**AQ1**

#include <iostream>

using namespace std;

int main()

{

    int n, x;

    cout << "Enter size of array: ";

    cin >> n;

    int arr[n];

    cout << "Enter elements in sorted order: ";

    for (int i = 0; i < n; i++)

    {

        cin >> arr[i];

    }

    cout << "Enter element to search: ";

    cin >> x;

    int l=0,r=n-1,mid,f=-1;

    while (l<=r)

    {

        mid=(l+r)/2;

        if (arr[mid]==x)

        {

            f=mid;

            break;

        }

        else if(arr[mid]<x)

        {

            l=mid+1;

        }

        else

        {

            r=mid-1;

        }

    }

    if (f!=-1)

    {

        cout << "Element found at index: "<<f<<endl;

    }

    else

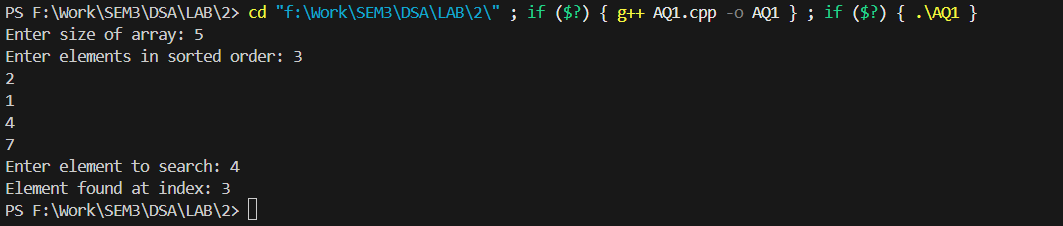
    {

        cout <<"Element not found"<<endl;

    }

}

**Output:**

****

**AQ2**

#include <iostream>

using namespace std;

int main()

{

    int arr[]={64, 34, 25, 12, 22, 11, 90};

    int n=7;

    for (int i=0;i<n-1;i++)

    {

        for (int j=0;j<n-i-1;j++)

        {

            if (arr[j]>arr[j+1])

            {

                int t = arr[j];

                arr[j] = arr[j+1];

                arr[j+1] =t;

            }

        }

    }

    cout << "Sorted array: ";

    for (int i=0; i<n;i++)

    {

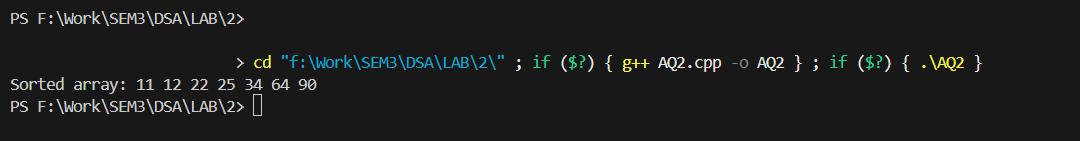
        cout <<arr[i]<<" ";

    }

    cout << endl;

}

**Output:**

****

**AQ3:**

#include <iostream>

#include <algorithm>

using namespace std;

int main() {

    int n;

    cout << "Enter the value of n (range 1 to n): ";

    cin >> n;

    int arr[n-1];

    cout << "Enter " << n-1 << " elements of the array: ";

    for (int i = 0; i < n-1; i++) {

        cin >> arr[i];

    }

    // Linear search

    int missing = -1;

    for (int i = 1; i <= n; i++) {

        bool found = false;

        for (int j = 0; j < n-1; j++) {

            if (arr[j] == i) {

                found = true;

                break;

            }

        }

        if (!found) {

            missing = i;

            break;

        }

    }

    cout << "Missing number: " << missing << endl;

    // Binary search

    int low = 0, high = n - 2;

    int missing2 = -1;

    while (low <= high) {

        int mid = (low + high) / 2;

        if (arr[mid] == mid + 1) {

            low = mid + 1;

        } else {

            high = mid - 1;

        }

    }

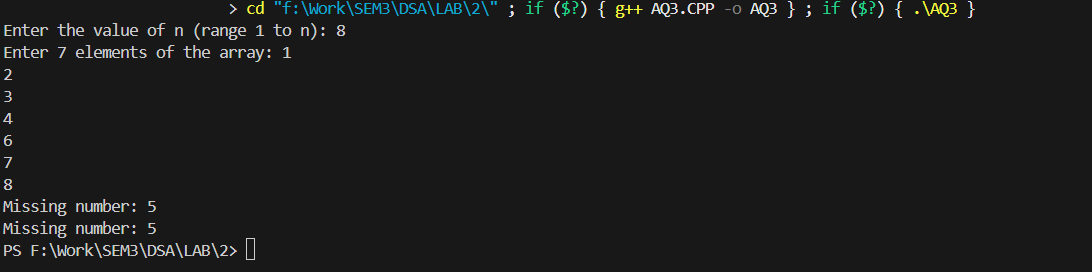
    missing2 = low + 1;

    cout << "Missing number: " << missing2 << endl;

    return 0;

}

**Output:**

****

**AQ4:**

#include <iostream>

using namespace std;

int main()

