LEARN DSA WITH C++

WEEK :: 04

LEARN DSA
WITH C++

PDF

COURSE INSTRUCTOR BY
ROHIT NEGI

MADE BY-
PRADUM SINGHA

Check Profile

My profile

LEARN DSA WITH C++

WEEK :: 04 **DAY: 01** **DATE: 08-05-2023**

TWO POINTER

Two pointers is an effective technique that is typically used for searching pairs in a sorted array.

3 important point for two Pointer :-

1. Where take pointer
2. Move Right Side = Value Increase
3. Move Left Side = Value Decrease

2	3	10	15	19	28

Find sum = 29;

```
First_P = 0, Last_P = n-1;
while(First_P < Last_P )
{
   if( arr[First_P] + arr[Last_P] == sum)
        count << answer;
        return 0;
   else if (arr[ First_P] + arr[Last_P] < sum )
         First_P++;
    else
        Last_P--;
}
```

Find Multiply = 29;

Try Question on yourself

2	3	10	15	19	28

Find Deff = 5;

Take Pointer

First = 0;

Second = 1;

Second move right side = different increase

First move right side = different increase

Container With Most Water << GeeksforGeeks >>

```
long long maxArea(long long A[], int len)
{
    // Your code goes here
    long long sum = 0;
    long long first = 0, last =len -1, length, breadth;

    while(first<last)
    {
        length = last - first;
        if(A[first]> A[last])
        breadth = A[last--];
        else
        breadth = A[first++];

        if(sum<length*breadth)
        sum = length*breadth;

    }
    return sum;
}
```

LEARN DSA WITH C++

WEEK :: 04 **DAY: 02** **DATE: 09-05-2023**

2 D ARRAY IN DETAIL

The two-dimensional array can be defined as an array of arrays.

2	6	8	7	5

1 - D array

2	6	8	7	5
5	5	22	3	7
9	7	7	5	8

2 - D Array

Define int arr[row][col] ;

Array A[3][4];

00	01	02	03
10	11	12	13
20	21	22	23

Store in Memory :: {Row major Order}

	00	01	02	03	10	11	12	13	20	21	22	23
Index =	0	1	2	3	4	5	6	7	8	9	10	11

Find Index = ((row_index * column) + Column_index) - - - - - 1

elem: 22 = (2 * 4) + 2

= 10

# Row_Index = Index / Column;

From 1:- divided by column

(Index/Column) = (Row_Index * Column / Column) + (Column_Index/Column)

Index / Column = row_Index if column = 4, Column_index = 0 to 4;

# Column_Index = Index % Column

From 1:- % by column

Index % Column = (Row_Index * Column) % column + (Column_Index % Column)

Index % Column = 0 + Column_Index

Solve :- 13 =>

row = 7/4=1 col = 7%4=3

Store in Memory {Column Major Order}

	00	10	20	01	11	21	02	12	22	03	13	23
Index =	0	1	2	3	4	5	6	7	8	9	10	11

Find Index = ((Col_index * row) + row_index) - - - - - 1

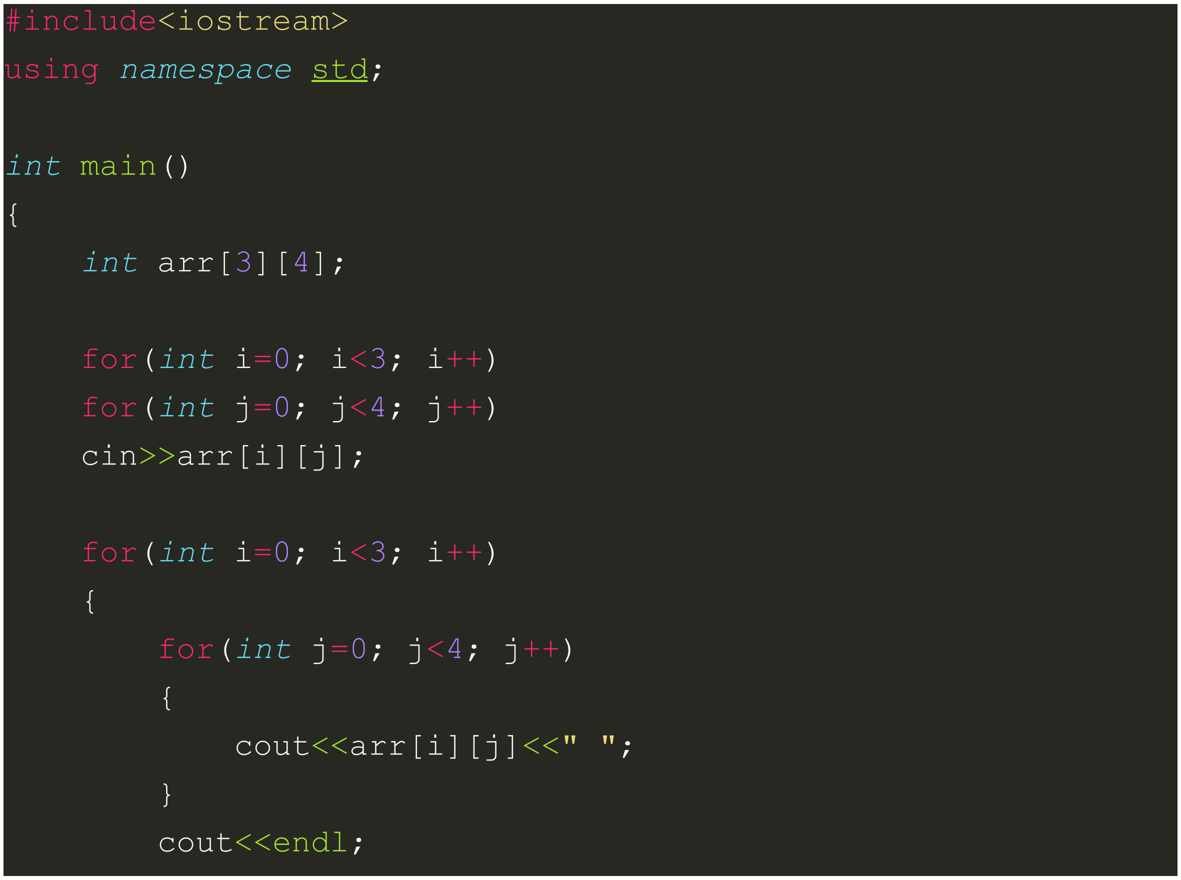
Elem 21 = (1 * 3) + 2 = 5

# Row_Index = Index / row;

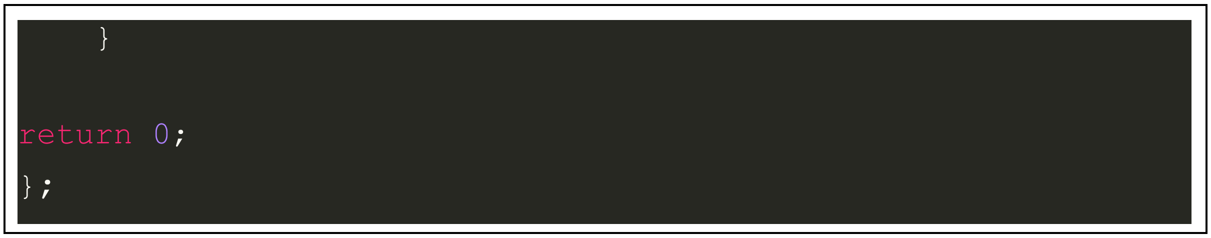
# Column_Index = Index % row;

22 => row_index = 8/3 = 2; coln_index = 8%3 = 2

#Code :: Print Row wise:-

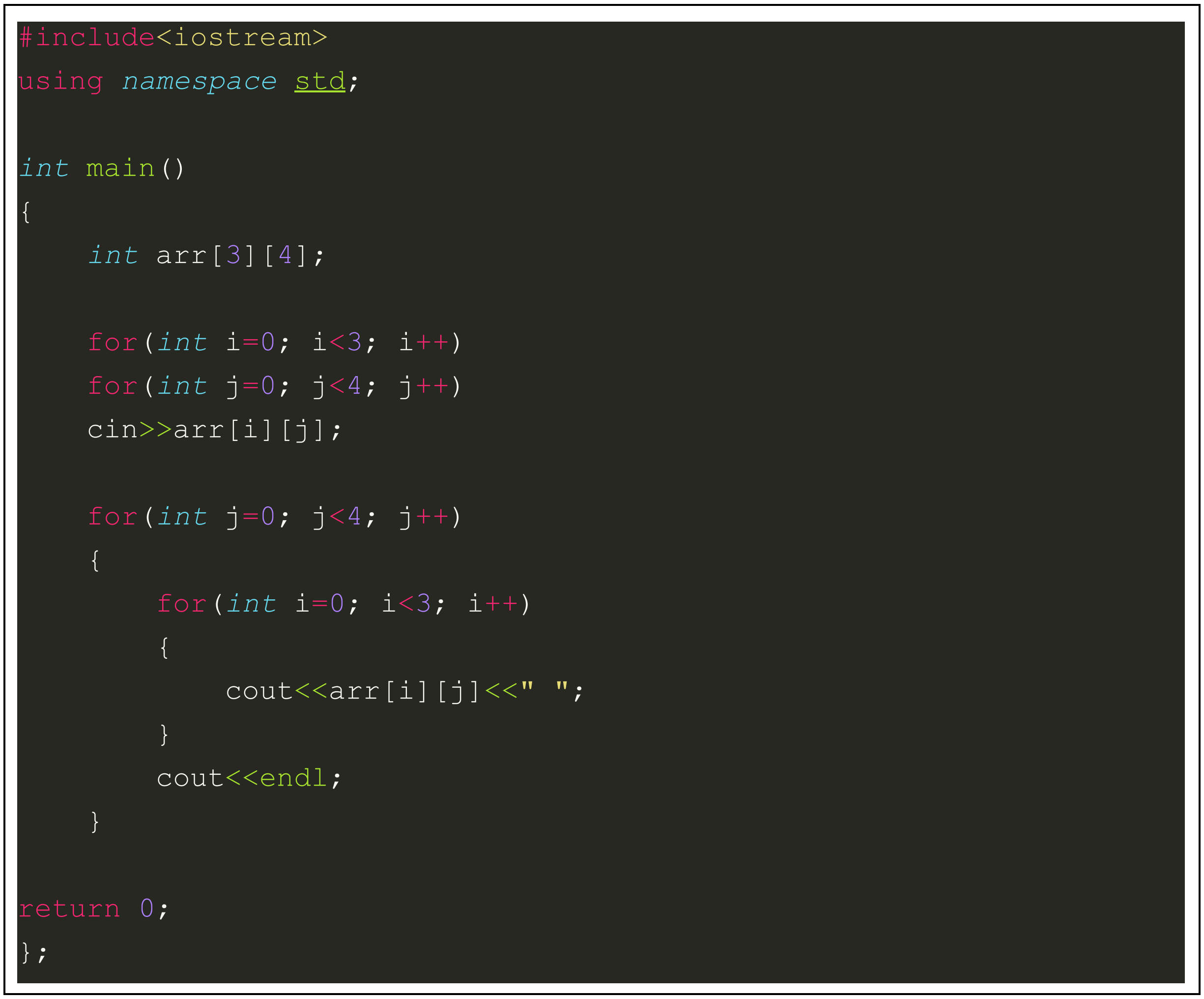


```
#include<iostream>
using namespace std;

int main()
{
    int arr[3][4];

    for(int i=0; i<3; i++)
    for(int j=0; j<4; j++)
    cin>>arr[i][j];

    for(int i=0; i<3; i++)
    {
        for(int j=0; j<4; j++)
        {
            cout<<arr[i][j]<<" ";
        }
        cout<<endl;
```



