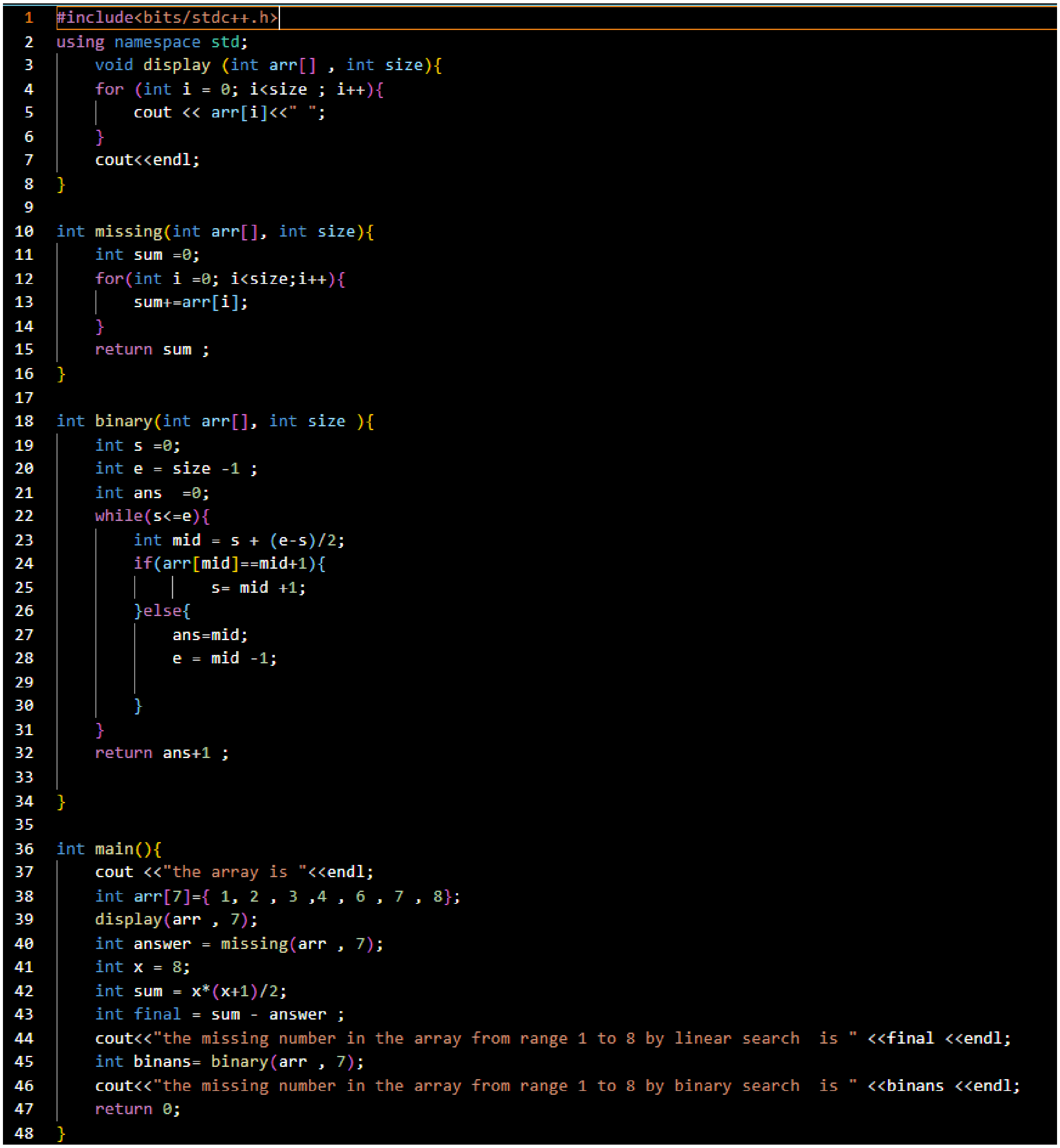
QUE2: Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in the wrong order. Code the Bubble sort with the following elements: 64 34 25 12 22 11 90



```
#include<bits/stdc++.h>
using namespace std;
    void display (int arr[] , int size){
    for (int i = 0; i<size ; i++){
        cout << arr[i]<<" ";
    }
    cout<<endl;
}
 void bubblesort( int arr[], int size ){
    for (int i =0; i<size ; i++){
        for (int j =0 ; j<size -1 -i ; j++){
            if(arr[j]>arr[j+1]){
                swap(arr[j], arr[j+1]);
            }
        }
    }
 }

int main(){
    int arr[7]={64 , 34 , 25 , 12 , 22 , 11 , 90};
    cout <<"current array is "<<endl;
    display ( arr , 7);
    cout <<"after applying bubble sort "<<endl;
    bubblesort(arr , 7 );
    display ( arr , 7);

    return 0;
}
```

QUE3: Given an array of n-1 distinct integers in the range of 1 to n, find the missing number in it in a Sorted Array (a) Linear time (b) Using binary search