{

    cout << "Part (a) \n";

    char s1[200], s2[100];

    cout << "Enter the first string: ";

    cin.getline(s1, 200);

    cout << "Enter the second string: ";

    cin.getline(s2, 100);

    int i=0, j=0;

    while(s1[i] != '\0')

    {

        i++;

    }

    while(s2[j] != '\0')

    {

        s1[i]=s2[j];

        i++;

        j++;

    }

    s1[i]='\0';

    cout << "Concatenated string: " << s1 << endl;

    cout << "Part (b) \n";

    char r[200];

    cout << "Enter a string to reverse: ";

    cin.getline(r, 200);

    int n=0;

    while(r[n] != '\0')

    {

        n++;

    }

    for(int k=0; k < n / 2; k++)

    {

        char t=r[k];

        r[k]=r[n-k-1];

        r[n-k-1]=t;

    }

    cout << "Reversed string: " << r << endl;

    cout << "Part (c) \n";

    char s[200];

    cout << "Enter a string to remove vowels: ";

    cin.getline(s, 200);

    j=0;

    for(int k=0; s[k] != '\0'; k++)

    {

        char c=s[k];

        if ((c != 'a' && c != 'A' && c != 'e' && c != 'E' && c != 'i' && c != 'I' && c != 'o' && c != 'O' && c != 'u' && c != 'U'))

        {

            s[j]=s[k];

            j++;

        }

    }

    s[j]='\0';

    cout << "String without vowels: " << s << endl;

    cout << "Part (d) \n";

    int m;

    cout << "Enter number of strings to sort: ";

    cin >> m;

    cin.ignore();

    char arr[m][100];

    for(int k=0; k < m; k++)

    {

        cout << "Enter string " << k + 1 << ": ";

        cin.getline(arr[k], 100);

    }

    for(i=0;i<m-1;i++)

    {

        for(j=i+1;j<m;j++)

        {

            int k=0;

            while(arr[i][k] != '\0' && arr[j][k] != '\0' && arr[i][k] == arr[j][k])

            {

                k++;

            }

            if (arr[i][k] > arr[j][k])

            {

                char temp[100];

                int p=0;

                while(arr[i][p] != '\0')

                {

                    temp[p]=arr[i][p];

                    p++;

                }

                temp[p]='\0';

                p=0;

                while(arr[j][p] != '\0')

                {

                    arr[i][p]=arr[j][p];

                    p++;

                }

                arr[i][p]='\0';

                p=0;

                while(temp[p] != '\0')

                {

                    arr[j][p]=temp[p];

                    p++;

                }

                arr[j][p]='\0';

            }

        }

    }

    cout << "Sorted strings:\n";

    for(int i=0; i < m; i++)

    {

        cout << arr[i] << endl;

    }

    cout << "Part (e) \n";

    char str1[200];

    cout << "Enter a string in Uppercase to convert to Lowercase: ";

    cin.getline(str1, 200);

    j=0;

    while(str1[j] != '\0')

    {

        if (str1[j] >= 'A' && str1[j] <= 'Z')

        {

            str1[j]=str1[j] + 32;

        }

        j++;

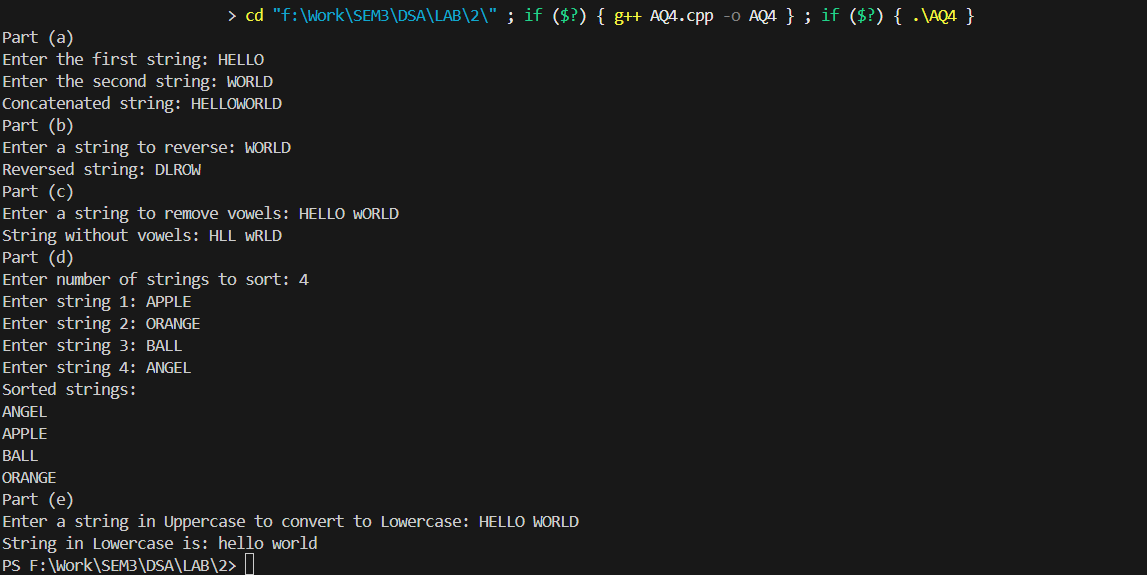
    }

    cout << "String in Lowercase is: " << str1;

    return 0;