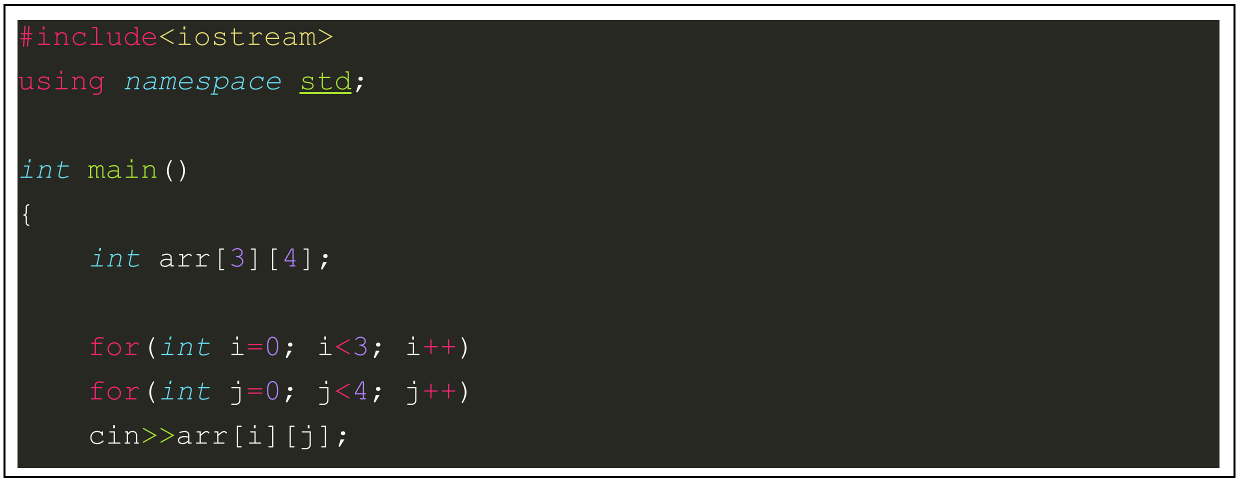
```
    }

return 0;
};
```

# Print column wise ::

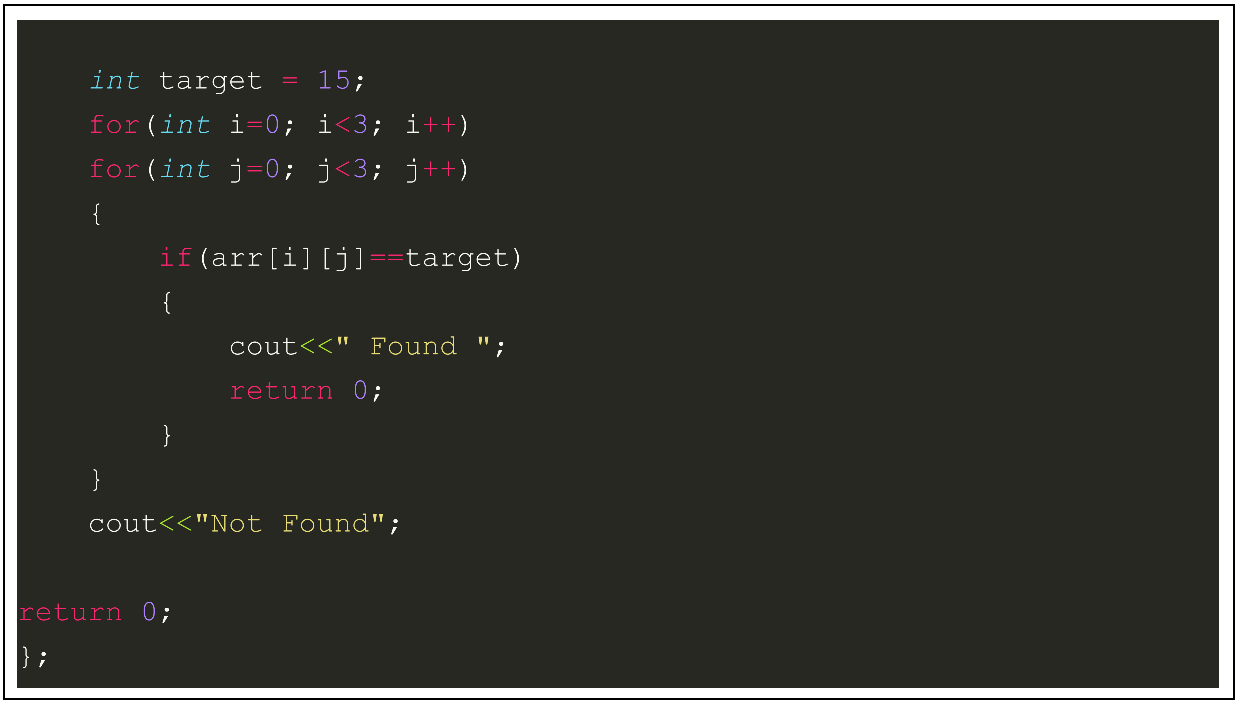


```
#include<iostream>
using namespace std;

int main()
{
    int arr[3][4];

    for(int i=0; i<3; i++)
    for(int j=0; j<4; j++)
    cin>>arr[i][j];

    for(int j=0; j<4; j++)
    {
        for(int i=0; i<3; i++)
        {
            cout<<arr[i][j]<<" ";
        }
        cout<<endl;
    }

return 0;
};
```

# Find Elem In 2 D array ::

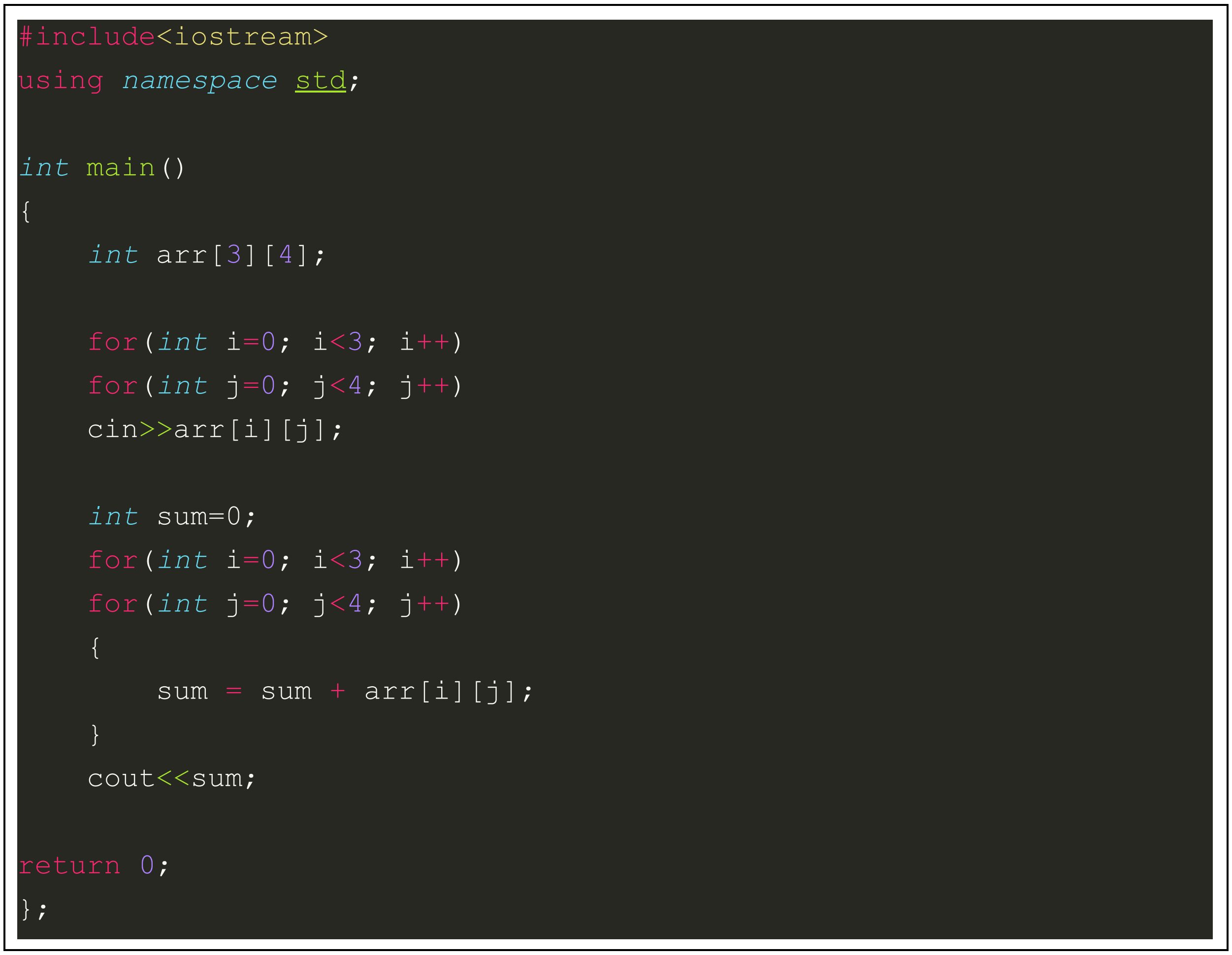


```
#include<iostream>
using namespace std;

int main()
{
    int arr[3][4];

    for(int i=0; i<3; i++)
    for(int j=0; j<4; j++)
    cin>>arr[i][j];
```