.



```cpp
#include<bits/stdc++.h>
using namespace std;
    void display (int arr[] , int size){
    for (int i = 0; i<size ; i++){
        cout << arr[i]<<" ";
    }
    cout<<endl;
}

int missing(int arr[], int size){
    int sum =0;
    for(int i =0; i<size;i++){
        sum+=arr[i];
    }
    return sum ;
}

int binary(int arr[], int size ){
    int s =0;
    int e = size -1 ;
    int ans  =0;
    while(s<=e){
        int mid = s + (e-s)/2;
        if(arr[mid]==mid+1){
                s= mid +1;
        }else{
            ans=mid;
            e = mid -1;

        }
    }
    return ans+1 ;

}

int main(){
    cout <<"the array is "<<endl;
    int arr[7]={ 1, 2 , 3 ,4 , 6 , 7 , 8};
    display(arr , 7);
    int answer = missing(arr , 7);
    int x = 8;
    int sum = x*(x+1)/2;
    int final = sum - answer ;
    cout<<"the missing number in the array from range 1 to 8 by linear search  is " <<final <<endl;
    int binans= binary(arr , 7);
    cout<<"the missing number in the array from range 1 to 8 by binary search  is " <<binans <<endl;
    return 0;
}
```

QUE4: String Related Programs

(a) Write a program to concatenate one string to another string.

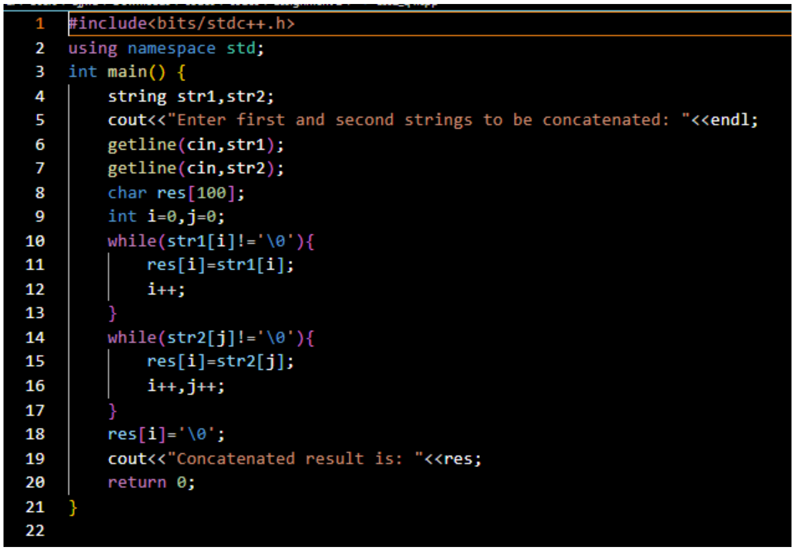
(b) Write a program to reverse a string.

(c) Write a program to delete all the vowels from the string.

(d) Write a program to sort the strings in alphabetical order.

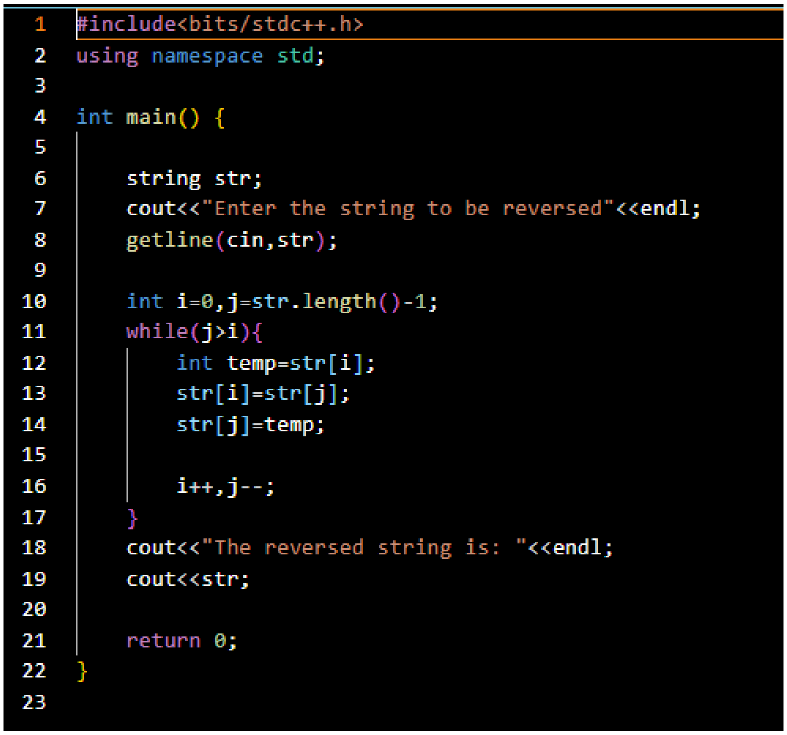
(e) Write a program to convert a character from uppercase to lowercase.

a)