}

**Output:**

****

**AQ5:**

#include <iostream>

using namespace std;

int main() {

    int n;

    cout << "Enter size of Diagonal Matrix n: ";

    cin >> n;

    int diagMat[n][n];

    cout << "Enter the " << n << "x" << n << " diagonal matrix:\n";

    for(int i=0;i<n;i++)

    {

        for(int j=0;j<n;j++)

        {

            cin >> diagMat[i][j];

        }

    }

    cout << "Input Matrix:\n";

    for(int i=0;i<n;i++)

    {

        for(int j=0;j<n;j++)

        {

            cout << diagMat[i][j] << " ";

        }

        cout << "\n";

    }

    int diag[n];

    for(int i=0;i<n;i++)

    {

        diag[i] = diagMat[i][i];

    }

    cout << "Linear storage (Diagonal): ";

    for(int i=0;i<n;i++)

    {

        cout << diag[i] << " ";

    }

    cout << "\n\n";

    cout << "Enter size of Tri-diagonal Matrix n: ";

    cin >> n;

    int triMat[n][n];

    cout << "Enter the " << n << "x" << n << " matrix:\n";

    for(int i=0;i<n;i++)

    {

        for(int j=0;j<n;j++)

        {

            cin >> triMat[i][j];

        }

    }

    cout << "Input Matrix:\n";

    for(int i=0;i<n;i++)

    {

        for(int j=0;j<n;j++)

        {

            cout << triMat[i][j] << " ";

        }

        cout << "\n";

    }

    int tri[3\*n - 2], k=0;

    for(int i=0;i<n;i++)

    {

        for(int j=0;j<n;j++)

        {

            if(i==j || i==j+1 || j==i+1)

            {

                tri[k++] = triMat[i][j];

            }

        }

    }

    cout << "Linear storage (Tri-diagonal): ";

    for(int i=0;i<3\*n-2;i++) cout << tri[i] << " ";

    cout << "\n\n";

    cout << "Enter size of Lower Triangular Matrix n: ";

    cin >> n;

    int lowMat[n][n];

    cout << "Enter the " << n << "x" << n << " matrix:\n";

    for(int i=0;i<n;i++)

    {

        for(int j=0;j<n;j++)

        {

            cin >> lowMat[i][j];

        }

    }

    cout << "Input Matrix:\n";

    for(int i=0;i<n;i++)

    {

        for(int j=0;j<n;j++)

        {

            cout << lowMat[i][j] << " ";

        }

        cout << "\n";

    }

    int lower[n\*(n+1)/2];

    k=0;

    for(int i=0;i<n;i++)

    {

        for(int j=0;j<=i;j++)

        {

            lower[k++] = lowMat[i][j];

        }

    }

    cout << "Linear storage (Lower Triangular): ";

    for(int i=0;i<n\*(n+1)/2;i++)

    {

        cout << lower[i] << " ";

    }

    cout << "\n\n";

    cout << "Enter size of Upper Triangular Matrix n: ";

    cin >> n;

    int upMat[n][n];

    cout << "Enter the " << n << "x" << n << " matrix:\n";

    for(int i=0;i<n;i++)

    {

        for(int j=0;j<n;j++)

        {

            cin >> upMat[i][j];

        }

    }

    cout << "Input Matrix:\n";

    for(int i=0;i<n;i++)

    {

        for(int j=0;j<n;j++)

        {

            cout << upMat[i][j] << " ";

        }

        cout << "\n";

    }

    int upper[n\*(n+1)/2];

    k=0;

    for(int i=0;i<n;i++)

    {

        for(int j=i;j<n;j++)

        {

            upper[k++] = upMat[i][j];

        }

    }

    cout << "Linear storage (Upper Triangular): ";

    for(int i=0;i<n\*(n+1)/2;i++)

    {

        cout << upper[i] << " ";

    }

    cout << "\n\n";

    cout << "Enter size of Symmetric Matrix n: ";

    cin >> n;

    int symMat[n][n];

    cout << "Enter the " << n << "x" << n << " matrix:\n";

    for(int i=0;i<n;i++)

    {

        for(int j=0;j<n;j++)

        {

            cin >> symMat[i][j];

        }

    }

    cout << "Input Matrix:\n";

    for(int i=0;i<n;i++)

    {

        for(int j=0;j<n;j++)

        {

            cout << symMat[i][j] << " ";

        }

        cout << "\n";

    }

    int sym[n\*(n+1)/2];

    k=0;

    for(int i=0;i<n;i++)

    {

        for(int j=0;j<=i;j++)

        {

            sym[k++] = symMat[i][j];

        }

    }

    cout << "Linear storage (Symmetric): ";

    for(int i=0;i<n\*(n+1)/2;i++)

