

C++ program for hashing with chaining

In **hashing** there is a hash function that maps keys to some values. But these hashing function may lead to collision that is two or more keys are mapped to same value. **Chain hashing** avoids collision. The idea is to make each cell of hash table point to a linked list of records that have same hash function value.

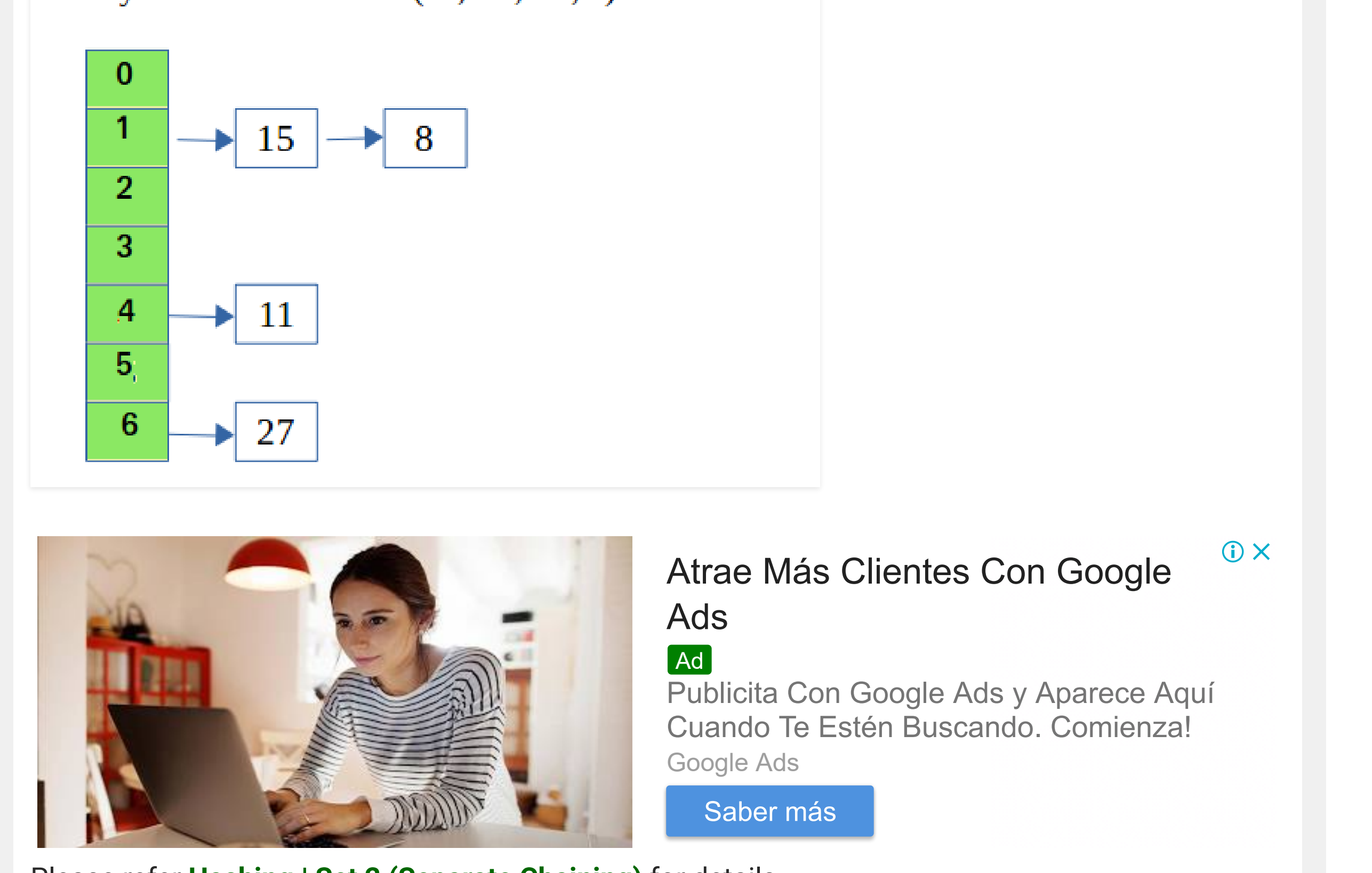
Let's create a hash function, such that our hash table has 'N' number of buckets.