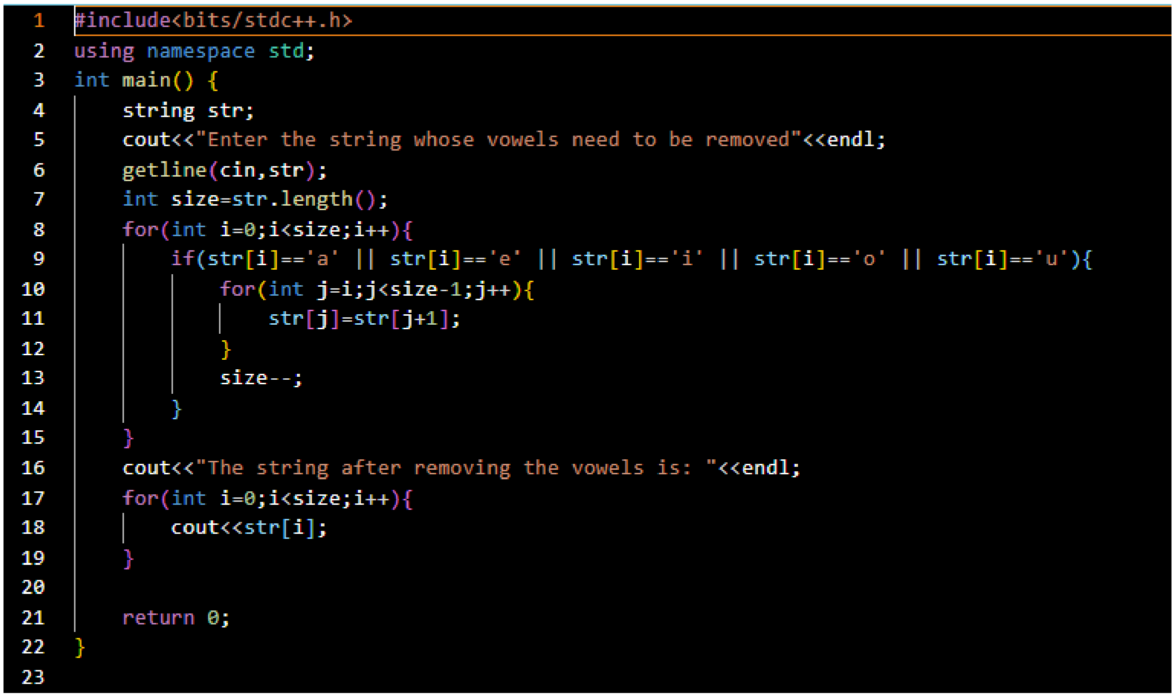
```cpp
#include<bits/stdc++.h>
using namespace std;
int main() {
    string str1,str2;
    cout<<"Enter first and second strings to be concatenated: "<<endl;
    getline(cin,str1);
    getline(cin,str2);
    char res[100];
    int i=0,j=0;
    while(str1[i]!='\0'){
        res[i]=str1[i];
        i++;
    }
    while(str2[j]!='\0'){
        res[i]=str2[j];
        i++,j++;
    }
    res[i]='\0';
    cout<<"Concatenated result is: "<<res;
    return 0;
}
```

b)



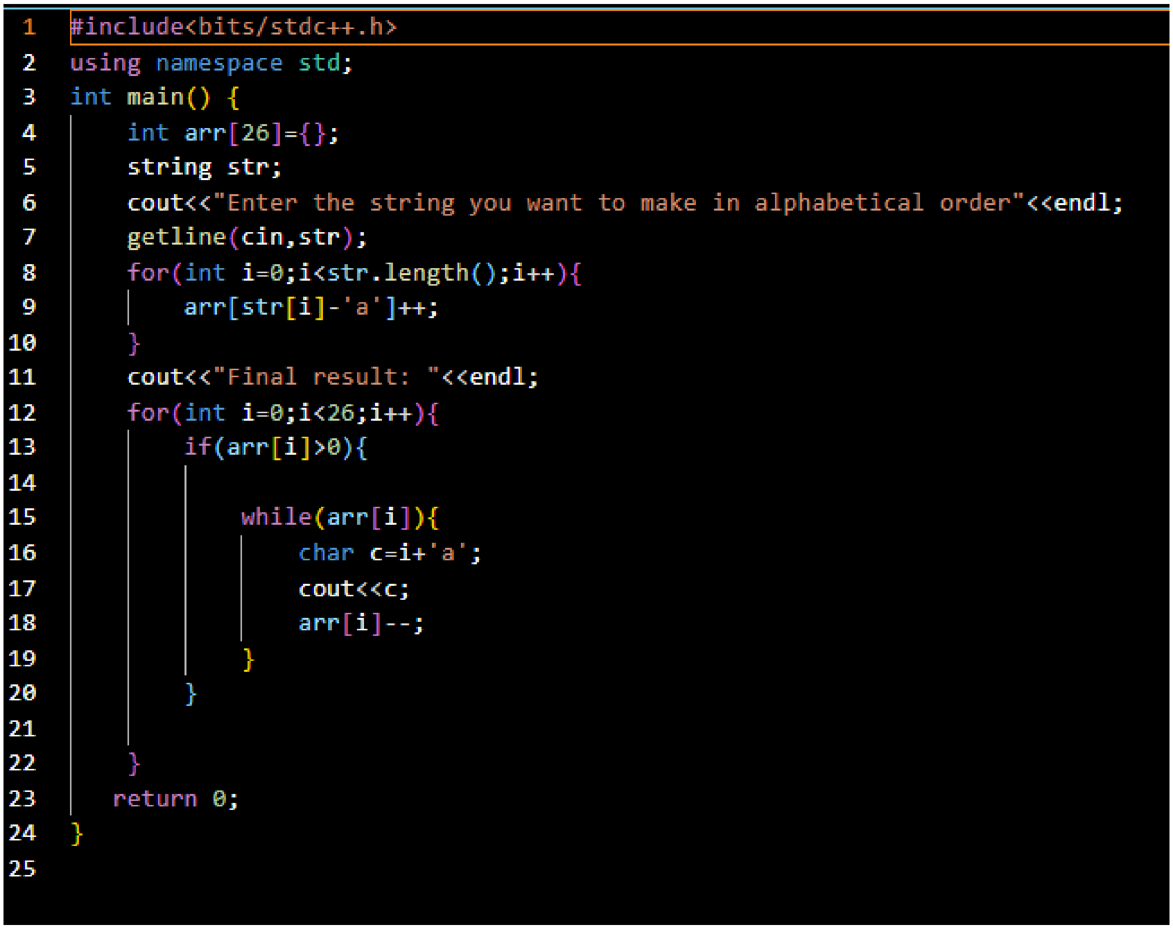
```cpp
#include<bits/stdc++.h>
using namespace std;

int main() {

    string str;
    cout<<"Enter the string to be reversed"<<endl;
    getline(cin,str);

    int i=0,j=str.length()-1;
    while(j>i){
        int temp=str[i];
        str[i]=str[j];
        str[j]=temp;

        i++,j--;
    }
    cout<<"The reversed string is: "<<endl;
    cout<<str;

    return 0;
}
```