    {

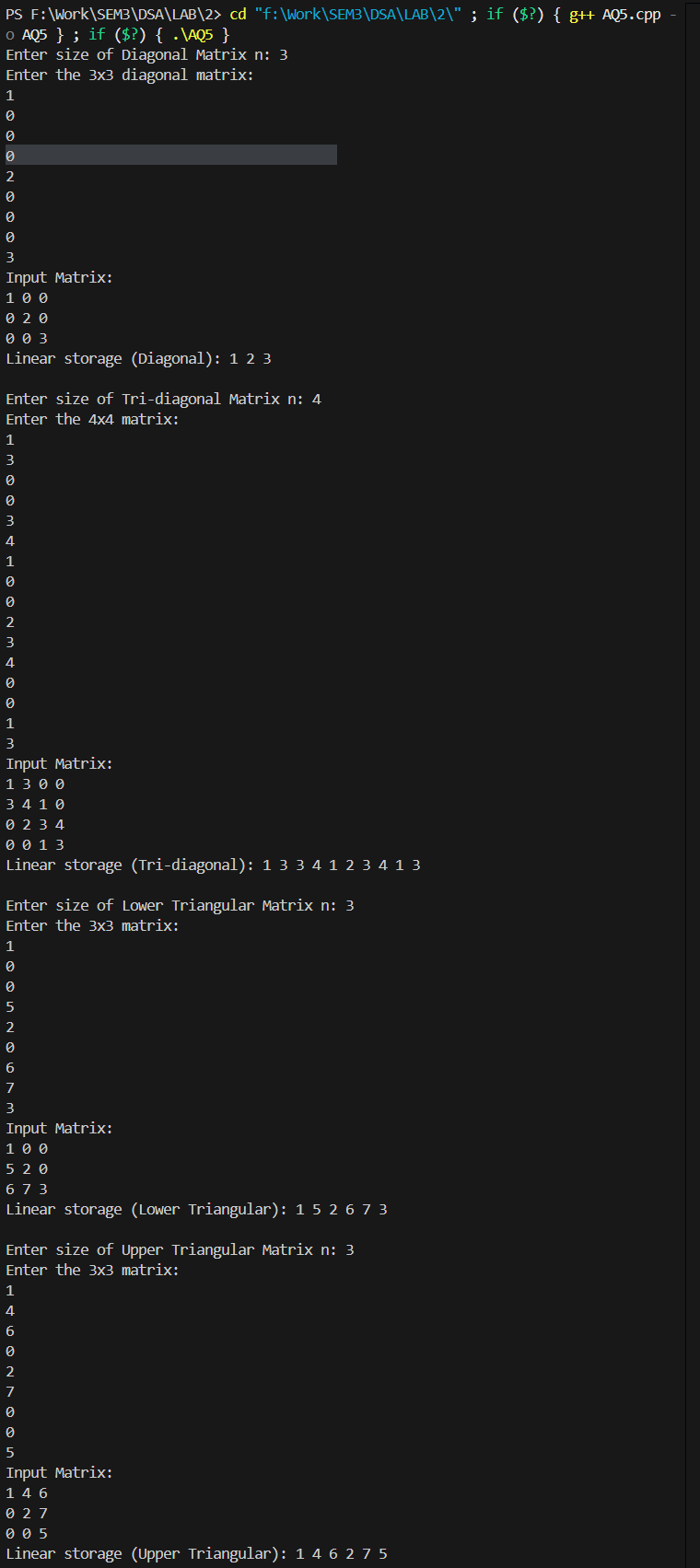
        cout << sym[i] << " ";

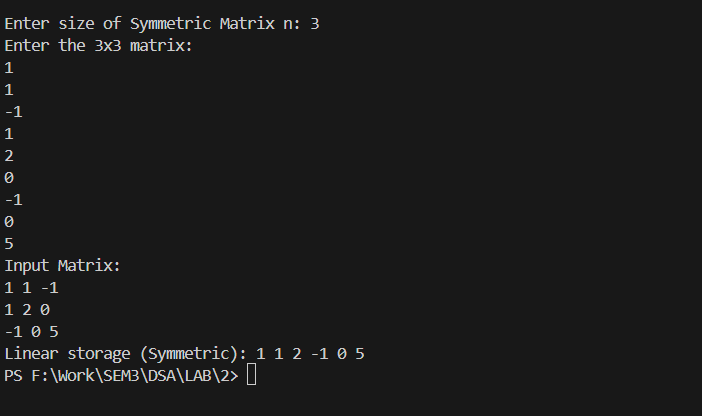
    }

    cout << "\n";

    return 0;

}

**Output:**

****

**AQ6**

#include <iostream>

using namespace std;

int main()

{

    int r, c;

    int size = 0;

    cout << "Part (a) \n";

    cout << "Enter Row and Column of Original Matrix:" << endl;

    cin >> r >> c;

    int arr[r][c];

    cout << "Enter Matrix elements:" << endl;

    for (int i = 0; i < r; i++)

    {

        for (int j = 0; j < c; j++)

        {

            cin >> arr[i][j];

            if ((arr[i][j]) != 0)

            {

                size++;

            }

        }

    }

    int sparse\_arr[3][size];

    cout << "Original Matrix is:" << endl;

    for (int i = 0; i < r; i++)

    {

        for (int j = 0; j < c; j++)