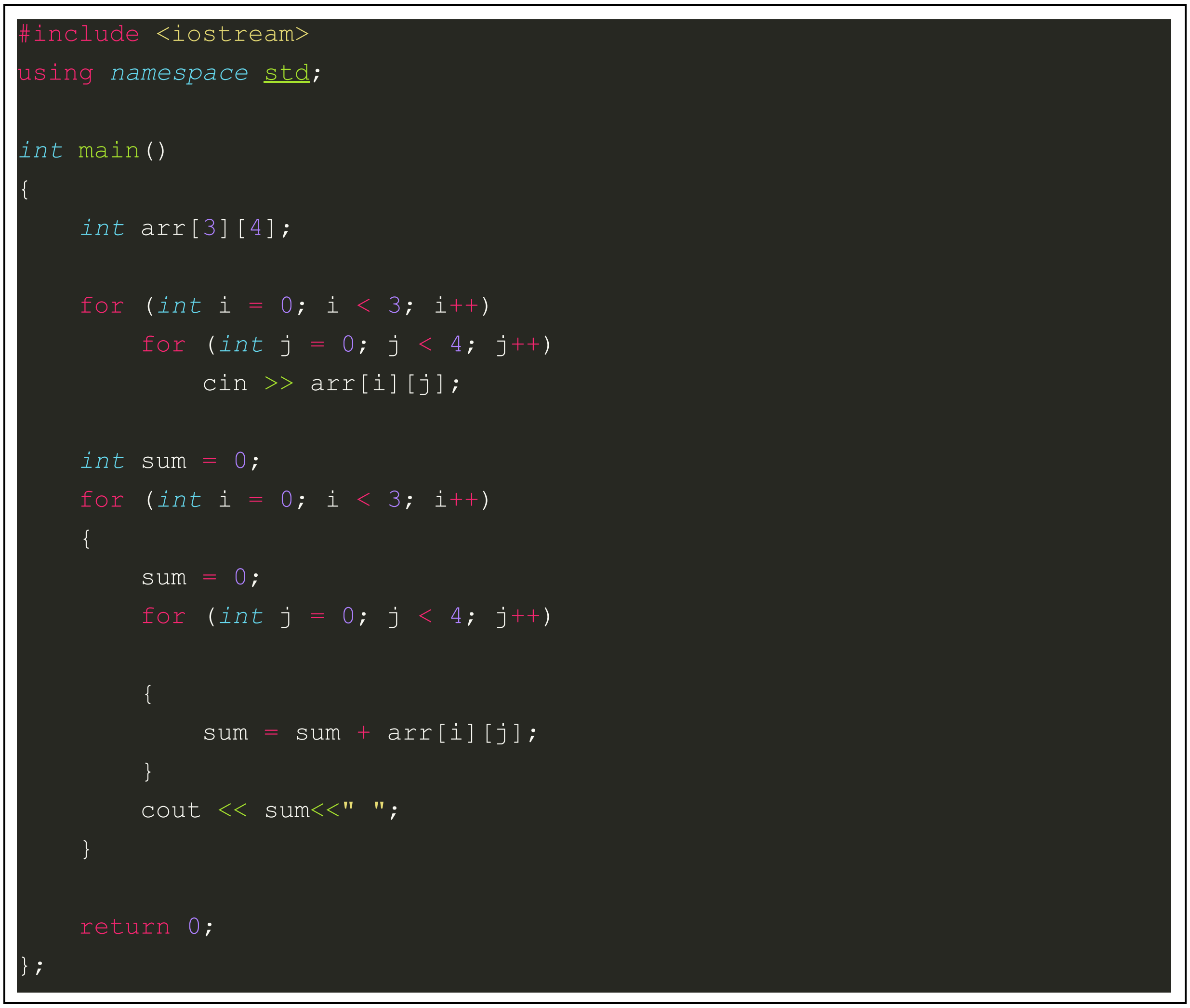
```
    int target = 15;
    for(int i=0; i<3; i++)
    for(int j=0; j<3; j++)
    {
        if(arr[i][j]==target)
        {
            cout<<" Found ";
            return 0;
        }
    }
    cout<<"Not Found";

return 0;
};
```

Print sum ::



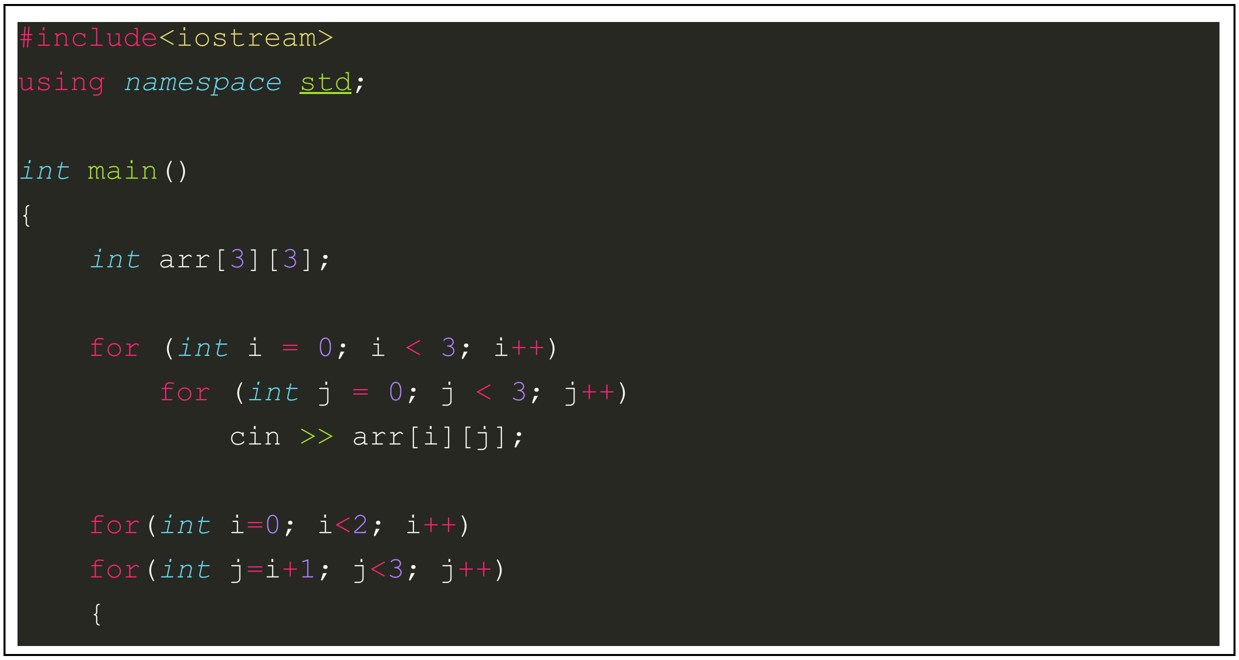
```
#include<iostream>
using namespace std;

int main()
{
    int arr[3][4];

    for(int i=0; i<3; i++)
    for(int j=0; j<4; j++)
    cin>>arr[i][j];

    int sum=0;
    for(int i=0; i<3; i++)
    for(int j=0; j<4; j++)
    {
        sum = sum + arr[i][j];
    }
    cout<<sum;

return 0;
};
```

# Print Array Row Sum ::

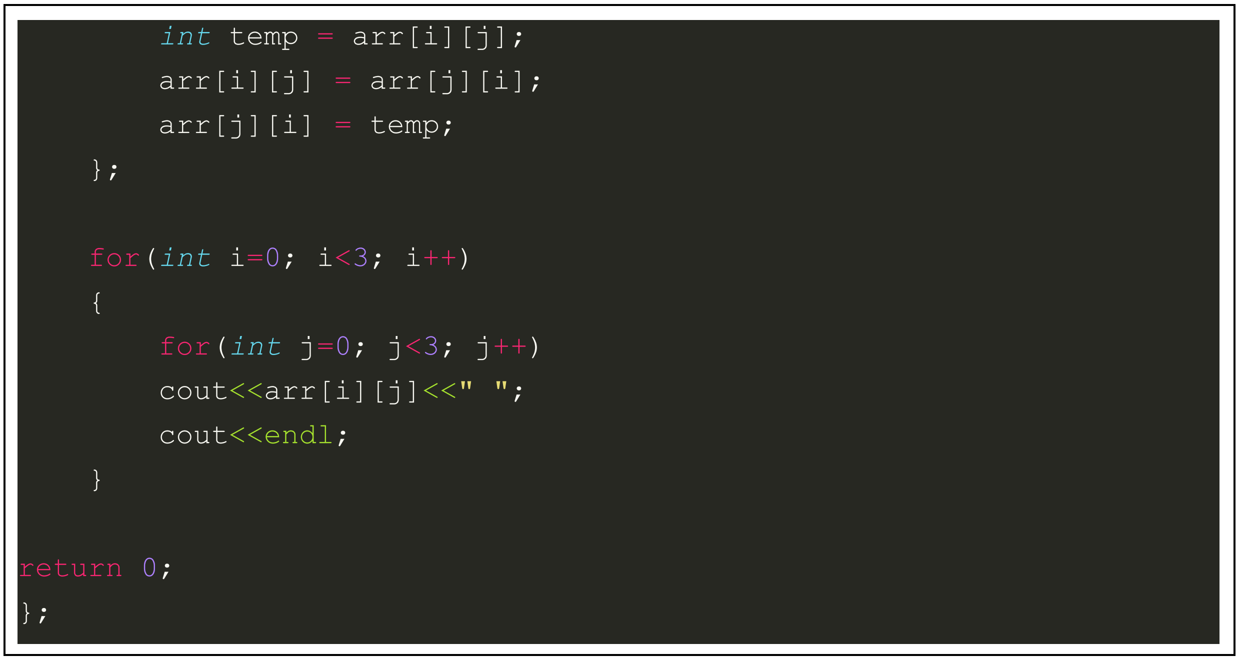


```
#include <iostream>
using namespace std;

int main()
{
    int arr[3][4];

    for (int i = 0; i < 3; i++)
        for (int j = 0; j < 4; j++)
            cin >> arr[i][j];

    int sum = 0;
    for (int i = 0; i < 3; i++)
    {
        sum = 0;
        for (int j = 0; j < 4; j++)

        {
            sum = sum + arr[i][j];
        }
        cout << sum<<" ";
    }

    return 0;
};
```