c)



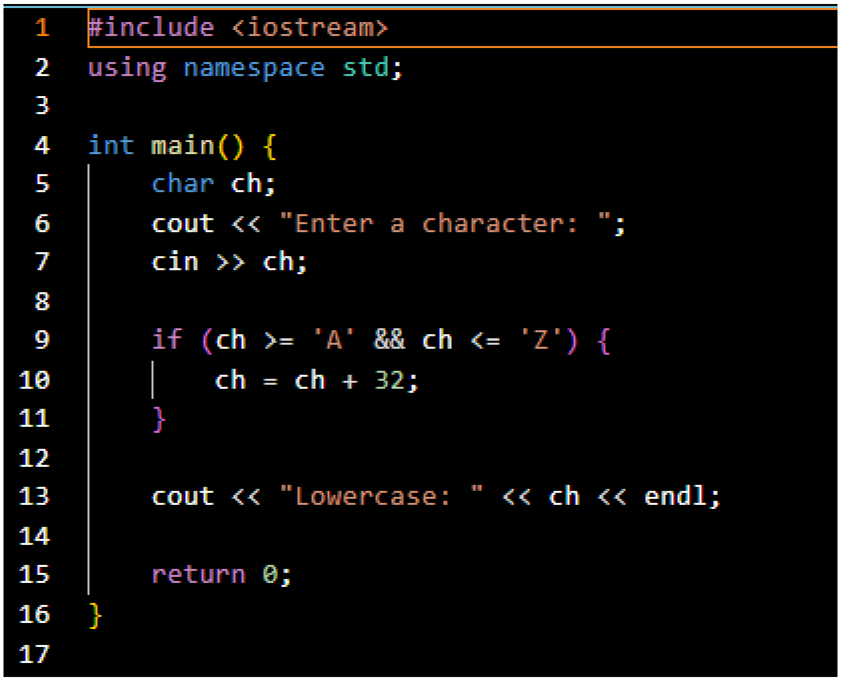
```cpp
#include<bits/stdc++.h>
using namespace std;
int main() {
    string str;
    cout<<"Enter the string whose vowels need to be removed"<<endl;
    getline(cin,str);
    int size=str.length();
    for(int i=0;i<size;i++){
        if(str[i]=='a' || str[i]=='e' || str[i]=='i' || str[i]=='o' || str[i]=='u'){
            for(int j=i;j<size-1;j++){
                str[j]=str[j+1];
            }
            size--;
        }
    }
    cout<<"The string after removing the vowels is: "<<endl;
    for(int i=0;i<size;i++){
        cout<<str[i];
    }

    return 0;
}
```

d)



```cpp
#include<bits/stdc++.h>
using namespace std;
int main() {
    int arr[26]={};
    string str;
    cout<<"Enter the string you want to make in alphabetical order"<<endl;
    getline(cin,str);
    for(int i=0;i<str.length();i++){
        arr[str[i]-'a']++;
    }
    cout<<"Final result: "<<endl;
    for(int i=0;i<26;i++){
        if(arr[i]>0){

            while(arr[i]){
                char c=i+'a';
                cout<<c;
                arr[i]--;
            }
        }

    }
   return 0;
}
```

e)



```
#include <iostream>
using namespace std;

int main() {
    char ch;
    cout << "Enter a character: ";
    cin >> ch;

    if (ch >= 'A' && ch <= 'Z') {
        ch = ch + 32;
    }

    cout << "Lowercase: " << ch << endl;

    return 0;
}

```

5) Space required to store any two-dimensional array is $number\ of\ rows \times number\ of\ columns$. Assuming an array is used to store elements of the following matrices, implement an efficient way that reduces the space requirement.

(a) Diagonal Matrix.