        {

            cout << arr[i][j] << " ";

        }

        cout << endl;

    }

    int k = 0;

    for (int i = 0; i < r; i++)

    {

        for (int j = 0; j < c; j++)

        {

            if (arr[i][j] != 0)

            {

                sparse\_arr[0][k] = i;

                sparse\_arr[1][k] = j;

                sparse\_arr[2][k] = arr[i][j];

                k++;

            }

        }

    }

    cout << "Sparse Matrix is:" << endl;

    for (int i = 0; i < 3; i++)

    {

        for (int j = 0; j < size; j++)

        {

            cout << sparse\_arr[i][j] << " ";

        }

        cout << endl;

    }

    for (int i = 0; i < size; i++)

    {

        int temp = sparse\_arr[0][i];

        sparse\_arr[0][i] = sparse\_arr[1][i];

        sparse\_arr[1][i] = temp;

    }

    cout << "Transpose of Sparse Matrix is:" << endl;

    for (int i = 0; i < 3; i++)

    {

        for (int j = 0; j < size; j++)

        {

            cout << sparse\_arr[i][j] << " ";

        }

        cout << endl;

    }

    cout << "Part (b) \n";

    cout << "Enter Row and Column of Matrices:" << endl;

    cin >> r >> c;

    int size\_A = 0, size\_B = 0;

    int A[r][c];

    int B[r][c];

    cout << "Enter Matrix A elements:" << endl;

    for (int i = 0; i < r; i++)

    {

        for (int j = 0; j < c; j++)

        {

            cin >> A[i][j];

            if ((A[i][j]) != 0)

            {

                size\_A++;

            }

        }

    }

    int sparse\_A[3][size\_A];

    cout << "Enter Matrix B elements:" << endl;

    for (int i = 0; i < r; i++)

    {

        for (int j = 0; j < c; j++)

        {

            cin >> B[i][j];

            if ((B[i][j]) != 0)

            {

                size\_B++;

            }

        }

    }

    int sparse\_B[3][size\_B];

    cout << "Original Matrix A is:" << endl;

    for (int i = 0; i < r; i++)

    {

        for (int j = 0; j < c; j++)

        {

            cout << A[i][j] << " ";

        }

        cout << endl;

    }

    cout << "Original Matrix B is:" << endl;

    for (int i = 0; i < r; i++)

    {

        for (int j = 0; j < c; j++)

        {

            cout << B[i][j] << " ";

        }

        cout << endl;

    }

    k = 0;

    for (int i = 0; i < r; i++)

    {

        for (int j = 0; j < c; j++)

        {

            if (A[i][j] != 0)

            {

                sparse\_A[0][k] = i;

                sparse\_A[1][k] = j;

                sparse\_A[2][k] = A[i][j];

                k++;

            }

        }

    }

    cout << "Sparse of Matrix A is:" << endl;

    for (int i = 0; i < 3; i++)

    {

        for (int j = 0; j < size\_A; j++)

        {

            cout << sparse\_A[i][j] << " ";

        }

        cout << endl;

    }

    k = 0;

    for (int i = 0; i < r; i++)

    {

        for (int j = 0; j < c; j++)

        {

            if (B[i][j] != 0)

            {

                sparse\_B[0][k] = i;

                sparse\_B[1][k] = j;

                sparse\_B[2][k] = B[i][j];

                k++;

            }

        }

    }

    cout << "Sparse of Matrix B is:" << endl;

    for (int i = 0; i < 3; i++)

    {

        for (int j = 0; j < size\_B; j++)

        {

            cout << sparse\_B[i][j] << " ";

        }

        cout << endl;

    }

    int size\_Sum = 0;

    int Sum[3][size\_A + size\_B];

    int i = 0, j = 0;

    k = 0;

    while (i < size\_A && j < size\_B)

    {

        if (sparse\_A[0][i] < sparse\_B[0][j])

        {

            Sum[0][k] = sparse\_A[0][i];

            Sum[1][k] = sparse\_A[1][i];

            Sum[2][k] = sparse\_A[2][i];

            i++;

            k++;

        }

        else if (sparse\_B[0][j] < sparse\_A[0][i])

        {

            Sum[0][k] = sparse\_B[0][j];

            Sum[1][k] = sparse\_B[1][j];

            Sum[2][k] = sparse\_B[2][j];

            j++;

            k++;

        }

        else

        {

            if (sparse\_A[1][i] < sparse\_B[1][j])

            {

                Sum[0][k] = sparse\_A[0][i];

                Sum[1][k] = sparse\_A[1][i];

                Sum[2][k] = sparse\_A[2][i];

                i++;

                k++;

            }

            else if (sparse\_B[1][j] < sparse\_A[1][i])

            {

                Sum[0][k] = sparse\_B[0][j];

                Sum[1][k] = sparse\_B[1][j];

                Sum[2][k] = sparse\_B[2][j];

                j++;

                k++;

            }

            else

            {

                Sum[0][k] = sparse\_A[0][i];

                Sum[1][k] = sparse\_A[1][i];

                Sum[2][k] = sparse\_A[2][i] + sparse\_B[2][j];

                i++;

                j++;

                k++;