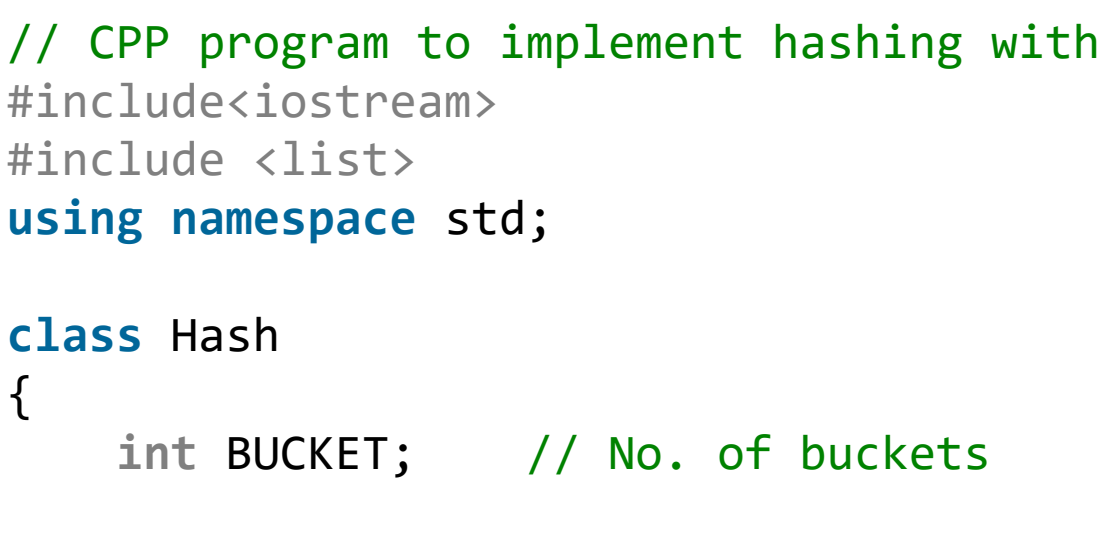
To insert a node into the hash table, we need to find the hash index for the given key. And it could be calculated using the hash function.

Example: hashIndex = key % noOfBuckets

Insert: Move to the bucket corresponds to the above calculated hash index and insert the new node at the end of the list.

Delete: To delete a node from hash table, calculate the hash index for the key, move to the bucket corresponds to the calculated hash index, search the list in the current bucket to find and remove the node with the given key (if found).





Atrae Más Clientes Con Google Ads

Publicita Con Google Ads y Aparece Aquí Cuando Te Estén Buscando. Comienza!

Saber más

Please refer **Hashing | Set 2 (Separate Chaining)** for details.

We use a **list in C++** which is internally implemented as linked list (Faster insertion and deletion).

```
// CPP program to implement hashing with chaining
#include<iostream>
#include <list>
using namespace std;

class Hash
{
    int BUCKET;    // No. of buckets

    // Pointer to an array containing buckets
    list<int> *table;
public:
    Hash(int V);   // Constructor

    // inserts a key into hash table
    void insertItem(int x);

    // deletes a key from hash table
    void deleteItem(int key);

    // hash function to map values to key
    int hashFunction(int x) {
        return (x % BUCKET);
    }

    void displayHash();
};

Hash::Hash(int b)
{
    this->BUCKET = b;
    table = new list<int>[BUCKET];
}

void Hash::insertItem(int key)
{
    int index = hashFunction(key);
    table[index].push_back(key);
}

void Hash::deleteItem(int key)
{
    // get the hash index of key
    int index = hashFunction(key);

    // find the key in (index)th list
    list<int> :: iterator i;
    for (i = table[index].begin();
         i != table[index].end(); i++) {
        if (*i == key)
            break;
    }

    // if key is found in hash table, remove it
    if (i != table[index].end())
        table[index].erase(i);
}

// function to display hash table
void Hash::displayHash() {
    for (int i = 0; i < BUCKET; i++) {
        cout << i;
        for (auto x : table[i])
            cout << " --> " << x;
        cout << endl;
    }
}

// Driver program
int main()
{
    // array that contains keys to be mapped
    int a[] = {15, 11, 27, 8, 12};
    int n = sizeof(a)/sizeof(a[0]);

    // insert the keys into the hash table
    Hash h(7);    // 7 is count of buckets in
                  // hash table
    for (int i = 0; i < n; i++)
        h.insertItem(a[i]);

    // delete 12 from hash table
    h.deleteItem(12);

    // display the Hash table
    h.displayHash();

    return 0;
}
```

Output:

```
0
1 --> 15 --> 8
2
3
4 --> 11
5
6 --> 27
```

- Recommended Posts:
- Hashing | Set 2 (Separate Chaining)

Hashtables Chaining with Doubly Linked Lists

Implementing our Own Hash Table with Separate Chaining in Java

Coalesced hashing

Hashing | Set 1 (Introduction)

Hashing in Java

Double Hashing

Applications of Hashing

Practice Problems on Hashing

Majority Element | Set-2 (Hashing)


Hashing | Set 3 (Open Addressing)

Address Calculation Sort using Hashing

Union and Intersection of two linked lists | Set-3 (Hashing)

Cuckoo Hashing - Worst case O(1) Lookup!

Top 20 Hashing Technique based Interview Questions



shubham_rana_77

Check out this Author's [contributed articles](#).

If you like GeeksforGeeks and would like to contribute, you can also write an article using [contribute.geeksforgeeks.org](#) or mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please Improve this article if you find anything incorrect by clicking on the "Improve Article" button below.