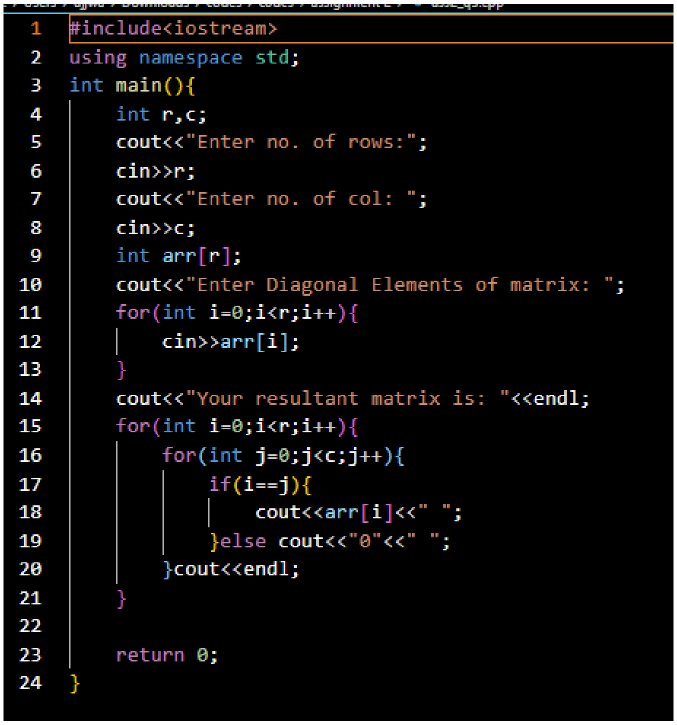
(b) Tri-diagonal Matrix.

(c) Lower triangular Matrix.

(d) Upper triangular Matrix.

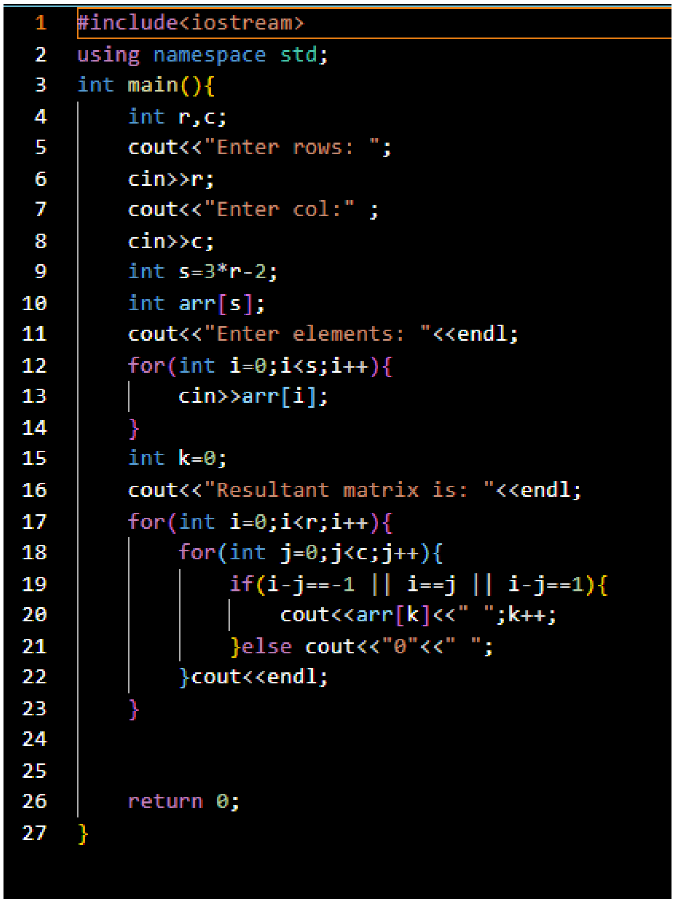
e) Symmetric Matrix

a)



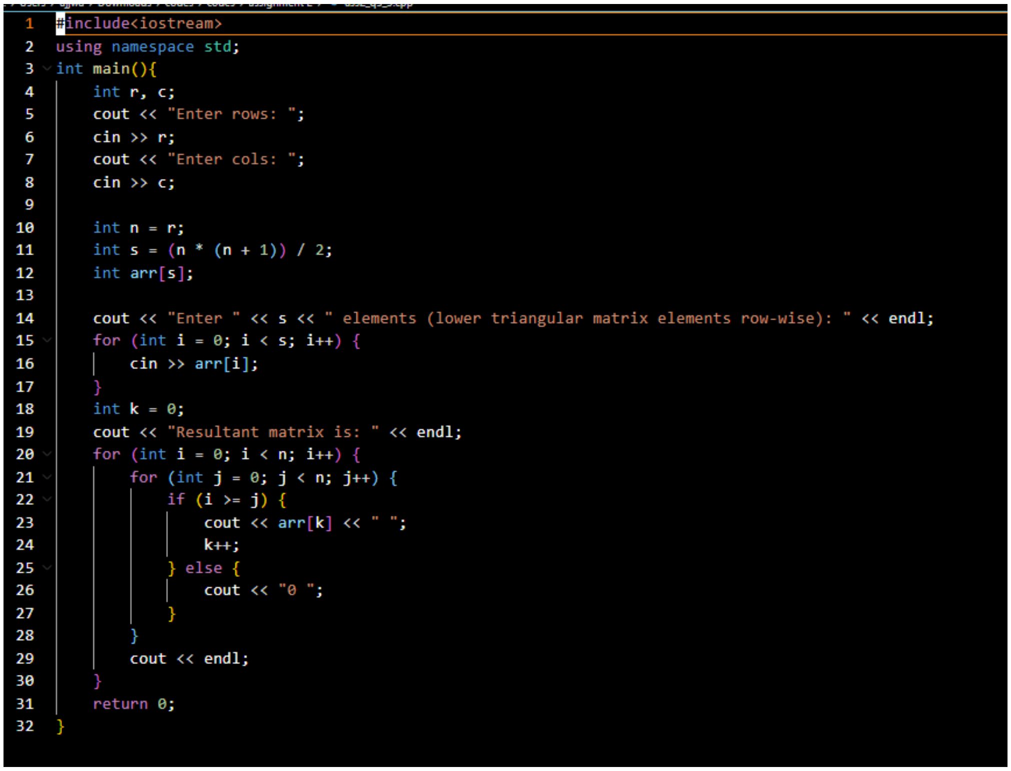
```cpp
#include<iostream>
using namespace std;
int main(){
    int r,c;
    cout<<"Enter no. of rows:";
    cin>>r;
    cout<<"Enter no. of col: ";
    cin>>c;
    int arr[r];
    cout<<"Enter Diagonal Elements of matrix: ";
    for(int i=0;i<r;i++){
        cin>>arr[i];
    }
    cout<<"Your resultant matrix is: "<<endl;
    for(int i=0;i<r;i++){
        for(int j=0;j<c;j++){
            if(i==j){
                cout<<arr[i]<<" ";
            }else cout<<"0"<<" ";
        }cout<<endl;
    }

    return 0;
}
```