            }

        }

    }

    while (i < size\_A)

    {

        Sum[0][k] = sparse\_A[0][i];

        Sum[1][k] = sparse\_A[1][i];

        Sum[2][k] = sparse\_A[2][i];

        i++;

        k++;

    }

    while (j < size\_B)

    {

        Sum[0][k] = sparse\_B[0][j];

        Sum[1][k] = sparse\_B[1][j];

        Sum[2][k] = sparse\_B[2][j];

        j++;

        k++;

    }

    size\_Sum = k;

    cout << "Addition of Sparse A and B is:" << endl;

    for (i = 0; i < 3; i++)

    {

        for (j = 0; j < size\_Sum; j++)

        {

            cout << Sum[i][j] << " ";

        }

        cout << endl;

    }

    cout << "Part (c) \n";

    int r1, c1, r2, c2;

    cout << "Enter Row and Column of Matrix X:" << endl;

    cin >> r1 >> c1;

    cout << "Enter Row and Column of Matrix Y:" << endl;

    cin >> r2 >> c2;

    if (c1 == r2)

    {

        int X[r1][c1], Y[r2][c2];

        int size\_X = 0, size\_Y = 0;

        cout << "Enter Matrix X elements:" << endl;

        for (int i = 0; i < r1; i++)

        {

            for (int j = 0; j < c1; j++)

            {

                cin >> X[i][j];

                if ((X[i][j]) != 0)

                {

                    size\_X++;

                }

            }

        }

        int sparse\_X[3][size\_X];

        cout << "Enter Matrix Y elements:" << endl;

        for (int i = 0; i < r2; i++)

        {

            for (int j = 0; j < c2; j++)

            {

                cin >> Y[i][j];

                if ((Y[i][j]) != 0)

                {

                    size\_Y++;

                }

            }

        }

        int sparse\_Y[3][size\_Y];

        cout << "Original Matrix X is:" << endl;

        for (int i = 0; i < r1; i++)

        {

            for (int j = 0; j < c1; j++)

            {

                cout << X[i][j] << " ";

            }

            cout << endl;

        }

        cout << "Original Matrix Y is:" << endl;

        for (int i = 0; i < r2; i++)

        {

            for (int j = 0; j < c2; j++)

            {

                cout << Y[i][j] << " ";

            }

            cout << endl;

        }

        k = 0;

        for (int i = 0; i < r1; i++)

        {

            for (int j = 0; j < c1; j++)

            {

                if (X[i][j] != 0)

                {

                    sparse\_X[0][k] = i;

                    sparse\_X[1][k] = j;

                    sparse\_X[2][k] = X[i][j];

                    k++;

                }

            }

        }

        cout << "Sparse of Matrix X is:" << endl;

        for (int i = 0; i < 3; i++)

        {

            for (int j = 0; j < size\_X; j++)

            {

                cout << sparse\_X[i][j] << " ";

            }

            cout << endl;

        }

        k = 0;

        for (int i = 0; i < r2; i++)

        {

            for (int j = 0; j < c2; j++)

            {

                if (Y[i][j] != 0)

                {

                    sparse\_Y[0][k] = i;

                    sparse\_Y[1][k] = j;

                    sparse\_Y[2][k] = Y[i][j];

                    k++;

                }

            }

        }

        cout << "Sparse of Matrix Y is:" << endl;

        for (int i = 0; i < 3; i++)

        {

            for (int j = 0; j < size\_Y; j++)

            {

                cout << sparse\_Y[i][j] << " ";

            }

            cout << endl;

        }

        int sparse\_P[3][size\_X \* size\_Y];

        int size\_P = 0;

        for (int i = 0; i < size\_X; i++)

        {

            for (int j = 0; j < size\_Y; j++)

            {

                if (sparse\_X[1][i] == sparse\_Y[0][j])

                {

                    int row = sparse\_X[0][i];

                    int col = sparse\_Y[1][j];

                    int val = sparse\_X[2][i] \* sparse\_Y[2][j];

                    bool found = false;

                    for (int z = 0; z < size\_P; z++)

                    {

                        if (sparse\_P[0][z] == row && sparse\_P[1][z] == col)

                        {

                            sparse\_P[2][z] += val;

                            found = true;

                            break;

                        }

                    }

                    if (!found)

                    {

                        sparse\_P[0][size\_P] = row;

                        sparse\_P[1][size\_P] = col;

                        sparse\_P[2][size\_P] = val;

                        size\_P++;

                    }

                }

            }

        }

        cout << "Sparse of Product is:" << endl;

        for (int i = 0; i < 3; i++)

        {

            for (int j = 0; j < size\_P; j++)

                cout << sparse\_P[i][j] << " ";

            cout << endl;