#Print 2 D array Transpose :-

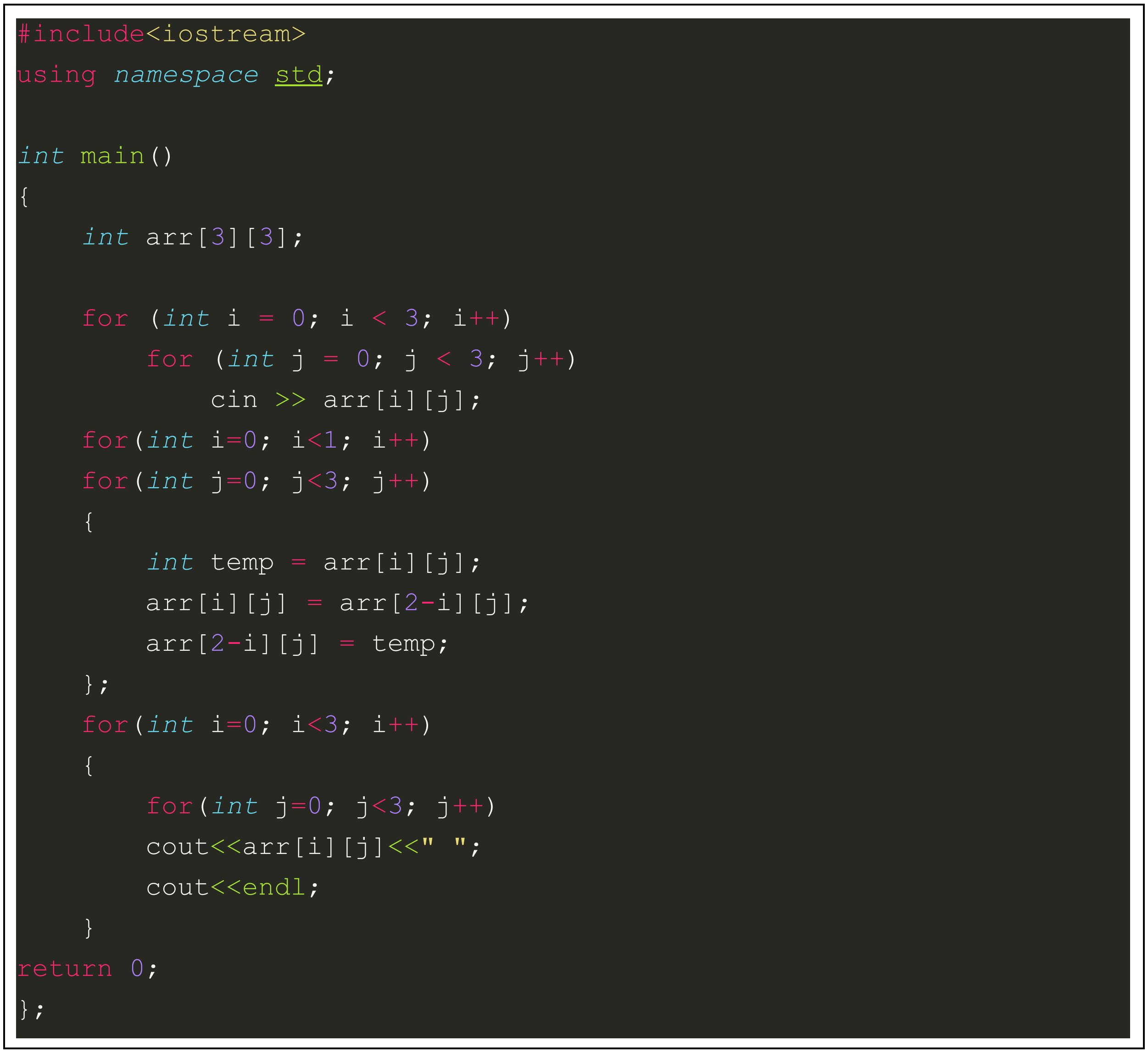


```
#include<iostream>
using namespace std;

int main()
{
    int arr[3][3];

    for (int i = 0; i < 3; i++)
        for (int j = 0; j < 3; j++)
            cin >> arr[i][j];

    for(int i=0; i<2; i++)
    for(int j=i+1; j<3; j++)
    {
```



```
        int temp = arr[i][j];
        arr[i][j] = arr[j][i];
        arr[j][i] = temp;
    };

    for(int i=0; i<3; i++)
    {
        for(int j=0; j<3; j++)
        cout<<arr[i][j]<<" ";
        cout<<endl;
    }

return 0;
};
```

# Print Array Row flip :-



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

int main()
{
    int arr[3][3];

    for (int i = 0; i < 3; i++)
        for (int j = 0; j < 3; j++)
            cin >> arr[i][j];
    for(int i=0; i<1; i++)
    for(int j=0; j<3; j++)
    {
        int temp = arr[i][j];
        arr[i][j] = arr[2-i][j];
        arr[2-i][j] = temp;
    };
    for(int i=0; i<3; i++)
    {
        for(int j=0; j<3; j++)
        cout<<arr[i][j]<<" ";
        cout<<endl;
    }
return 0;
};
```

#Search array 2d matrix rotate 90 degrees?

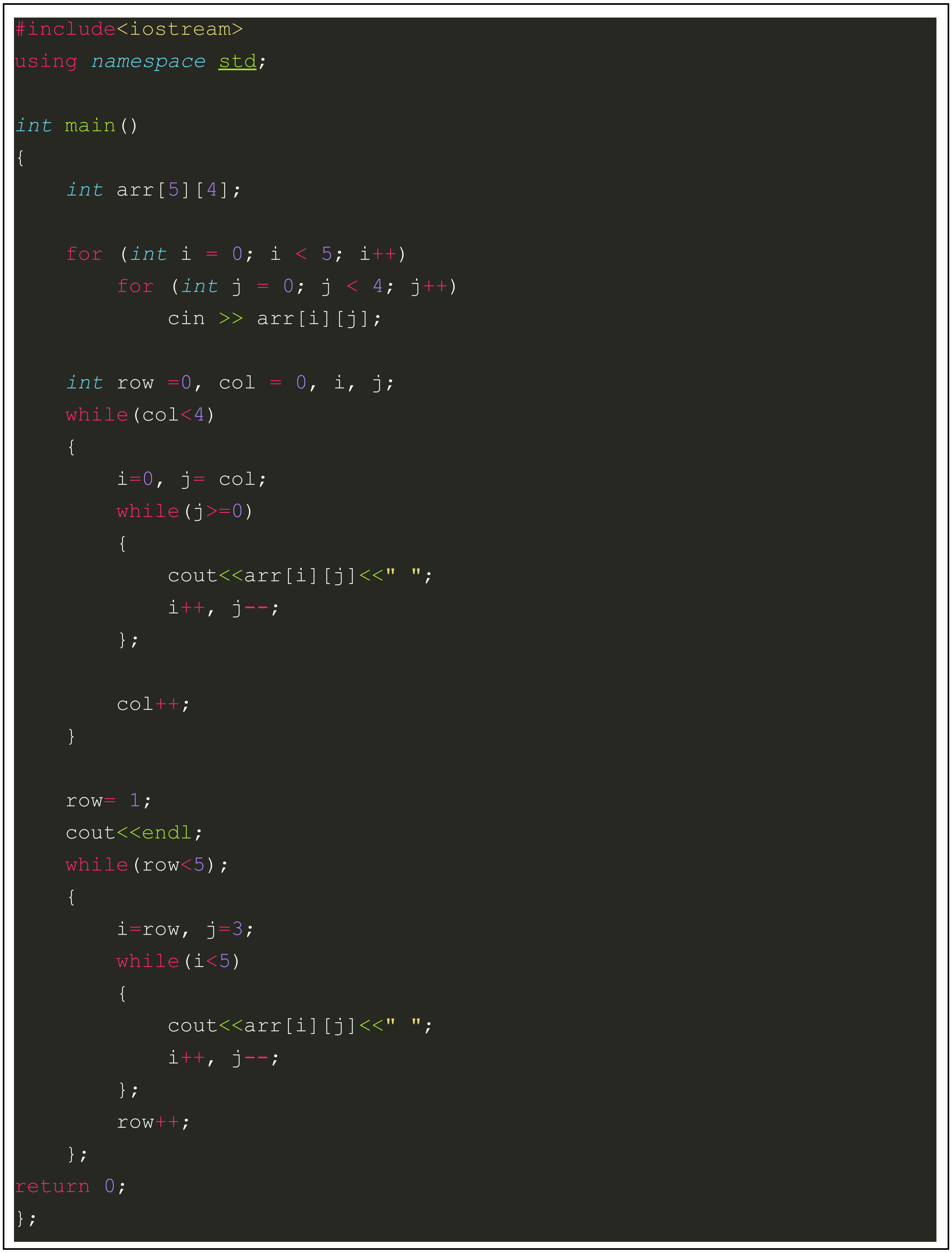
```
class Solution {
public:
    bool searchMatrix(vector<vector<int>>& matrix, int target) {

        int row = matrix.size(), col = matrix[0].size();
        int i= row -1, j=0;

        while(i>=0 && j<col)
        {
            if(matrix[i][j]==target)
            return 1;
            else if (matrix[i][j]<target)
            j++;
            else
            i--;
        };
        return 0;
    }
};
```

LEARN DSA WITH C++

WEEK :: 04 **DAY: 03** **DATE: 10-05-2023**

2 D ARRAY + VECTOR

# Print Diagonally the matrix



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

int main()
{
    int arr[5][4];

    for (int i = 0; i < 5; i++)
        for (int j = 0; j < 4; j++)
            cin >> arr[i][j];

    int row =0, col = 0, i, j;
    while(col<4)
    {
        i=0, j= col;
        while(j>=0)
        {
            cout<<arr[i][j]<<" ";
            i++, j--;
        };

        col++;
    }

    row= 1;
    cout<<endl;
    while(row<5);
    {
        i=row, j=3;
        while(i<5)
        {
            cout<<arr[i][j]<<" ";
            i++, j--;
        };
        row++;
    };
return 0;
};
```