b)



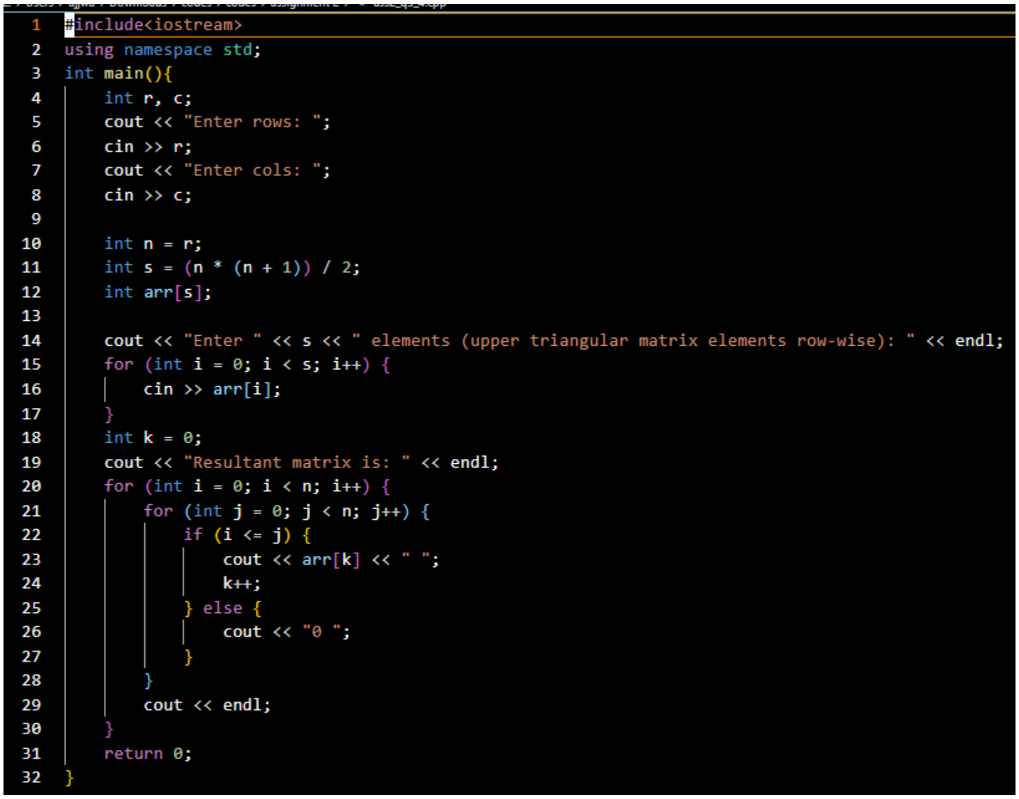
```cpp
#include<iostream>
using namespace std;
int main(){
    int r,c;
    cout<<"Enter rows: ";
    cin>>r;
    cout<<"Enter col:" ;
    cin>>c;
    int s=3*r-2;
    int arr[s];
    cout<<"Enter elements: "<<endl;
    for(int i=0;i<s;i++){
        cin>>arr[i];
    }
    int k=0;
    cout<<"Resultant matrix is: "<<endl;
    for(int i=0;i<r;i++){
        for(int j=0;j<c;j++){
            if(i-j==-1 || i==j || i-j==1){
                cout<<arr[k]<<" ";k++;
            }else cout<<"0"<<" ";
        }cout<<endl;
    }


    return 0;
}
```

c)