Article Tags : C++ Programs Hash cpp-list

Practice Tags : Hash

4

To-do

Done

3.2

Based on 5 vote(s)

Feedback/ Suggest Improvement

Add Notes

Improve Article

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

Previous

Program to print Swastika Pattern

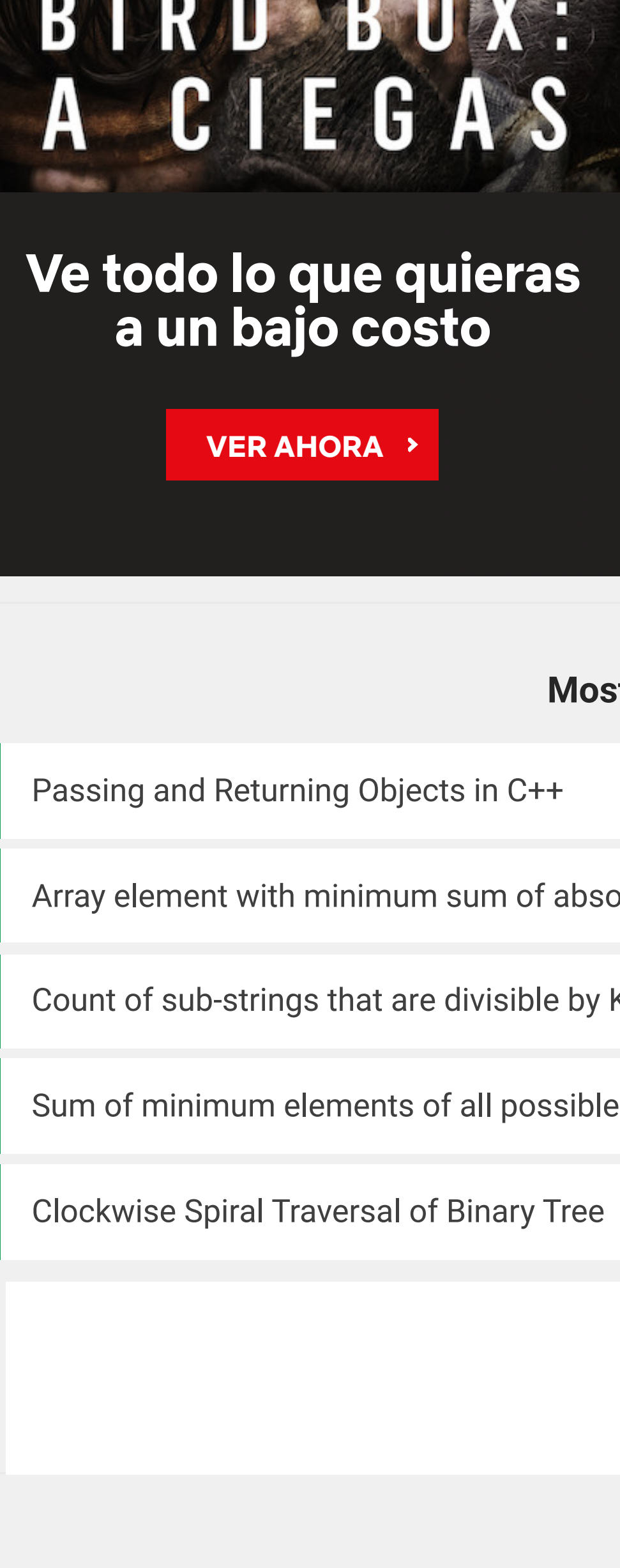
Next

Wishing your Valentine with a Program!!

Writing code in comment? Please use [ide.geeksforgeeks.org](#), generate link and share the link here.

Load Comments

Share this post!



GeeksClasses

Classroom program on DS & Algo in Noida

Mentored by Mr. Sandeep Jain

Batch Starts from 16th Mar, 2019

Register Now

- Most popular in C++ Programs
- Passing and Returning Objects in C++

Array element with minimum sum of absolute differences

Count of sub-strings that are divisible by K

Sum of minimum elements of all possible sub-arrays of an array

Clockwise Spiral Traversal of Binary Tree

- Most visited in Hash
- Remove duplicates from unsorted array

Design a data structure for LRU Cache

Sort elements by frequency | Set 5 (using Java Map)

Hashing in Java

Design a Hit Counter

- Most visited in C++ Programs
- Count all Prime Length Palindromic Substrings

Generate a random permutation of elements from range [L, R] (Divide and Conquer)

MakeFile in C++ and its applications

Determine the count of Leaf nodes in an N-ary tree

Find Maximum and Minimum element in a Set in C++ STL

Minimum operations of given type to make all elements of a matrix equal

C++ program to create a file

Program to implement Linear Extrapolation

Largest sphere that can be inscribed in a right circular cylinder inscribed in a frustum

Generate a random permutation of 1 to N

Nested switch statement in C++

Recursive Program for Binary to Decimal

Geometric Median

Filling diagonal to make the sum of every row, column and diagonal equal of 3x3 matrix

C++ Program to swap two members using Friend Function