#Find 0 in matrix array

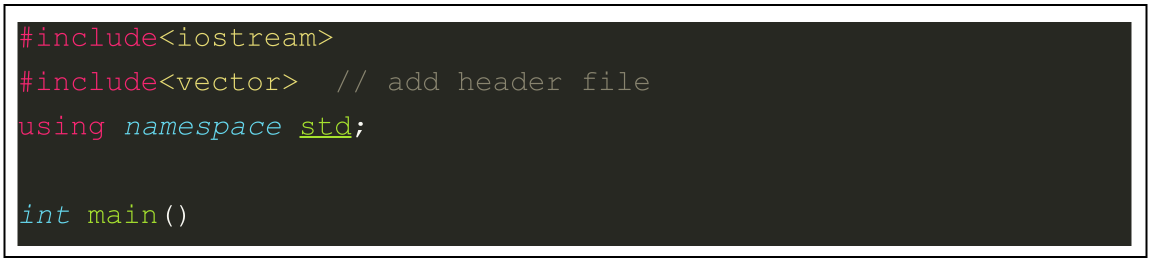
```
class Solution{
public:
        int countZeros(vector<vector<int>>A)
        {
                //code here
                int count =0, row = A.size(), col = A[0].size();
                int i=0, j= col -1;

                while(i<row && j>= 0)
                {
                     while(j>=0 && A[i][j] ==1)
                     j--;

                     count+=j+1;
                     i++;
                }

                return count;
        }
};
```

STL : Standard Template Library

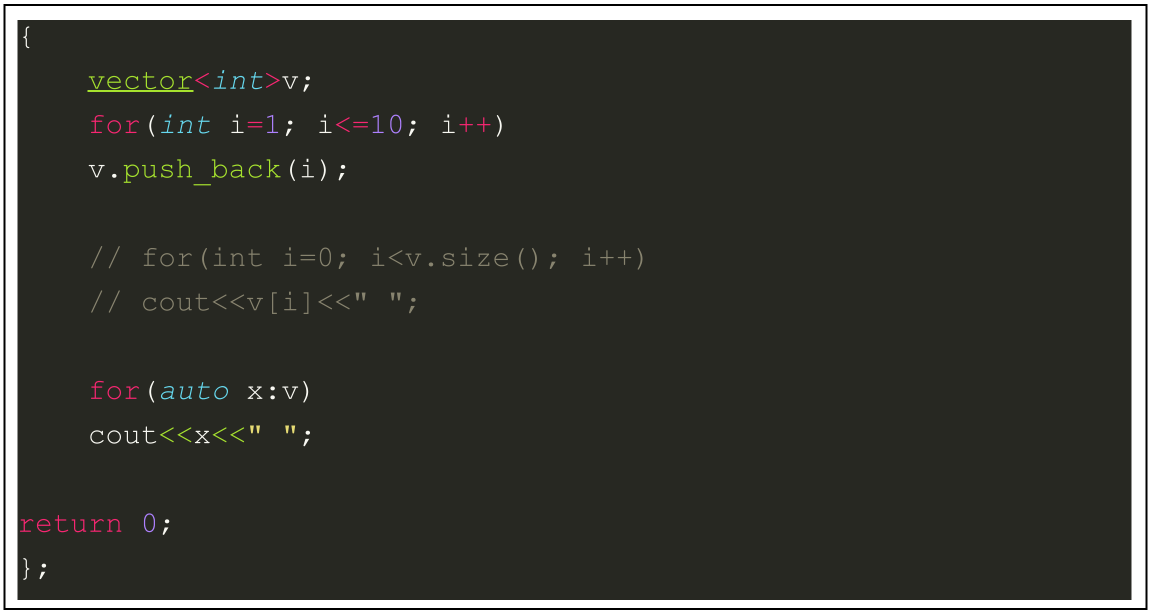
The C++ STL (Standard Template Library) is a powerful set of C++ template classes to provide general-purpose classes and functions with templates that implement many popular and commonly used algorithms and data structures like vectors, lists, queues, and stacks.

# Vector : vector<int> variable_name (array)
# Insert : push-back array.push-back(element) element add in arr
#Delete : pop-back arr.pop-back() element delete in arr
# Size : arr.size( )
# arr[0] : first element in array
# arr[n-1] : arr.back( ) : last element in array
# Clear : arr.clear( ) clear all array elements

#Vector Capacity : 0, 1, 2, 4, 8, 16, 32 —-- increase
# Create Vector ::

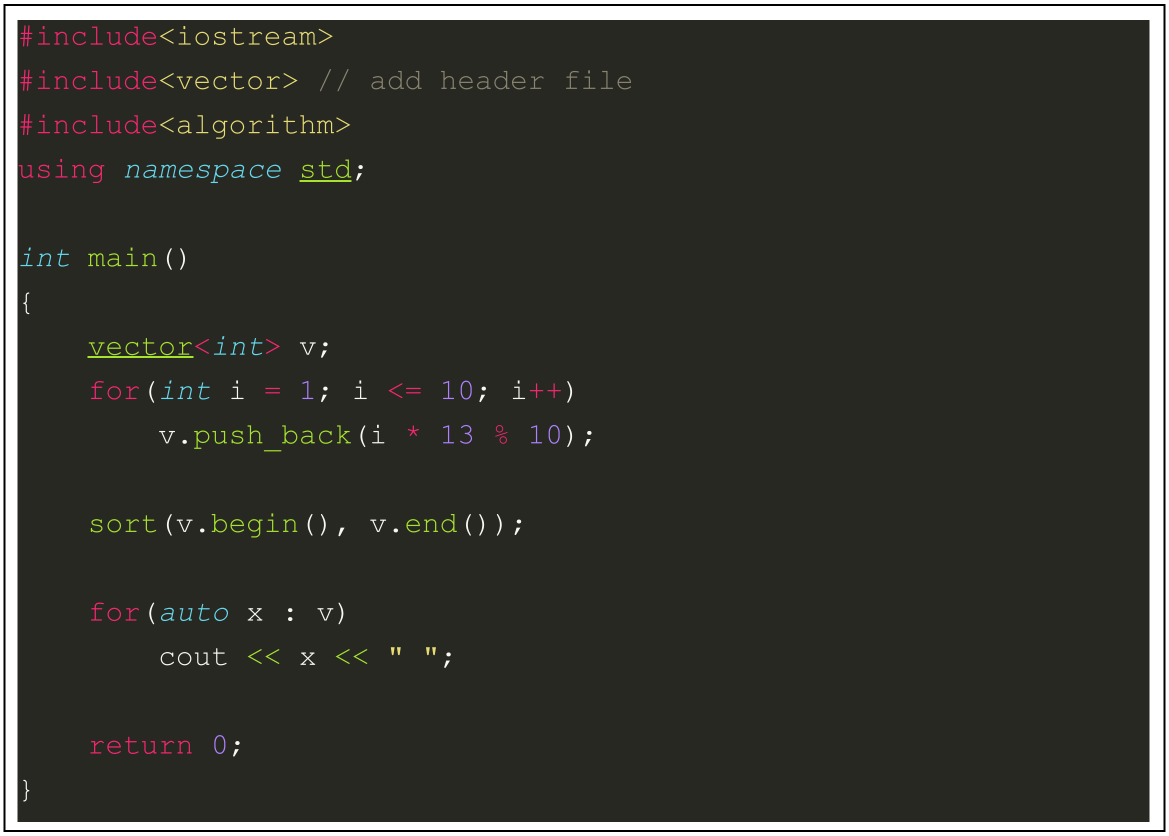


```
#include<iostream>
#include<vector>  // add header file
using namespace std;

int main()
```



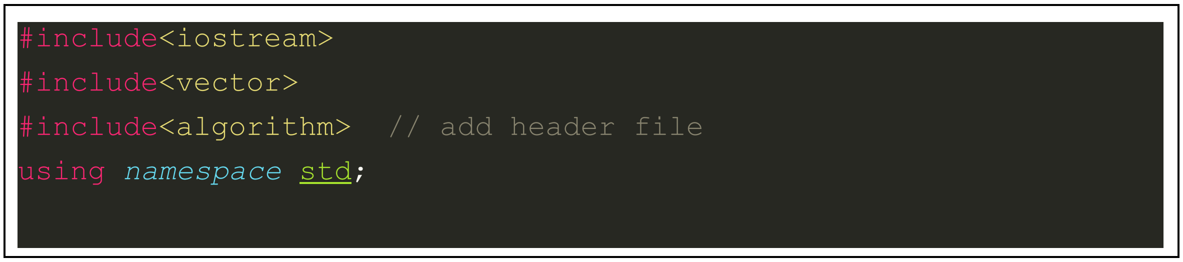
```
{
    vector<int>v;
    for(int i=1; i<=10; i++)
    v.push_back(i);

    // for(int i=0; i<v.size(); i++)
    // cout<<v[i]<<" ";

    for(auto x:v)
    cout<<x<<" ";

return 0;
};
```

#Sorting Algoring using vector :