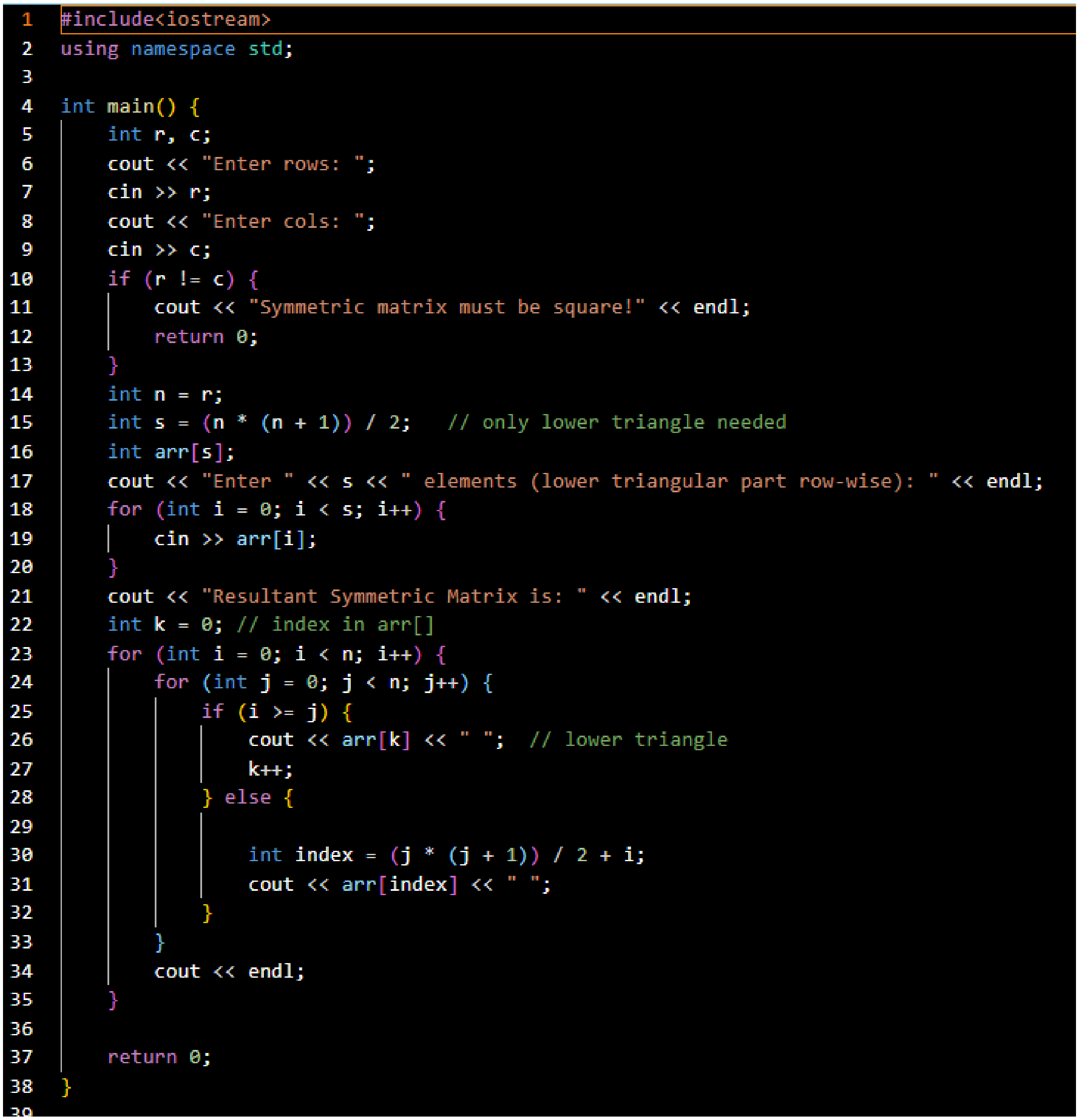
```
#include<iostream>
using namespace std;
int main(){
    int r, c;
    cout << "Enter rows: ";
    cin >> r;
    cout << "Enter cols: ";
    cin >> c;

    int n = r;
    int s = (n * (n + 1)) / 2;
    int arr[s];

    cout << "Enter " << s << " elements (lower triangular matrix elements row-wise): " << endl;
    for (int i = 0; i < s; i++) {
        cin >> arr[i];
    }
    int k = 0;
    cout << "Resultant matrix is: " << endl;
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            if (i >= j) {
                cout << arr[k] << " ";
                k++;
            } else {
                cout << "0 ";
            }
        }
        cout << endl;
    }
    return 0;
}
```

d)