        }

    }

    else

    {

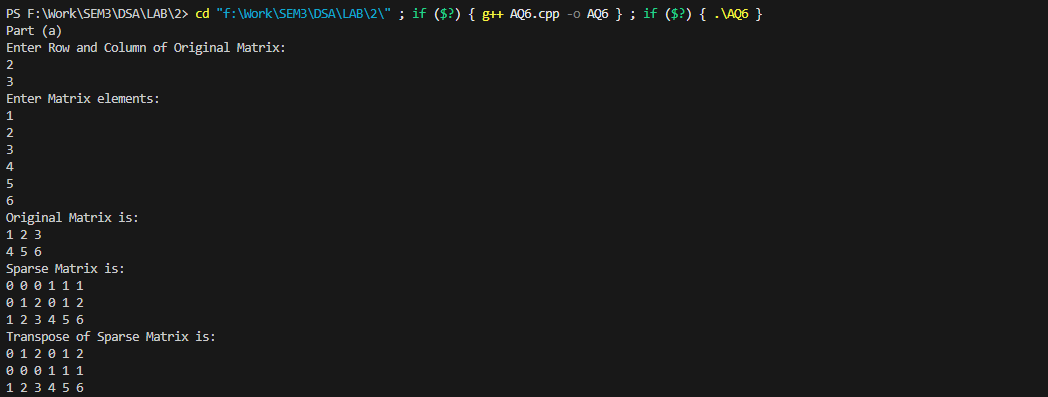
        cout << "Multiplication Not Possible" << endl;