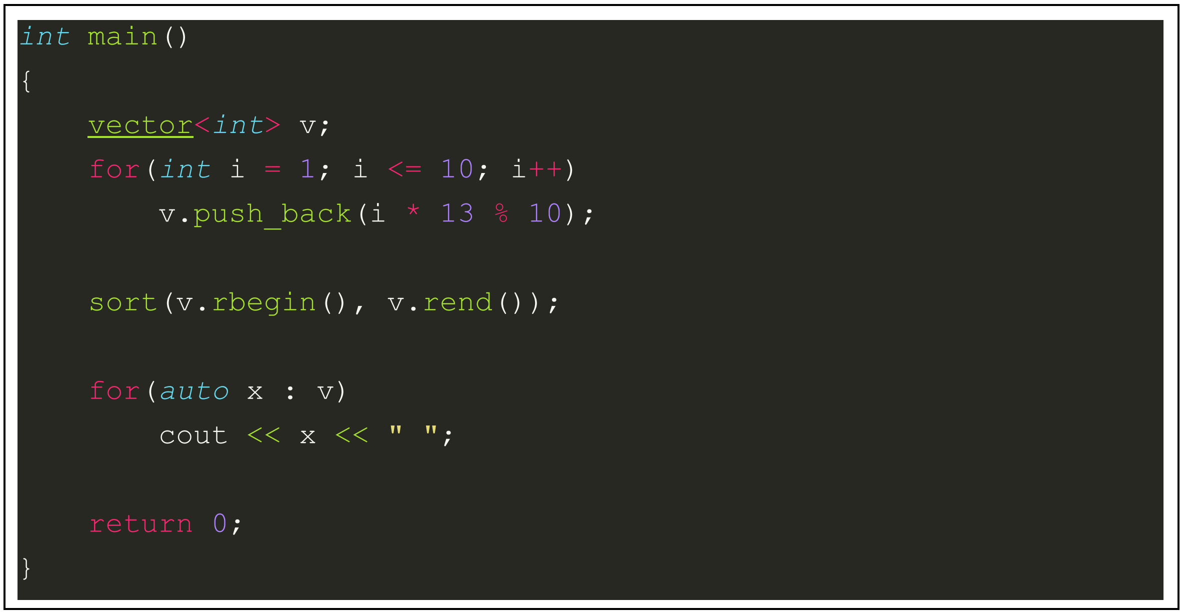


```
#include<iostream>
#include<vector> // add header file
#include<algorithm>
using namespace std;

int main()
{
    vector<int> v;
    for(int i = 1; i <= 10; i++)
        v.push_back(i * 13 % 10);

    sort(v.begin(), v.end());

    for(auto x : v)
        cout << x << " ";

    return 0;
}
```

#Descending order:

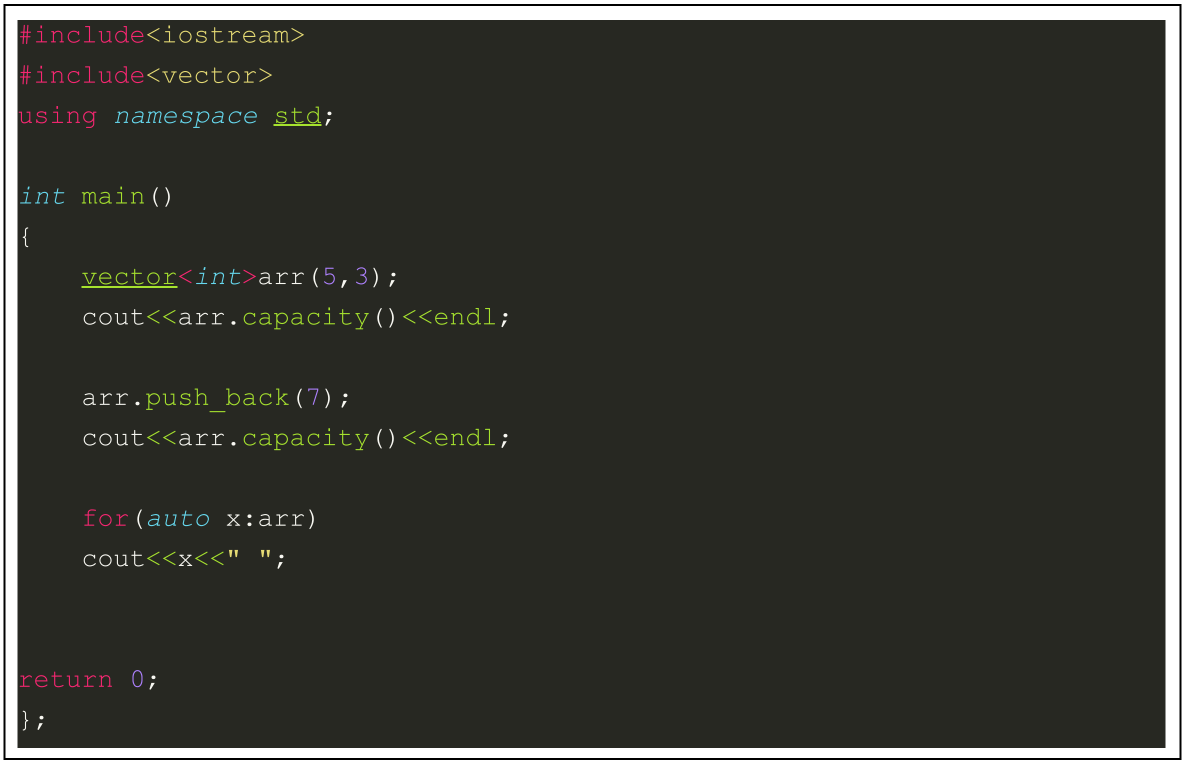


```
#include<iostream>
#include<vector>
#include<algorithm>  // add header file
using namespace std;
```



```
int main()
{
    vector<int> v;
    for(int i = 1; i <= 10; i++)
        v.push_back(i * 13 % 10);

    sort(v.rbegin(), v.rend());

    for(auto x : v)
        cout << x << " ";

    return 0;
}
```

#Define Array in vector:

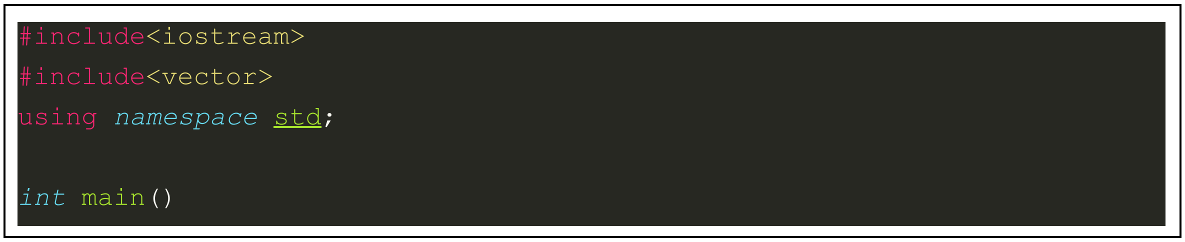


```
#include<iostream>
#include<vector>
using namespace std;

int main()
{
    vector<int>arr(5,3);
    cout<<arr.capacity()<<endl;

    arr.push_back(7);
    cout<<arr.capacity()<<endl;

    for(auto x:arr)
    cout<<x<<" ";


return 0;
};
```

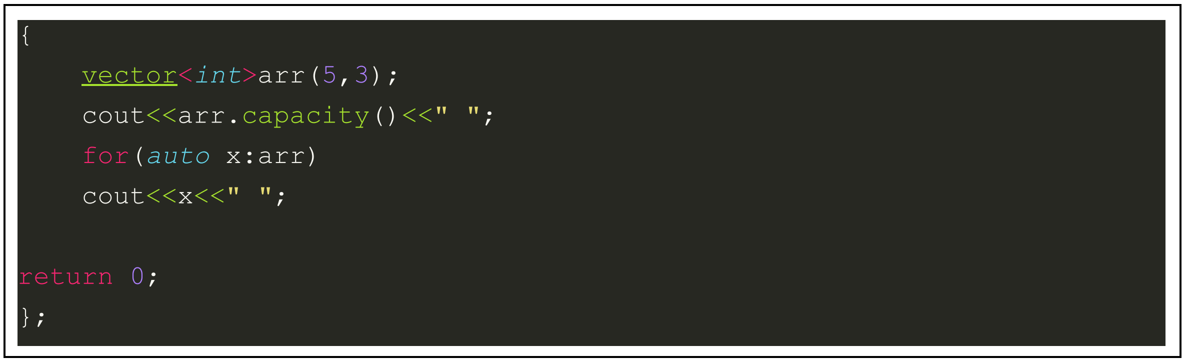
#Initialize in array using vector

vector <int>arr(size, Initialize)

vector <int>arr(5, 2)



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

int main()
```



```
{
    vector<int>arr(5,3);
    cout<<arr.capacity()<<" ";
    for(auto x:arr)
    cout<<x<<" ";