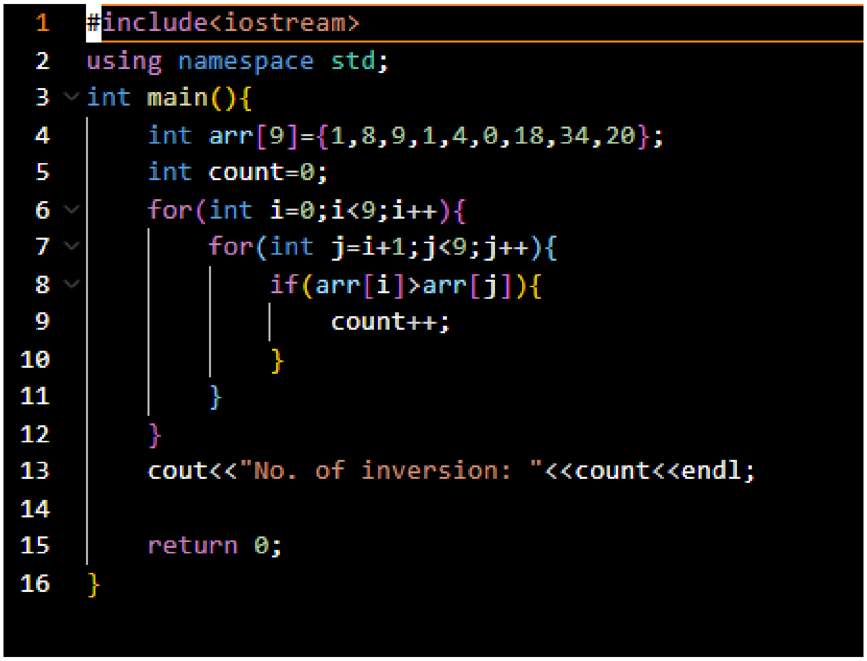
```
#include<iostream>
using namespace std;
int main(){
    int r, c;
    cout << "Enter rows: ";
    cin >> r;
    cout << "Enter cols: ";
    cin >> c;

    int n = r;
    int s = (n * (n + 1)) / 2;
    int arr[s];

    cout << "Enter " << s << " elements (upper triangular matrix elements row-wise): " << endl;
    for (int i = 0; i < s; i++) {
        cin >> arr[i];
    }
    int k = 0;
    cout << "Resultant matrix is: " << endl;
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            if (i <= j) {
                cout << arr[k] << " ";
                k++;
            } else {
                cout << "0 ";
            }
        }
        cout << endl;
    }
    return 0;
}
```

e)



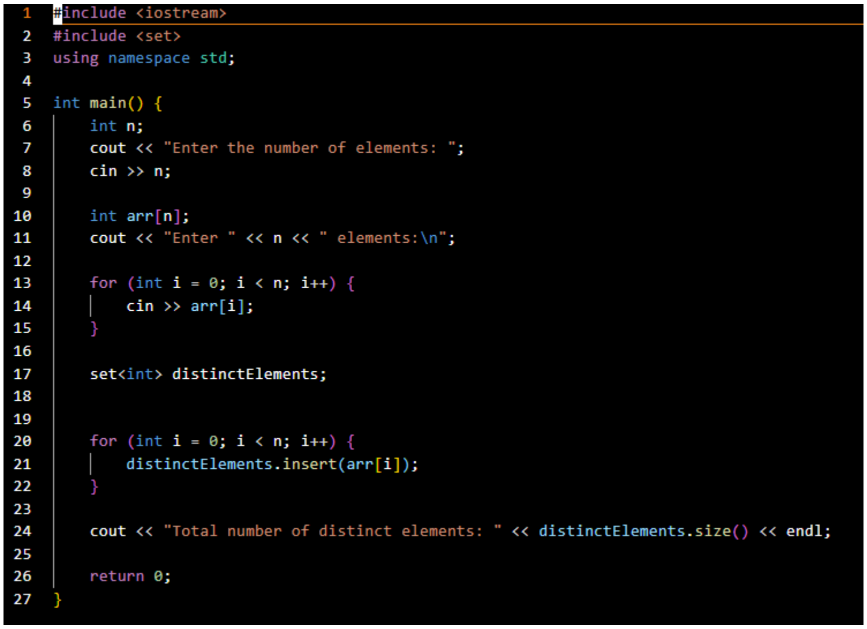
```cpp
#include<iostream>
using namespace std;

int main() {
    int r, c;
    cout << "Enter rows: ";
    cin >> r;
    cout << "Enter cols: ";
    cin >> c;
    if (r != c) {
        cout << "Symmetric matrix must be square!" << endl;
        return 0;
    }
    int n = r;
    int s = (n * (n + 1)) / 2;   // only lower triangle needed
    int arr[s];
    cout << "Enter " << s << " elements (lower triangular part row-wise): " << endl;
    for (int i = 0; i < s; i++) {
        cin >> arr[i];
    }
    cout << "Resultant Symmetric Matrix is: " << endl;
    int k = 0; // index in arr[]
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            if (i >= j) {
                cout << arr[k] << " ";  // lower triangle
                k++;
            } else {

                int index = (j * (j + 1)) / 2 + i;
                cout << arr[index] << " ";
            }
        }
        cout << endl;
    }

    return 0;
}
```

QUE7: Let A[1 .... n] be an array of n real numbers. A pair (A[i], A[j ]) is said to be an inversion if these numbers are out of order, i.e., i < j but A[i]>A[j ]. Write a program to count the number of inversions in an array



```cpp
#include<iostream>
using namespace std;
int main(){
    int arr[9]={1,8,9,1,4,0,18,34,20};
    int count=0;
    for(int i=0;i<9;i++){
        for(int j=i+1;j<9;j++){
            if(arr[i]>arr[j]){
                count++;
            }
        }
    }
    cout<<"No. of inversion: "<<count<<endl;

    return 0;
}
```

QUE8: Write a program to count the total number of distinct elements in an array of length n.



```cpp
#include <iostream>
#include <set>
using namespace std;

int main() {
    int n;
    cout << "Enter the number of elements: ";
    cin >> n;

    int arr[n];
    cout << "Enter " << n << " elements:\n";

    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    set<int> distinctElements;


    for (int i = 0; i < n; i++) {
        distinctElements.insert(arr[i]);
    }

    cout << "Total number of distinct elements: " << distinctElements.size() << endl;

    return 0;
}
```