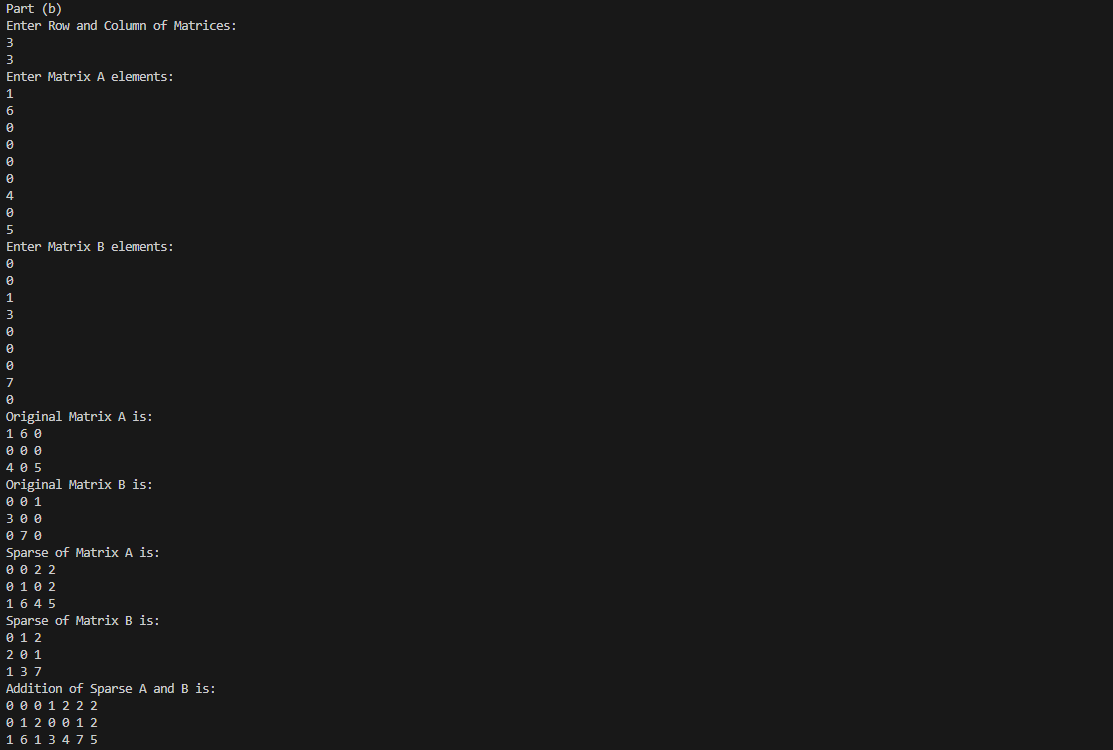
    }

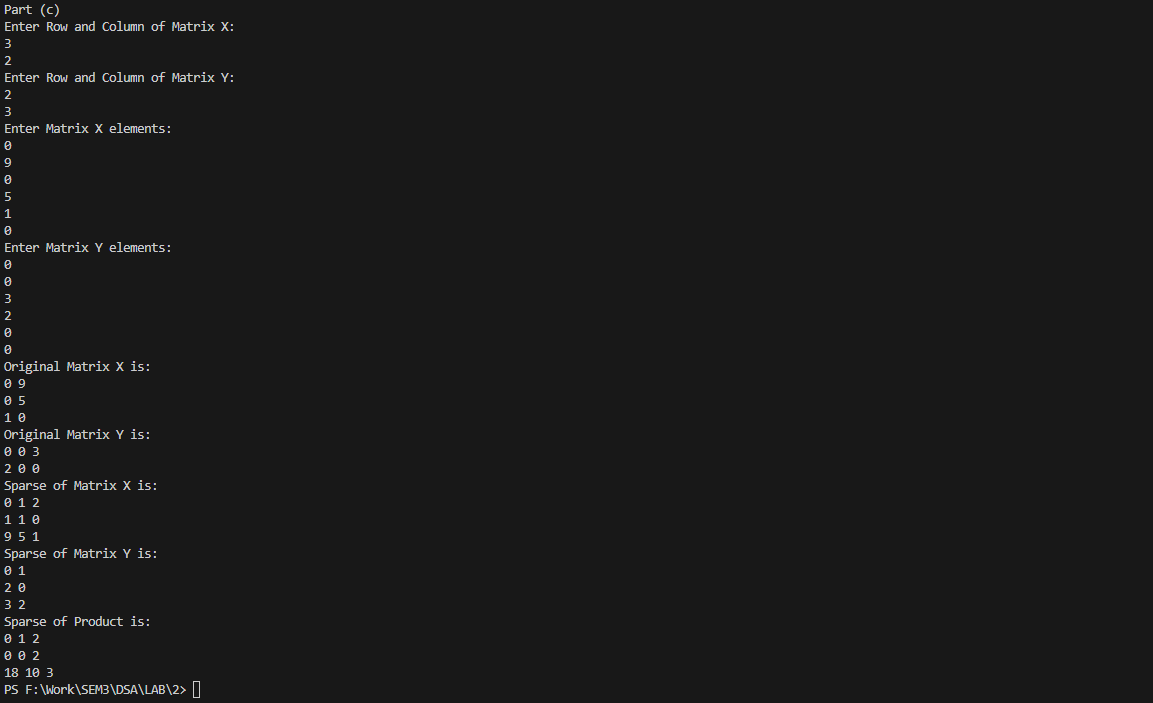
    return 0;

}

**Output:**







**AQ7**

#include <iostream>

using namespace std;

int main()

{

    int n;

    cout << "Enter number of elements: ";

    cin >> n;

    int arr[n];

    cout << "Enter array elements: " << endl;

    for (int i=0;i<n;i++)

    {

        cin >>arr[i];

    }

    cout << "The array is: ";

    for (int i = 0; i < n; i++)

    {

        cout << arr[i] << " ";

    }

    cout << endl;

    int count=0;

    for (int i=0;i<n-1;i++)

    {

        for (int j=i+1;j<n;j++)

        {

            if (arr[i]>arr[j])

            {

                count++;

            }

        }

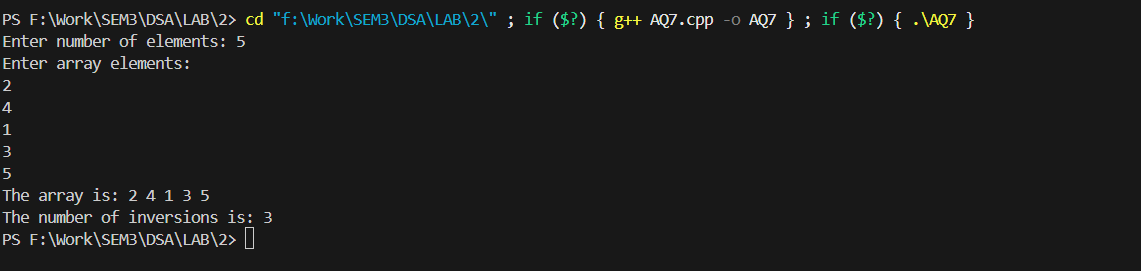
    }

    cout << "The number of inversions is: " << count << endl;

    return 0;

}

**Output:**



**AQ8**

#include <iostream>

using namespace std;

int main()

{

    int n;

    cout << "Enter number of elements: ";

    cin >> n;

    int arr[n];

    cout << "Enter array elements: "<< endl;

    for(int i=0; i < n; i++)

    {

        cin >> arr[i];

    }

    int count=0;

    for(int i=0; i < n; i++)

    {

        int duplicate=0;

        for(int j=0; j < i; j++)

        {

            if (arr[i] == arr[j])

            {

                duplicate=1;

                break;

            }

        }

        if (duplicate == 0)

        {

            count++;

        }

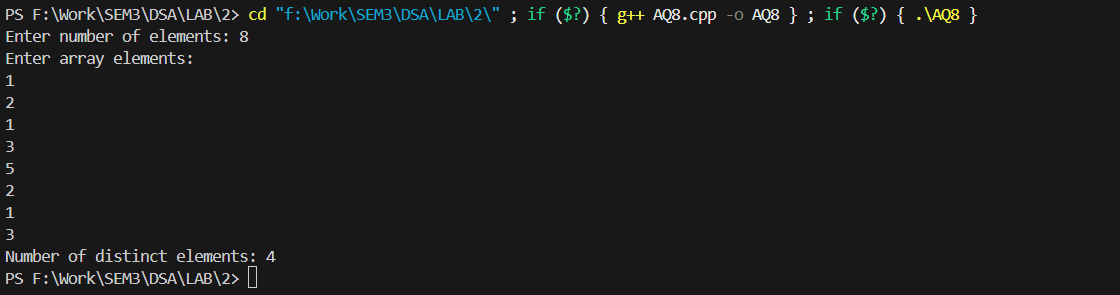
    }

    cout << "Number of distinct elements: " << count << endl;

    return 0;

}

**Output:**

****