return 0;
};
```

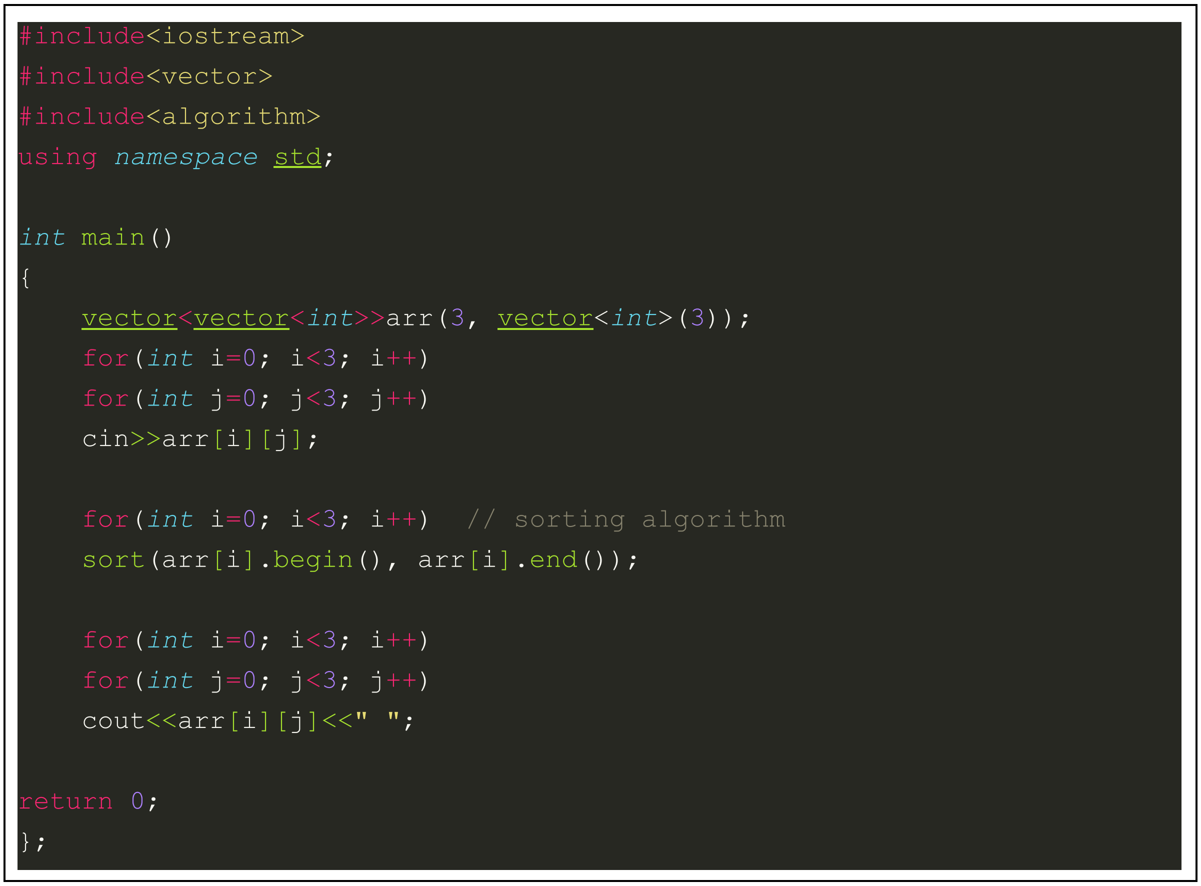
#2D Vector :

No of rows and col:-

Rows = arr.size( );

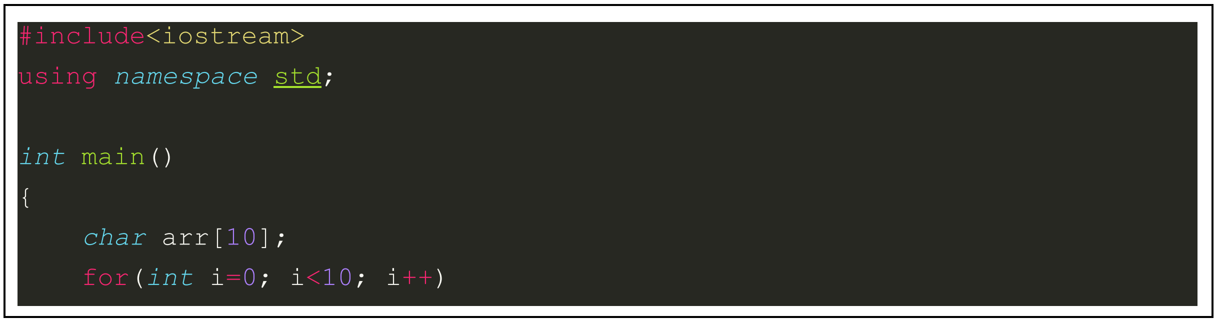
Cols = arr[0].size( );

Total element = rows * col

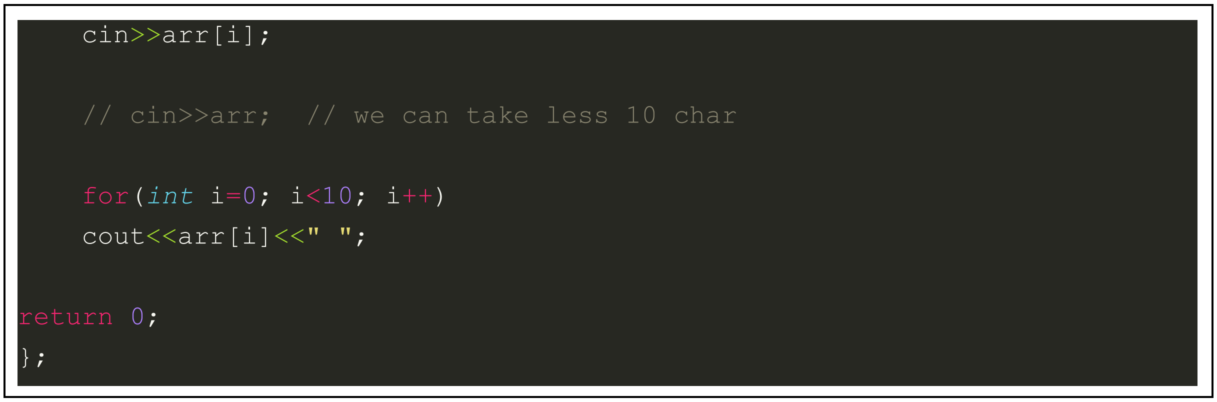


```
#include<iostream>
#include<vector>
#include<algorithm>
using namespace std;

int main()
{
    vector<vector<int>>arr(3, vector<int>(3));
    for(int i=0; i<3; i++)
    for(int j=0; j<3; j++)
    cin>>arr[i][j];

    for(int i=0; i<3; i++)  // sorting algorithm
    sort(arr[i].begin(), arr[i].end());

    for(int i=0; i<3; i++)
    for(int j=0; j<3; j++)
    cout<<arr[i][j]<<" ";

return 0;
};
```

#Character Array :



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

int main()
{
    char arr[10];
    for(int i=0; i<10; i++)
```



```
    cin>>arr[i];

    // cin>>arr;  // we can take less 10 char

    for(int i=0; i<10; i++)
    cout<<arr[i]<<" ";

return 0;
};
```

LEARN DSA WITH C++

WEEK :: 04 **DAY: 04** **DATE: 11-05-2023**

STRING

#Define::

String string-name; input = “pradum”;
cin>>string-name;

Store in memory:

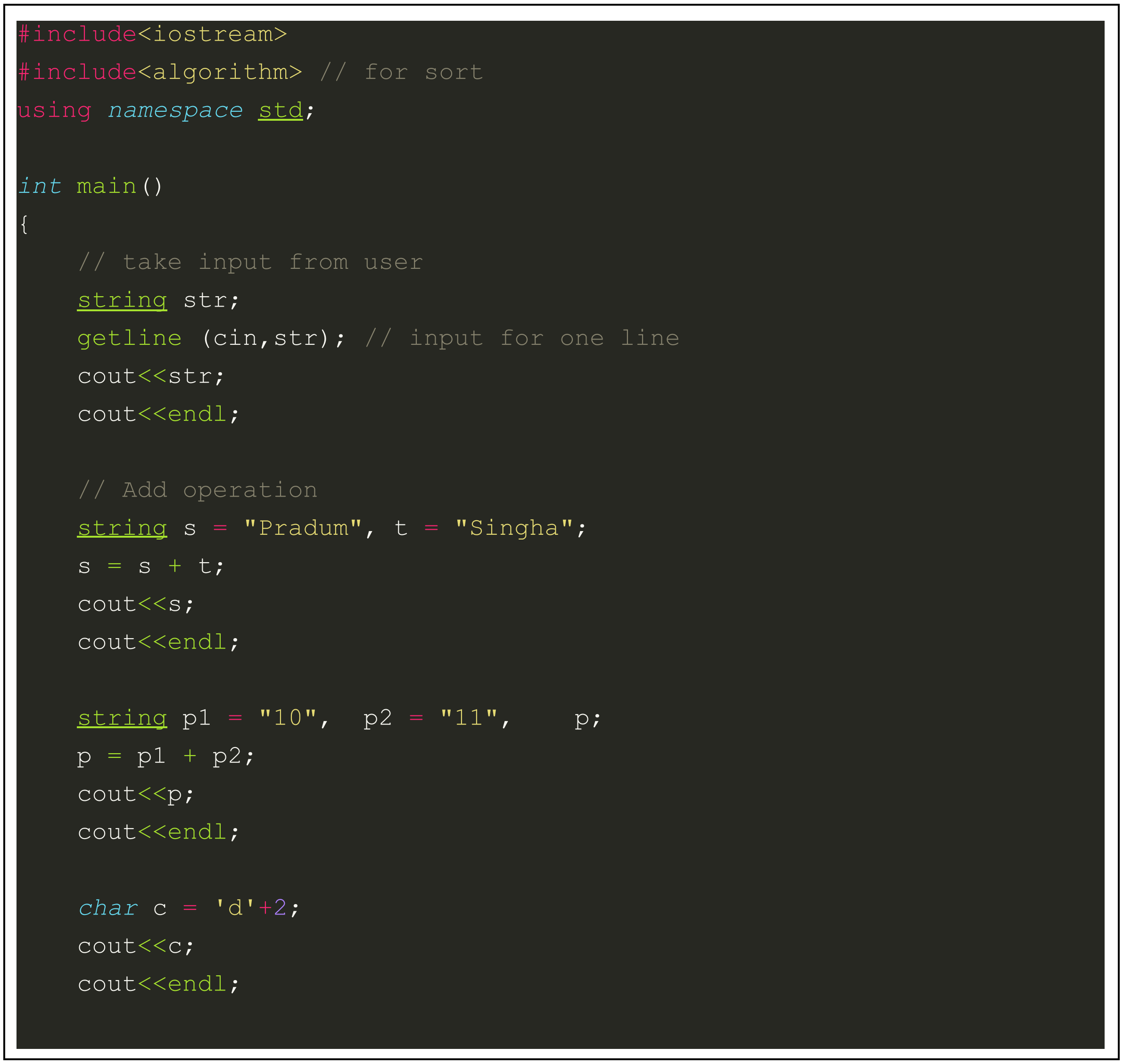
	p	r	a	d	u	m
Index =	0	1	2	3	4	5

Excise[0] = “p”;

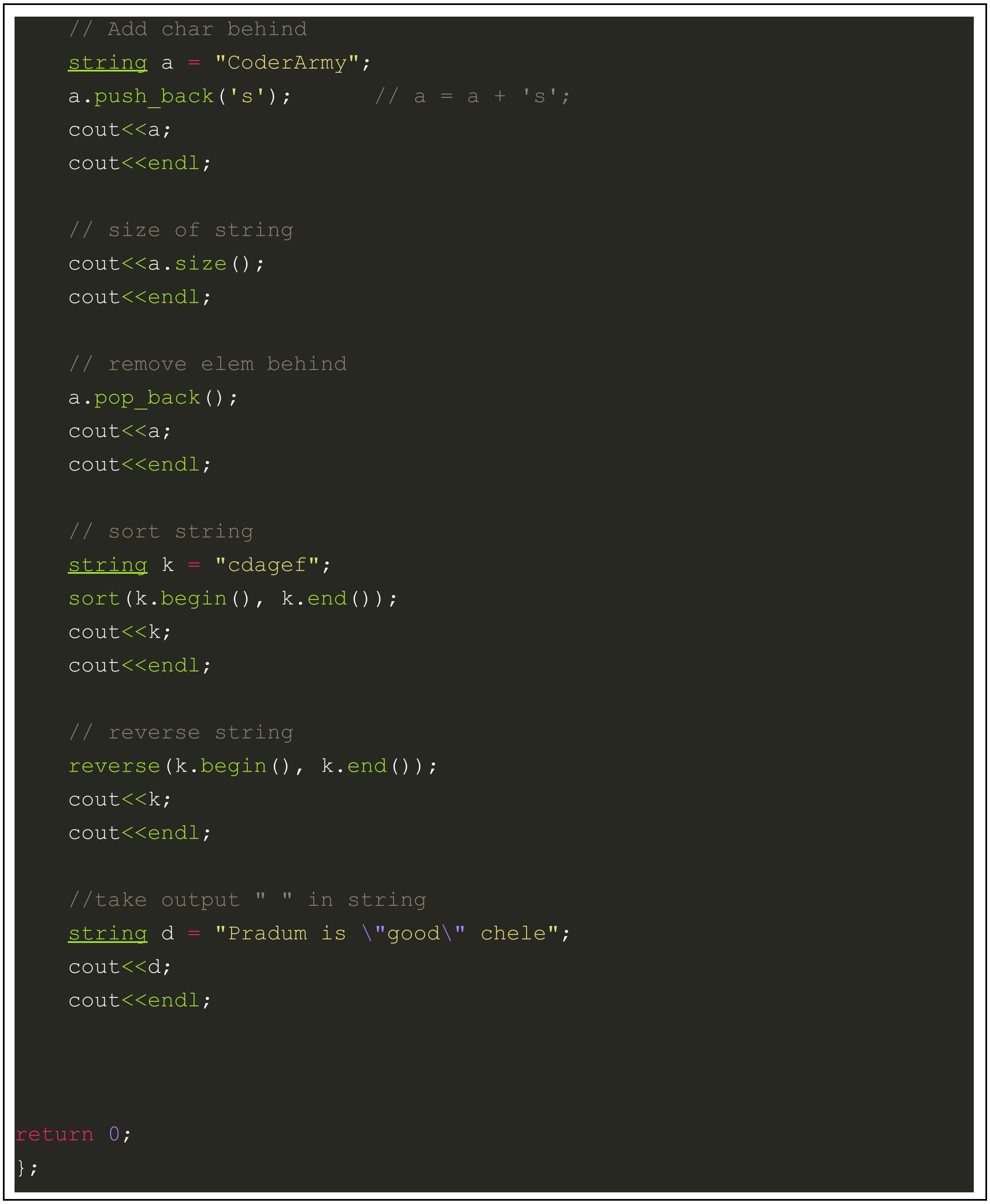
#Operation :

Add:

S = “Pradum”, T = “Singha”;
S = S + T = “Pradum” + “Singha”; = “PradumSingh”;

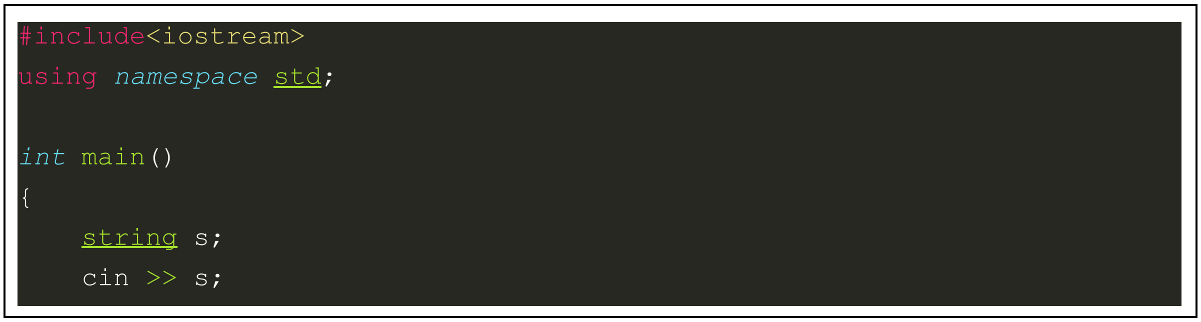


```
#include<iostream>
#include<algorithm> // for sort
using namespace std;

int main()
{
    // take input from user
    string str;
    getline (cin,str); // input for one line
    cout<<str;
    cout<<endl;

    // Add operation
    string s = "Pradum", t = "Singha";
    s = s + t;
    cout<<s;
    cout<<endl;

    string p1 = "10",  p2 = "11",     p;
    p = p1 + p2;
    cout<<p;
    cout<<endl;

    char c = 'd'+2;
    cout<<c;
    cout<<endl;
```



```
    // Add char behind
    string a = "CoderArmy";
    a.push_back('s');       // a = a + 's';
    cout<<a;
    cout<<endl;

    // size of string
    cout<<a.size();
    cout<<endl;

    // remove elem behind
    a.pop_back();
    cout<<a;
    cout<<endl;

    // sort string
    string k = "cdagef";
    sort(k.begin(), k.end());
    cout<<k;
    cout<<endl;

    // reverse string
    reverse(k.begin(), k.end());
    cout<<k;
    cout<<endl;

    //take output " " in string
    string d = "Pradum is \"good\" chele";
    cout<<d;
    cout<<endl;



return 0;
};
```

#sorting string optimization code:



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

int main()
{
    string s;
    cin >> s;
    int n = s.size();
    int freq[26];
    for (int i = 0; i < 26; i++)
        freq[i] = 0;
    for (int i = 0; i < n; i++)
    {
        int index = s[i] - 'a';
        freq[index]++;
    }
    for (int i = 0; i < 26; i++)
    {
        for (int j = 0; j < freq[i]; j++)
        {
            char c = 'a' + i;
            cout << c;
        }
    }

    return 0;
}
```

#Palindrome String :-

```
class Solution {
public:

    int isPalindrome(string S) {
        string K = S;
        reverse(K.begin(),
K.end());

        if (K == S) {
            return 1;
        } else {
            return 0;
        }
    }

};
```

```
class Solution{
public:

      int isPalindrome(string S)
      {
          // Your code goes here
          int start = 0, end =
S.size()-1;

          while(start<end)
          {
              if(S[start]!=S[end])
              return 0;

              start++, end--;
          }
          return 1;
      }

}
```

#Min Number of Flips :-

```
int minFlips(string S) {
    int count1 = 0, count2 = 0;
    bool flag = 0;
    for (int i = 0; i < S.size(); i++) {
        if (flag != S[i] - '0') {
            count1++;
        }
        flag = !flag;
    }
    flag = 1;
    for (int i = 0; i < S.size(); i++) {
        if (flag != S[i] - '0') {
            count2++;
        }
        flag = !flag;
    }
    return min(count1, count2);
}
```

LEARN DSA WITH C++

WEEK :: 04 **DAY: 05** **DATE: 12-05-2023**

STRING IN HARD

#Length of the longest substring:

```
class Solution{
    public:
    int longestUniqueSubsttr(string S){
        //code
        bool count[26];
        for(int i=0; i<26; i++)
        count[i]=0;

        int total = 1, first =0, second = 1;
        count[S[0]- 'a']++;
        while(second<S.size())
        {
            while(count[S[second] - 'a'])
            {
                count[S[first] - 'a'] =0;
                first++;
            }
            count[S[second] - 'a'] = 1;
            total = max(total, second - first +1);
            second++;
        }
        return total;

    }
};
```

#Longest Common Prefix in an Array

```
User function template for C++

class Solution{
  public:

    string longestCommonPrefix (string arr[], int N)
    {
        // your code here
        int count =0;
        int M=INT_MAX;
        for(int i=0; i<N; i++)
        {
            if(M>arr[i].size())
            M=arr[i].size();
        }

        for(int i=0; i<M;i++)
        {
            for(int j=1; j<N; j++)
            {
                if(arr[j-1][i] != arr[j][i])
                {
                    if(count)
                    return arr[0].substr(0, count);

                    else
                    return "-1";
                }
            }
            count++;
        }

        if(count)
        return arr[0].substr(0, count);

        else
        return "-1";
    }
};
```

#Sum of two large numbers

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

class Solution {
  public:
    string findSum(string X, string Y) {
        int Xend = X.size()-1, Yend = Y.size()-1;
        string ans = "";
        int num, rem, carry=0;
        char c;
        while(Xend>=0 && Yend>=0)
        {
            num = X[Xend] - '0' + Y[Yend]-'0'+ carry;
            rem = num%10;
            carry = num/10;
            c = rem + '0';
            ans +=c;
            Xend--, Yend--;
        }

        while(Xend>=0)
        {
            num = X[Xend] - '0' + carry;
            rem = num%10;
            c = rem + '0';
            ans +=c;
            carry = num/10;
            Xend--;
        }

        while(Yend>=0)
        {
            num = Y[Yend] - '0' + carry;
            rem = num%10;
            c = rem + '0';
            ans +=c;
            carry = num/10;
            Yend--;
        }

        if(carry)
            ans+="1";

        int i=ans.size()-1;
        while(i>0 && ans[i] == '0')
        {
            ans.pop_back();
            i--;
        }

        reverse(ans.begin(),ans.end());
        return ans;
    }
};
```